



POLISH-JAPANESE ACADEMY OF INFORMATION TECHNOLOGY

Faculty of Computer Science

Department of Multimedia
Multimedia

Antoni Malinowski
Album number s20824

**Design and implementation of an application dedicated to ADHD
people, including strategies for dealing with procrastination,
memory and attention management**

Diploma
Written under the direction
mgr inż. Wichrowski Marcin

Warsaw 2024



POLSKO-JAPOŃSKA AKADEMIA TECHNIK KOMPUTEROWYCH

Wydział Informatyki

Katedra Multimediów
Multimedia

Antoni Malinowski
Numer albumu s20824

**Projekt i realizacja aplikacji dedykowanej osobom ADHD
uwzględniająca strategię radzenia sobie z prokrastynacją,
zarządzaniem pamięcią i uwagą**

Praca inżynierska napisana pod
kierunkiem:
mgr inż. Wichrowski Marcin

Warszawa 2024

Abstract

Attention Deficit Hyperactivity Disorder (ADHD) is a neurodevelopmental disorder characterized by a persistent pattern of inattention, hyperactivity and impulsivity that is more frequent and severe than typically observed in individuals at a comparable level of development.

This thesis presents the design and implementation of an application aimed at assisting ADHD adults¹ in managing daily activities and improving memory management. Given the challenges faced by this population in planning and organization, the application seeks to offer a solution that addresses these needs.

The document presents an overview of the ADHD, evaluates existing market solutions, and identifies their shortcomings. It then introduces a novel approach tailored for ADHD adults, detailing the design, modeling, and implementation phases of the application. The work concludes with a discussion on potential enhancements and a summary of the contributions made towards supporting ADHD individuals in their daily lives.

Keywords: ADHD, mobile application, accessibility, design, software engineering, free software, .NET MAUI, cross platform

¹ Many neurodivergent individuals, including the author, prefer identity-first language as it recognizes their condition as an inherent part of their identity rather than an external "condition" they have. Therefore instead of saying "person with ADHD" (person-first language) the author will use "ADHD person" and other equivalents of this expression throughout the work.

Abstrakt

Zespół nadpobudliwości psychoruchowej z deficytem uwagi (ADHD) to zaburzenie neurorozwojowe charakteryzujące się utrzymującym się wzorcem braku uwagi, nadpobudliwości i impulsywności, które występuje częściej i jest poważniejsze niż zwykle obserwowane u osób na porównywalnym poziomie rozwoju.

Niniejsza praca przedstawia projekt i implementację aplikacji, której celem jest wsparcie dorosłych osób ADHD² w zarządzaniu codziennymi czynnościami oraz usprawnieniu zarządzania pamięcią. Biorąc pod uwagę wyzwania stojące przed tą populacją w zakresie planowania i organizacji, aplikacja ma na celu zaproponowanie rozwiązania, które zaspokoi te potrzeby.

W dokumencie opisano ogólne zapoznanie z ADHD, dokonano oceny istniejących rozwiązań rynkowych oraz wskazano ich wady. Następnie przedstawiono nowatorskie podejście dostosowane do potrzeb dorosłych z ADHD, szczegółowo opisując fazy projektowania, modelowania i wdrażania aplikacji. Praca kończy się dyskusją na temat potencjalnych ulepszeń przedstawionego rozwiązania i podsumowaniem wkładu we wspieranie osób z ADHD w ich codziennym życiu.

Słowa kluczowe: ADHD, aplikacja mobilna, dostępność, projektowanie, inżynieria oprogramowania, wolne oprogramowanie, .NET MAUI, wieloplatformowość

² Wiele neuroroznorodnych osób, w tym autor, preferuje język tożsamościowy, ponieważ uznaje swój stan za integralną część ich tożsamości, a nie zewnętrzną "przypadłość", którą posiadają. Dlatego zamiast mówić "osoba z ADHD" (język stawiający osobę na pierwszym miejscu), autor będzie używał w całej pracy określenia "osoba ADHD" i pozostałych odpowiedników tego określenia.

Introduction.....	9
The structure of the diploma.....	9
Dictionary of technical terms.....	10
1. Theoretical introduction to the topic of ADHD.....	12
1.1. The ADHD brain vs typical brain.....	12
1.2. The 3 types of ADHD.....	14
1.2.1. Inattentive Type.....	15
1.2.2. Hyperactive-Impulsive Type.....	15
1.2.3. Combined ADHD Type.....	16
1.3. ADHD as a spectrum disorder.....	17
1.4. Procrastination.....	17
1.5. Memory management.....	20
1.6. Attention management.....	22
2. Overview of the chosen applications for planning and task structuring available on the market.....	23
2.1. General-audience apps.....	23
2.1.1. Notion.....	24
2.1.2. Any.do.....	25
2.2. ADHD-friendly apps.....	26
2.2.1. Numo ADHD.....	26
2.2.2. Remember The Milk.....	27
3. Author's solution proposal to the presented problem.....	29
3.1. How to approach common ADHD problems?.....	29
3.1.1. Generic functionalities that fulfill strategies for managing procrastination.....	29
3.1.2. Generic functionalities that fulfill strategies for managing memory.....	30
3.1.3. Generic functionalities that fulfill strategies for managing attention.....	30
3.2. Solution proposal to the mentioned problems.....	31
3.2.1. Timer panel.....	31
3.2.2. Notes panel.....	31
3.2.3. Knowledge base panel & Challenges panel.....	32
3.2.4. Routines panel.....	33
3.2.5. Forums panel.....	34
3.2.6. Helpline panel.....	34
3.2.7. Affirmations panel.....	35
3.2.8. Graph tasks module.....	35
3.2.9. Assessment test panel.....	37
3.3. Specialist approved solution.....	38
4. Modeling and analysis of the proposed solution.....	39
4.1. Requirements.....	39
4.1.1. Functional.....	39

4.1.2. Non-functional.....	46
4.2. System actors.....	51
4.3. Diagrams.....	52
4.3.1. Use case diagrams.....	52
4.3.1.1. Screening test module.....	53
4.3.1.2. Attention module.....	54
4.3.1.3. Organization module.....	55
4.3.1.4. People module.....	56
4.3.1.5. Mindfulness module.....	57
4.3.1.6. Graph tasks module.....	58
4.3.2. General activity diagram.....	59
5. Design of the proposed solution.....	61
5.1. Initial design prototype.....	61
5.1.1. Paper prototype.....	61
5.1.2. Digital prototype.....	62
5.2. On-device design.....	65
5.2.1. UI wireframe implementation.....	65
5.2.2. Final design iterations.....	66
5.2.2.1. Navigating to the Attention module (considered as task 0).....	67
5.2.2.2. Creating custom routine.....	69
5.2.2.3. Complete the custom routine.....	72
5.2.2.4. Creating custom note (reminding about deleting the routine).....	74
5.2.2.5. Visiting one of the forums.....	76
5.2.2.6. Deleting created earlier custom routine.....	77
6. Technologies used for the implementation of the project.....	78
6.1. Libre/Free software.....	78
6.1.1. UMLet.....	79
6.1.1.1. Introduction.....	79
6.1.1.2. History.....	79
6.1.2. Git.....	80
6.1.2.1. Introduction.....	80
6.1.2.2. History.....	81
6.1.3. SQLite.....	83
6.1.3.1. Introduction.....	83
6.1.3.2. History.....	83
6.1.3.3. Public domain license.....	85
6.1.3.4. Reasons to choose SQLite.....	85
6.2. Open-source software.....	86
6.2.1. .NET MAUI.....	86
6.2.1.1. Introduction.....	86
6.2.1.2. History.....	87

6.2.1.3. Open-Source, but Not Libre.....	88
6.2.1.4. Cross-Platform Architecture of .NET MAUI.....	90
6.2.1.4.1. .NET Base Class Library (BCL).....	90
6.2.1.4.2. Platform-Specific UI Frameworks.....	90
6.2.1.4.3. Native App Packages.....	91
6.2.1.4.4. C# language.....	92
6.2.1.4.5. XAML.....	92
6.2.1.4.5.1. XAML Hot Reload and Live Visual Tree.....	92
6.2.2. unDraw.....	93
6.2.2.1. Introduction & history.....	93
6.2.2.2. Open-Source, but Not Libre.....	94
6.2.3. Iconoir icons.....	95
6.2.3.1. Open-Source, but Not Libre.....	95
6.3. Proprietary software.....	96
6.3.1. Microsoft Visual Studio IDE.....	96
6.3.1.1. Introduction.....	96
6.3.1.2. History.....	97
6.3.1.3. Proprietary nature.....	98
6.3.1.4. Core features of Visual Studio.....	98
6.3.1.4.1. Integrated Development Environment.....	98
6.3.1.4.2. Cross-Platform Development.....	99
6.3.1.4.3. Extensibility and Customization.....	99
7. Implementation of the project.....	100
7.1. Navigating to the Attention module.....	101
7.2. Navigating between modules.....	106
7.3. Creating custom routine.....	110
7.4. Completing the custom routine.....	120
7.5. Creating custom note.....	124
7.6. Visiting one of the forums.....	129
7.7. Deleting created earlier custom routine.....	133
8. Conducting user tests.....	135
8.1. User group.....	135
8.2. Prepared tasks.....	135
8.3. Results of task completion.....	136
8.4. Questionnaire.....	137
8.4.1. Tasks questionnaire.....	137
8.4.2. General impressions questionnaire.....	138
8.5. Results of the questionnaire.....	140
8.5.1. Tasks questionnaire.....	140
8.5.1.1. Task 1 - Creating custom routine.....	140
8.5.1.2. Task 2 - Completing the custom routine.....	142

8.5.1.3. Task 3 - Creating custom note.....	144
8.5.1.4. Task 4 - Visiting one of the forums.....	146
8.5.1.5. Task 5 - Deleting created earlier custom routine.....	148
8.5.2. General impressions questionnaire.....	151
8.6. Conclusions from the carried out tests.....	154
9. Consideration of future improvements.....	155
9.1. Platform-specific application.....	155
9.2. Implementing suggested changes.....	155
9.3. Maintenance of the project in the spirit of free software.....	155
9.4. Paid access to features.....	155
9.5. Account creation.....	156
10. Summary.....	156
Bibliography.....	157

Introduction

The ability to effectively manage time, memory and attention often comes with a struggle for ADHD adults. These tasks can present significant challenges, often leading to feelings of frustration and underachievement. To fulfill them, ADHD people often require a different approach that is tailored to their needs.

There are different types of ADHD and every patient requires a unique approach. However, there are well-known strategies that tend to be effective when dealing with common ADHD struggles. This paper aims to describe these common struggles and present strategies and techniques for effectively tackling these problems.

As the paper consists of technical documentation of an engineering project, it also presents a proposal of a mobile application implementing functionalities that include those strategies and techniques.

Both the design and implementation is tailored for ADHD adults. As there are well established, universal rules of good design and implementation, the ADHD audience requires even more careful insight into these rules and reconsideration as to what are the most essential elements allowing for inclusivity of users.

The structure of the diploma

The first part is divided into two chapters. It serves as a massive preface to the technical documentation in the later chapters. First chapter defines ADHD on the neurological and psychiatric basis, summarizes the historical timeline of ADHD diagnosis and lists symptoms for the three subtypes of the disorder. Second chapter presents chosen applications for planning, time and memory management available on the market. This chapter serves as a broad overview of available solutions listing their chosen functionalities within the topic of tasks management, memory and attention.

The second part is a single chapter describing the author's proposed solution of an application dedicated to ADHD adults, including strategies for dealing with procrastination, memory and attention management. The chapter explains why the author has decided to pursue this topic and why he believes this application will effectively help ADHD adults. Subchapters list the functions that include strategies and techniques for dealing with the mentioned problems and present the panels to be implemented in the application along with their served functionalities.

The third part documents the process of modeling and analysis, design and implementation of the proposed solution. It also discusses a listing of the technologies used for the technical project - software, programming languages, platforms and tools. Third part ends with the presentation of the results gathered from users testing the implemented application.

Finally, the last fourth part is dedicated to describing future considerations for the application. Possible new features are described and possible plans to improve the application and further research is discussed.

Dictionary of technical terms

Presented below in table 1 is the dictionary of technical terms - full names, their acronyms (if applicable) and definition.

Acronym	Full name	Description
ADHD	Attention Deficit Hyperactivity Disorder	neurodevelopmental disorder characterized by a persistent pattern of inattention, hyperactivity and impulsivity that is more frequent and severe than typically observed in individuals at a comparable level of development.
.NET MAUI	.NET Multi-platform App User Interface	.NET MAUI is a cross-platform framework within the .NET ecosystem that enables developers to create applications for iOS, Android, macOS, and Windows using a single codebase, enhancing performance and user interface consistency.
	free software/libre software	<i>“When we call software ‘free’, we mean that it respects the users' essential freedoms: the freedom to run it, to study and change it, and to redistribute copies with or without changes. This is a matter of freedom, not price, so think of ‘free speech’, not ‘free drink’³. ”[1]</i>
fMRI	Functional magnetic resonance imaging	<i>“a procedure that uses MRI technology to measure and map brain activity by detecting changes in the brain's blood flow and</i>

³ The literal quote uses the term “free beer” instead of “free drink”. The author stands against using terms widely associated with alcoholic beverages, hence the redacted quote.

		<i>oxygenation” [2]</i>
DSM-5-TR	Diagnostic and Statistical Manual for Mental Health Disorders Fifth Edition Text Revision	<i>“comprehensive and critical resource for mental health clinicians and researchers, providing updated diagnostic criteria, codes, and text based on the latest scientific literature” [3]</i>
APA	American Psychological Association	prominent scientific and professional organization representing psychologists in the United States, known for setting rigorous standards, providing guidance, and advancing psychology as a science [4]

Table 1. Dictionary of technical terms

1. Theoretical introduction to the topic of ADHD

This chapter is devoted to a general overview of the ADHD topic. Author's intent is to familiarize the reader with the fundamental terms and knowledge about ADHD. The goal of this chapter should be to introduce the theoretical basis on which the proceeding chapters are built upon and referenced.

1.1. The ADHD brain vs typical brain

Research suggests that differences between typical versus ADHD brains can be found on the structural, functional and chemical basis.

Structural examination of an ADHD brain using the MRI study may result in finding decreased size of prefrontal cortex and basal ganglia [5] Prefrontal cortex is the region that regulates one's behavior, emotions and attention whereas the basal ganglia is responsible for motor learning as well as regulating behavior, emotions, ability to plan focus and multitasking [6]. Figure 1 visualizes the differences in the structure of the two brains.

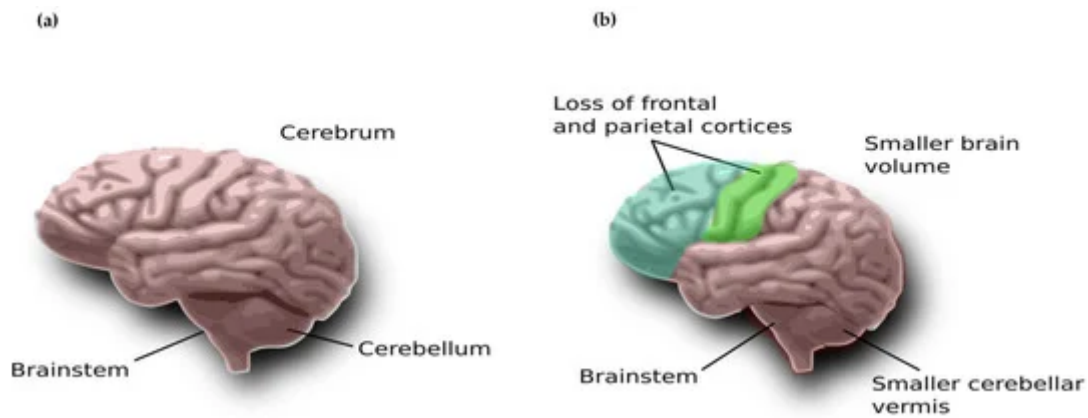


Figure 1. “(a) Normal brain and (b) ADHD brain with smaller volume” [7]

Research using functional magnetic resonance imaging (fMRI) has revealed irregular brain activity in areas involved in motor functions, attention, and executive function in ADHD individuals as shown on Figure 2. Especially the prefrontal area of the brain that houses the executive functions which are responsible for many tasks including planning, organizing, paying attention, remembering, and emotional reactions. *“There are alterations in blood flow to various areas of the brains in ADHD people⁴ compared to non ADHD people, including decreased blood flow to certain prefrontal areas. Decreased blood flow indicates decreased brain activity.”* [8]

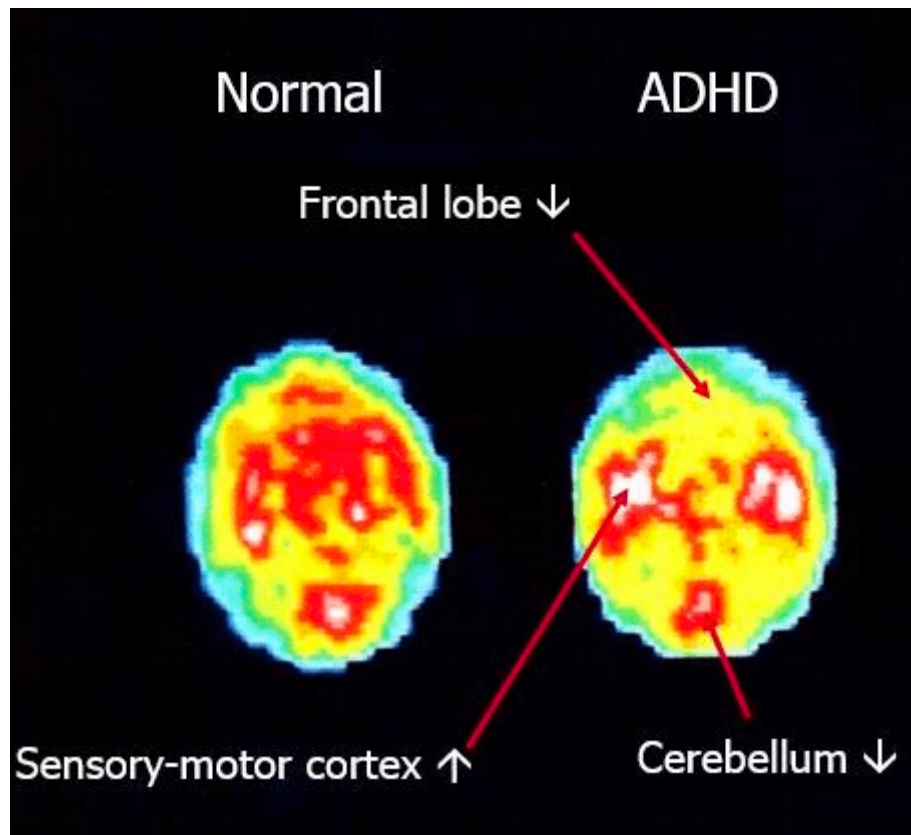


Figure 2. “Scanned Image of Brain Showing Affected Portion due to ADHD” [9]

One of the most significant differences between an ADHD brain versus a normal brain are the levels of neurotransmitters - norepinephrine and dopamine (norepinephrine is synthesized from dopamine). Research shows that individuals with ADHD have lower levels of certain markers related to dopamine in the brain's reward pathway, specifically in areas linked to symptoms of inattention. This suggests that there is a disruption in how dopamine functions in the brain, which is important for attention, motivation, and processing rewards. [10][5]

⁴ The original quote by Jacqueline Sinfield uses person-first terminology and so it was redacted in this paper to use identity-first terminology. The author respects freedom of language and so considers the redaction necessary in the author's own work.

1.2. The 3 types of ADHD

“The American Psychiatric Association’s Diagnostic and Statistical Manual of Mental Disorders (DSM) is a classification of mental disorders with associated criteria designed to facilitate more reliable diagnoses of these disorders.” [11 - page 23]

DSM-5-TR is the first published text revision to DSM-5, which updated the original published DSM-5 for over 70 disorders, one of which is Attention Deficit/Hyperactivity Disorder or ADHD. [11]

As per the manual, ADHD diagnostic criteria specifies whether it is classified as:

- Predominantly inattentive presentation (F90.0⁵)
- Predominantly hyperactive/impulsive presentation (F90.1)
- Combined presentation (F90.2)

Despite such a clear distinction, the ADHD stereotype depicts a young hyperactive boy who cannot control his impulses [13]. This depiction may have been so widespread because for a very long time the access to diagnosis was unattainable or it was considered unnecessary or even harmful as it would talk patients and their parents into believing something that is only really laziness or insubordination.

To give a bit of context as to how much of a developing research area this disorder is, the next paragraphs summarize the timeline of ADHD.

ADHD was first mentioned in 1902 but only in 1968 it was published in the DSM-2 and recognized as a disorder. In fact, in DSM-2 it was referred to as hyperkinetic reaction of childhood. In 1980 The APA released a third edition of the DSM (DSM-III) where they changed the name of the disorder to attention deficit disorder (ADD). At that time, scientists believed hyperactivity was not a common symptom of the disorder.

This listing created two subtypes of ADD: ADD with hyperactivity, and ADD without hyperactivity.

Up until 1987, ADD was diagnosed in ADHD people (as currently considered) for which the hyperactivity and impulsivity symptoms occurred less frequently than for others.

In 2000 when the APA released the fourth edition of the DSM, the three subtypes of ADHD were established. [14]

Luckily, today it is very much possible for an ADHD person to both get diagnosed and adapt to live meaningfully. As mentioned previously, ADHD is classified into three main presentations, widely referred to as types. The following subchapters describe these types as they currently stand and present the common symptoms associated with each.

⁵ ICD-10-CM codes, part of the International Classification of Diseases, Tenth Revision, Clinical Modification (ICD-10-CM), are a system of diagnosis codes used to represent various health conditions, diseases, abnormal findings, signs and symptoms, injuries, and external causes of injuries and diseases. These codes are essential for medical claim reporting in all healthcare settings and play a crucial role in establishing medical necessity for payment of healthcare services and procedures. [12]

1.2.1. Inattentive Type

According to the Diagnostic and Statistical Manual for Mental Health Disorders Fifth Edition Text Revision (DSM-5-TR), the common symptoms of Inattentive Type are the following [11 - page 69]:

1. Often makes careless mistakes
2. Difficulty sustaining attention
3. Does not seem to listen when in a conversation
4. Difficulty following instructions
5. Struggles in organization
6. Does not prefer tasks that require sustained mental effort
7. Loses things frequently
8. Gets easily distracted and has trouble paying attention
9. Being forgetful which can affect daily functioning

“Predominantly inattentive presentation may be diagnosed if six (or more) of the mentioned symptoms have persisted for at least 6 months to a degree that is inconsistent with developmental level and that negatively impacts directly on social and academic/occupational activities, and at the same time, criterion for the Hyperactivity and impulsivity symptoms have not been met for the past 6 months.” [11 - page 69]

1.2.2. Hyperactive-Impulsive Type

According to the Diagnostic and Statistical Manual for Mental Health Disorders Fifth Edition Text Revision (DSM-5-TR), the common symptoms of Hyperactive-Impulsive Type are the following [11 - page 70]:

1. Often fidgets with hands, feet or squirms in seat
2. Difficulty remaining still
3. Excessive running or ascending
4. Often unable to play or engage in leisure activities quietly
5. Often struggles to keep still for extended periods of time
6. Tends to talk excessively
7. Often blurts out an answer before a question has been completed
8. Struggles to wait his or her turn
9. Interrupts others

“Predominantly hyperactive/impulsive presentation may be diagnosed if six (or more) of the mentioned symptoms have persisted for at least 6 months to a degree that is inconsistent with developmental level and that negatively impacts directly on social and academic/occupational activities, and at the same time, criterion for the Inattention symptoms have not been met for the past 6 months.” [11 - page 70]

1.2.3. Combined ADHD Type

Combined Type can be diagnosed if the criteria for symptoms of both types has been met for the past 6 months.

It is crucial to note that severity of the official symptoms can vary for every individual and that there are many symptoms that are not being clinically recognized as such but may be an effect of the same neurological cause as the official symptoms. Many ADHD individuals report experiencing symptoms not listed in the DSM-5, such as:

1. sleeping difficulties
2. time blindness
3. hyperfocus and hyperfixation
4. sensory sensitivity
5. social awkwardness
6. emotional dysregulation

These symptoms are unofficial not because they have been medically debunked but because they are currently not a part of the diagnostic criteria for ADHD [15].

1.3. ADHD as a spectrum disorder

While diagnostic thresholds are useful for identifying neurodivergent individuals⁶ some researchers argue that the traditional classification into subtypes may not fully capture the dimensional nature of ADHD.

One study mentioned that the DSM-5 criteria for ADHD, although allowing for severity classifications, may not accurately reflect the diverse presentations and developmental trajectories of the disorder. The study suggests that future revisions should consider other nosological devices to better represent the dimensionality of ADHD and its impact on overall functioning. [16]

Another study draws from diverse fields such as taxonomy, epidemiology, genetics, neurobiology, and neuropsychology. The main message of the research indicates that accumulating evidence supports ADHD being predominantly dimensional rather than qualitatively distinct. This perspective does not disregard the clinical needs of individuals with significant ADHD symptom clusters but emphasizes the lack of discontinuity in the disorder, suggesting a more nuanced understanding of ADHD as existing along a spectrum rather than in discrete categories. [17]

1.4. Procrastination

“Experts define procrastination as a self-defeating behavior pattern marked by short-term benefits and long-term costs.” It is often characterized as putting off things that we need to get done, no matter the level of difficulty behind the task. [18]

Procrastination is a prevalent challenge for ADHD individuals, often rooted in difficulties with executive functions like planning, time management, and prioritization. ADHD can make it harder to start tasks, especially those perceived as uninteresting or overwhelming. This delay is not due to laziness but rather to how ADHD affects the brain's processing and motivation systems. [19]

As described in chapter 1.1 *The ADHD brain vs typical brain*, the brain of an individual with ADHD is structurally and functionally different, particularly in areas responsible for executive function and impulse control. These differences play a pivotal role in the tendency toward procrastination:

- ➔ **Executive Function Impairment:** The executive functions, which include planning, prioritizing, and task initiation, are often impaired in ADHD [20]. Such an impairment makes starting and staying on task challenging, leading to procrastination.
- ➔ **Dopamine and Reward Systems:** ADHD brains have an altered dopamine reward pathway [21]. This alteration means that the brain may not receive the usual 'feel good' signals from task completion, making mundane tasks feel unrewarding and thus easy to avoid.

⁶ Neurodivergence is a broader term than ADHD itself but the sentence still holds true.

As there are neurological causes there are also triggers that may lead to procrastination:

- ➔ Task Initiation Struggles - For an ADHD person, starting a task can be the most significant hurdle. The brain's executive dysfunction makes transitioning from intention to action exceptionally challenging as if there is a literal wall a person cannot break through.
- ➔ Overwhelm and Decision Fatigue - ADHD individuals may experience overwhelm when faced with complex tasks or decisions, leading to procrastination as a coping mechanism.
- ➔ Fear of Failure and Perfectionism - The fear of not meeting high standards or making mistakes can trigger procrastination in ADHD individuals, as they may avoid tasks to prevent perceived failure.
- ➔ Impulsivity and Distraction - The ADHD brain is more susceptible to impulsivity and distraction. These factors contribute to procrastination, as person often find perself⁷ sidetracked by other stimuli or tasks, deviating from their original intentions.
- ➔ Time Tracking Challenges - Difficulty in estimating time accurately and keeping track of deadlines can contribute to procrastination in ADHD individuals. Misjudging the time needed for a task can result in last-minute rushes and increased stress.
- ➔ Procrastivity: faced by the overwhelm of actually doing a task person is supposed to be doing, person might divert our attention towards more manageable and predictable goals, thus effectively procrastinating. [23]

Procrastination is not an official symptom of ADHD (as it is not present in the DSM-5-TR [11 - page 69-77]) but is often regarded as a coping strategy when faced with difficult, unpleasant or uneasy tasks [24]. That being said, to minimize triggers causing ADHD and act on the issue when the procrastination occurs, there are strategies and techniques for dealing with procrastination, some of which are:

1. Break projects into actionable microtasks - Divide tasks into smaller, more manageable, atomic parts to make them less overwhelming, easier to grasp and start
2. Prepare an organizational system - Organizing tasks in the form of notes or to-do lists helps in visualizing the steps required to complete a task, structures execution order of each step and prepares a ahead-of-time plan of essential steps and items needed to perform a task. It helps with decision fatigue, structures actions ahead of time and clearly defines the goal of the task.
3. Overcome perfectionism - Recognize that perfectionism can contribute to procrastination by setting unrealistically high standards that can be paralyzing. Embrace a mindset of progress over perfection, focusing on completing tasks to the best of your ability rather than aiming for flawless outcomes.
4. Set reminders - Setting notifications on performing tasks may help in breaking the time blindness and triggering the action. Making gratification reminders of what has

⁷ The author has adopted a method of making English gender-neutral while still distinguishing between singular and plural as suggested by Richard Stallman [22]. For that reason the author uses the gender-neutral pronouns of “perself” that come from “person” or “pers” or “perse” in short, to refer to an individual when it is needed.

been accomplished and recognizing the completed work may also motivate a person into continuing the plan and provide dopamine boost.

5. Plan the work according to the productivity cycles - Setting work blocks during periods of optimal productivity is a great way to adapt to one's natural cycle. People have different habits of working hours and recognizing what time is optimal may help in managing attention, focus, memory and mental capabilities.
 6. Act according to set routines - Managing new activities by inserting them into already existing routines is the optimal way of incorporating new habits. By building upon existing routines a person can easier incorporate new habits, captivate his focus and attention effectively and ease everyday activity by performing blocks of tasks as if they were single, automatic tasks.
 7. Organize work surrounding - Organizing work surrounding gives the sense of control and minimizes distractions. Decluttering workspace provides a sense of accomplishment, control of circumstances and creates a hygienic habit of being able to take care of oneself and the workspace.
 8. Minimize distractions - Create a workspace free from distractions to enhance focus and productivity. Eliminate unnecessary items, reduce noise, and establish boundaries to maintain concentration on tasks.
 9. Kickstart the task with short mini-task - Begin a task by tackling small, manageable components to build momentum and overcome the initial hurdle of starting a larger project.
 10. Change tasks when work becomes ineffective - Recognize when a task is no longer productive and switch to a different task to maintain efficiency, prevent stagnation and frustration for not progressing with the task as effectively as before.
 11. Forgive perself for procrastination - Practice self-compassion and understanding towards moments of procrastination, acknowledging that it is a common challenge and not a reflection of personal failure.
 12. Reward perself for completing a task - Celebrate accomplishments, no matter how small, to reinforce positive behavior and motivate continued progress. Positive reinforcement can help reinstantiate productive behavior and motivate to continue the task.
 13. Rest and keep the head sharp - Prioritize rest and self-care to maintain mental clarity and prevent burnout, ensuring optimal cognitive function for task completion. Ensure adequate sleep, breaks during work sessions, and relaxation activities to recharge and optimize productivity.
 14. Remember about the overall goal - Keep the ultimate objective in mind to stay motivated and focused on the bigger picture, guiding actions towards long-term success and fulfillment. Reminding oneself of the purpose behind tasks can provide clarity, direction, and a sense of a bigger picture for which the task is being done.
 15. Gamification - Incorporating a gaming element into a task may increase engagement and passion for completing it. Using such a competitive approach may help in clearly defining goals, keeping track of the progress and introducing enjoyment into the process which itself minimizes the chances of boredom and burn out from the task.
- [25][26]

1.5. Memory management

To accurately describe memory management issues in the ADHD paradigm it is necessary to first define differences between the types of memory. There are the main types of memory:

1. Short-term memory - *“Short-term memory (or short-term storage; the two are often used interchangeably) refers to retention of information in a system after information has been categorized and reached consciousness. In fact, contents of short-term memory are sometimes equated with the information of which a person is consciously aware. Information can be continually processed in short-term storage (e.g., via rehearsal or subvocal repetition). If a person is distracted, information is rapidly lost from this store.”* [27]
2. Working memory - *“Working memory is the small amount of information that can be held in mind and used in the execution of cognitive tasks, in contrast with long-term memory, the vast amount of information saved in one’s life. It facilitates planning, comprehension, reasoning, and problem-solving.”* [28]
3. Long-term memory - Long-term memory refers to the transfer of information from short-term memory into long-term storage in order to create enduring memories. This type of memory is unlimited in capacity and stable—lasting for years or even a lifetime. Short-term memories can become long-term memories through a process known as consolidation. [29]

The differences between long-term, short-term, and working memory are a subject of considerable discussion in the literature. Short-term memory (STM) and working memory (WM) are distinct theoretical concepts that reflect different cognitive functions. STM is a system for holding sensory events, movements, and cognitive information briefly, with an average capacity of around seven or four chunks of information. On the other hand, WM, as proposed by Baddeley and Hitch, is a more complex system that acts as an interface between LTM and action, involving multiple components like the phonological loop and visuospatial sketchpad. It may be said that STM is a simpler system for temporary storage, while WM is a more intricate cognitive system that not only holds information temporarily but also manipulates it for various cognitive tasks. [30]

As described in chapter 1.1 *The ADHD brain vs typical brain*, because the ADHD brain is fundamentally different it plays a crucial role in the source of memory management problems. Research suggests association between ADHD and impaired working memory capacity, altered brain regions activation during executive function tasks, and slower brain maturation compared to healthy individuals.

The same way struggles with procrastination can be minimized by adopting appropriate strategies and techniques, issues with memory management can be dealt with by following certain rules:

1. Keep a calendar - writing down important dates, events, plans and obligations relieves the memory from such duty. People may struggle with remembering dates regardless of ADHD and so having ADHD increases this struggle. Wanting to keep one's memory sharp may come in different practice than remembering the date. Saving the time in a physical or virtual calendar guarantees a reliable place for keeping track of all the events.
2. Keep a notebook - For all the other notes that need to be remembered temporarily it is advisable to keep a notebook. Such notes should not save important dates and events but should rather serve as external short-term storage for all-things that come to mind and are not to be forgotten temporarily. List of groceries for the day or a reminder to walk the dog out should be placed there. These notes can be easily erased as soon as the task gets completed or at the end of the day.
3. Keep an organized list of tasks for the day - A person should plan the following day with a clearly structured schedule of what will be done that day. The schedule may change and be slightly dynamic but it is important to wake up with a preset goal for the day to avoid confusion and feeling lost.
4. Setting visual reminders - Visual notifications aid with reminding of what has been planned. Setting visual reminders captures attention of an ADHD person and may prevent them from forgetting to perform a certain action or losing an item. Commands should be straight-forward and visually attractive so that they capture a person's attention.
5. Structuring the task - as mentioned previously, tasks should be broken down into atomic commands to be better understood, easy to follow and remember. A well structured task should be comprehensible after a single read but also easy to go back to without losing the context.
6. Visualizing the end result - defining a clear end result of an action is very important. Firstly, it gives a clear understanding of what is being accomplished. Secondly, it serves as an intuitive reminder of the preceding steps. Defining a clear goal also gives the sense of control over one's decisions and ruling and may be additionally motivating.

1.6. Attention management

“Attention is the ability to actively process specific information in the environment while tuning out other details.”[31]

In his 1890 book “The Principles of Psychology,” psychologist and philosopher William James wrote that attention *"is the taking possession by the mind, in clear and vivid form, of one out of what may seem several simultaneously possible objects or trains of thought. Focalization, concentration, of consciousness are of its essence. It implies withdrawal from some things in order to deal effectively with others[...]"* [32].

As described in chapter 1.1 *The ADHD brain vs typical brain*, the brain of an ADHD individual is different on multiple bases. For years now, researchers have been pointing to the neurological basis of attention impairments in people with ADHD [33]. That being said, there are established strategies and techniques suited for people with ADHD:

1. Break down tasks into atomic actions - Simple and clear commands are easier to understand, remember and follow. Simple steps also provide a clear distinction between proceeding and the next action and minimize ambiguity of action to be performed eg. place all dishes on the lowest shelf of the dishwasher.
2. Set clear, unambiguous goals - To understand what is the final outcome of the action a person should clearly name each of the steps as well as the main goal of this list of steps.
3. Set time blocks - Reserving time slots for activities aids in structuring the day and limits wasting too much time on a single project. It should also provide a clear division between working hours and leisure hours in order to optimize effectiveness and minimize burn out.
4. Reduce distractions - Taking care of a clear environment helps to minimize distractions and harvest effective focus.
5. Make use of the flow effect - Noticing and harvesting on the flow effect (strict focus on the task) may be the most effective way to capture attention with mindfulness. Realizing being focused and consciously utilizing that effect brings satisfaction from such conscious control.
6. Practice mindfulness - Making conscious effort of reminding oneself of the current state is important to verify if the course of action is right and that this is a process of conscious progress. Celebrating progress and learning from failures is important for a healthy mental state and to further proceed with the effort.

2. Overview of the chosen applications for planning and task structuring available on the market

Market is saturated with self-help apps. Simple web search provides countless results for applications designed for structuring projects, making to-do lists, saving events in calendar form, writing notes, mindfulness etc. It is hard to compete with richness and advancement of their features. However, most of them are designed to fit the needs of the majority of clients (understandable), hence they fulfill the needs of a typical person. In this chapter, the author presents an overview of those general-audience apps as well as neurodivergent-friendly apps - specifically ADHD-suited. This overview will shortly describe the chosen apps, present their main features and lists pros and cons of them in respect to the needs of an adult with ADHD as discussed in chapters 1.4 through 1.6.

2.1. General-audience apps

General-audience apps have been designed for the general audience. They target a broad majority of people. This often makes them feature-rich and reliable for most types of usages - individual or team needs. However, their richness may also make them too complex for users requiring special care. They may be not suited for specific needs of ADHD adults and so confusing, complex and in result ineffectively used.

2.1.1. Notion

Notion is a versatile productivity and note-taking application offering a wide range of organizational tools, including task management, project tracking, to-do lists, and bookmarking as shown on Figure 3.

- a. some of the features:
 - i. Visualize workflow and tasks (Kanban board)
 - ii. Create lists for tasks and projects
 - iii. Schedule and track events and deadlines
 - iv. Visualize tasks and projects over time
 - v. Plan and track project timelines
 - vi. Track time spent on tasks
 - vii. Generate reports on tasks and projects
- b. pros:
 - i. Great for notes & to-do lists - Helps in organizing thoughts and tasks
 - ii. Good project management - Provides tools for managing projects effectively
 - iii. Provides clear structure for tasks and projects
 - iv. Very feature-rich - provides a lot of possibilities
 - v. Very customizable
 - vi. Offers free basic plan
- c. cons:
 - i. Proprietary software - the app restricts user from per freedom
 - ii. Requires registering an account with personal data
 - iii. May pose challenges in finding information quickly
 - iv. Design may be polarizing
 - v. Very feature-rich - may be overwhelming to start and poses a thread of perfectionism paralysis
 - vi. Extra features are behind paywall
 - vii. Writing long content may be cumbersome due to the app's structure

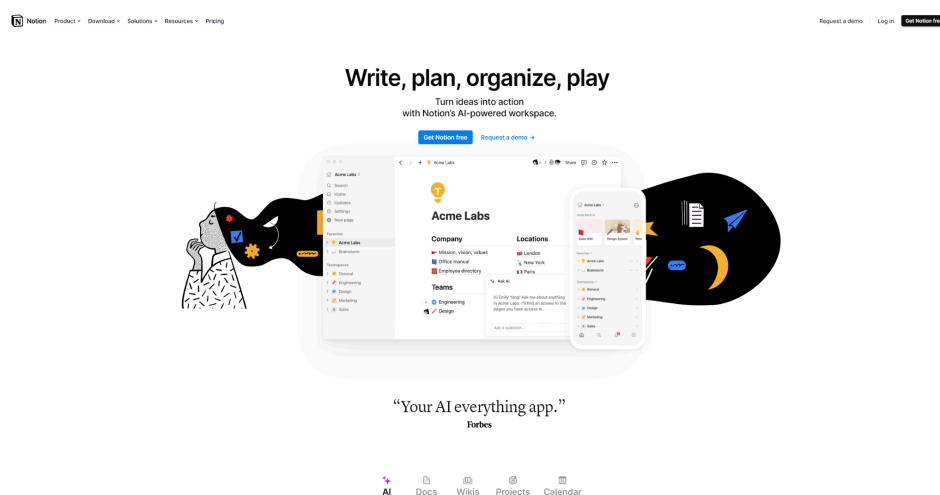


Figure 3. Notion app homepage [34]

2.1.2. Any.do

Any.do is a comprehensive productivity and task management application offering a wide range of organizational tools, including to-do lists, calendar integration, reminders, and collaboration features as shown on Figure 4.

- a. some of the features:
 - i. Task management with to-do lists, reminders, and notes
 - ii. Daily planner with customizable tasks and events
 - iii. Calendar integration for seamless scheduling
 - iv. Voice entry for hands-free task creation
 - v. Collaboration tools for sharing lists and assigning tasks
 - vi. Syncs across devices (mobile, desktop, web)
 - vii. Subtasks and priority settings for detailed task management
- b. pros:
 - i. User-friendly Interface - Easy to navigate and use, suitable for all types of users.
 - ii. Comprehensive Features - Combines to-do lists, calendar, and reminders in one app.
 - iii. Collaboration Tools - Allows sharing and assigning tasks to others, great for teams.
 - iv. Cross-platform Sync - Syncs across mobile, desktop, and web for seamless access.
 - v. Voice Entry - Convenient for quick, hands-free task creation.
 - vi. Location-based Reminders - Useful for tasks tied to specific places.
- c. cons:
 - i. Proprietary software - the app restricts user from per freedom
 - ii. Requires registering an account with personal data
 - iii. May pose challenges in finding information quickly and in effect cause overwhelm
 - iv. Design may be polarizing
 - v. Very feature-rich - may be overwhelming to start and poses a thread of perfectionism paralysis
 - vi. Extra features are behind paywall

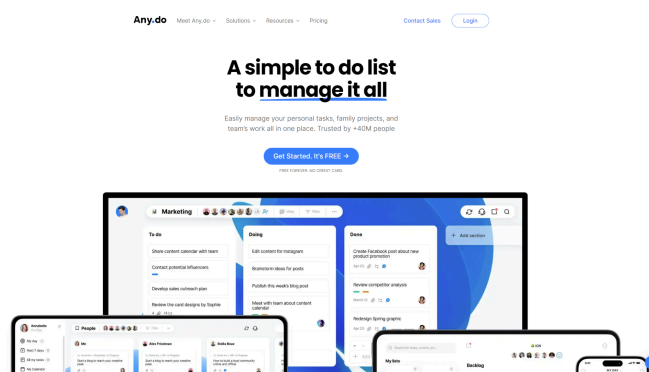


Figure 4. Any.do app homepage [35]

2.2. ADHD-friendly apps

ADHD-friendly apps are designed with the unique needs of ADHD individuals in mind. These apps offer features that cater to the challenges faced by those who are neurodivergent, such as difficulty with focus, organization, and time management. They often include tools for habit-building, mood tracking and personalized reminders, making them highly effective for managing ADHD symptoms as shown on Figure 5.

2.2.1. Numo ADHD

Numo is a specialized app designed to help ADHD individuals manage their symptoms through personalized tips, habit-building tools, mood tracking, and customizable planning features (figure 5).

- a. some of the features:
 - i. Personalized tips and strategies for managing ADHD symptoms
 - ii. Daily planner with focus on building healthy habits
 - iii. Mood tracking and journaling to monitor emotional well-being
 - iv. Goal setting with progress tracking
 - v. Reminders and notifications for task completion
 - vi. Customizable interface to suit individual needs
 - vii. Insights and analytics to understand patterns and improve productivity
- b. pros:
 - i. Tailored for ADHD - Specifically designed with ADHD users in mind, offering relevant tips and strategies.
 - ii. Beautiful and welcoming design
 - iii. Habit-building Focus - Helps users build and track healthy habits.
 - iv. Mood Tracking - Allows users to monitor their emotional well-being.
 - v. Personalization - Customizable interface to meet individual preferences and needs.
 - vi. Insightful Analytics - Provides data to understand patterns and improve productivity.
 - vii. Included forum of fellow ADHD people
- c. cons:
 - i. Proprietary software - the app restricts user from per freedom
 - ii. Requires registering an account with personal data
 - iii. Too many “instructive popups” can be overwhelming
 - iv. The app has so many hidden features that it may be hard to navigate through.
 - v. Limited General Use - Less suitable for users without ADHD, as it's tailored specifically for ADHD needs.
 - vi. Subscription Cost - Some features may require a paid subscription.
 - vii. Learning Curve - Might take some time for users to fully utilize all features effectively.

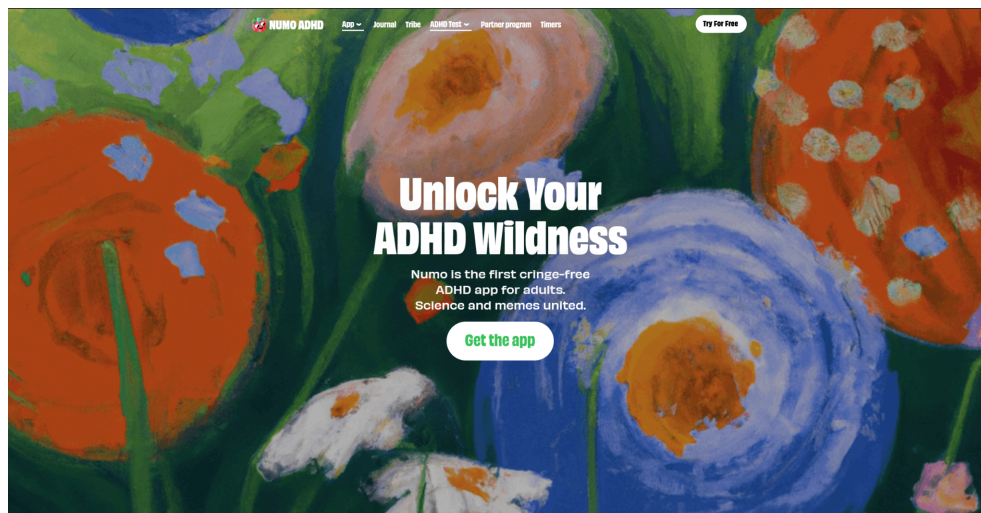


Figure 5. Numo app homepage [36]

2.2.2. Remember The Milk

Remember The Milk is a task management and to-do list application that helps users organize their tasks with features like smart lists, tags, priority settings, and multiple reminder options, while offering seamless integration with popular services and multi-platform support as shown on Figure 6.

- a. some of the features:
 - i. Task management with simple, intuitive interface
 - ii. Smart lists and tags for easy organization and retrieval
 - iii. Priority settings and due dates to manage tasks effectively
 - iv. Reminders via email, text, and mobile notifications
 - v. Integration with popular services like Gmail, Google Calendar, and Evernote
 - vi. Offline access and syncing across devices
 - vii. Subtasks for breaking down complex tasks
 - viii. Collaboration features for sharing lists and tasks with others
- b. pros:
 - i. Simplicity - Intuitive and straightforward interface, easy to use.
 - ii. Customization - Smart lists and tags allow for personalized organization.
 - iii. Multi-platform Support - Accessible on various devices and syncs seamlessly.
 - iv. Integration - Works well with popular services like Gmail and Google Calendar.
 - v. Offline Access - Can access tasks even without an internet connection.
 - vi. Comprehensive Reminders - Offers multiple reminder options to ensure tasks are completed.

c. cons:

- i. Proprietary software - the app restricts user from per freedom
- ii. Requires registering an account with personal data
- iii. Basic Design - The interface may feel outdated or too simple for some users.
- iv. Premium Features - Advanced features require a paid subscription.
- v. Limited Collaboration - Collaboration features are not as robust as some competitors.
- vi. Subtask Complexity - Managing complex subtasks can be less intuitive.

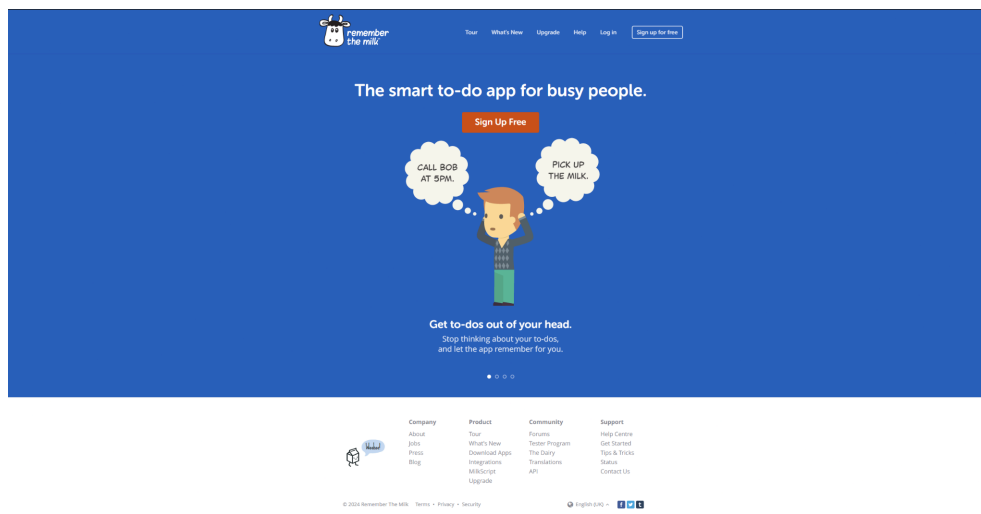


Figure 6. Remember The Milk app homepage [37]

3. Author's solution proposal to the presented problem

This chapter is dedicated to list generic functionalities of an application that may fulfill the known strategies for dealing with ADHD problems that were described in chapters 1.4 - 1.6. Moreover, some of those generic functionalities were realized within the presented application and will be shortly introduced in this chapter and described in detail in later chapters.

3.1. How to approach common ADHD problems?

3.1.1. Generic functionalities that fulfill strategies for managing procrastination

There is not a single verbatim definition of procrastination. It can be described as:

- “[...] voluntarily delaying an intended course of action despite expecting to be worse off because of the delay and is common, especially among younger people” [38]
- “[...] a form of self-regulatory failure linked to personality traits such as impulsiveness, distractibility, and low conscientiousness.” [38]
- “Procrastination involves unnecessary and unwanted delay, be it decisional, implemental, or lack of timeliness[...] core characteristic of procrastination is the realization by the actor that one will be worse off because of the delay. Hence, procrastination can be seen as irrational behavior—delaying some intended course of action, realizing that it is disadvantageous” [39]

So while there is no single verbatim definition listed, the key elements that emerge across the medical publications are:

1. procrastination involves voluntarily/intentionally delaying or postponing tasks or intended actions,
2. despite being aware that delaying will likely lead to negative consequences, and
3. it represents a failure of self-regulation.

With that in mind, there can be realized generic functionalities that meet the needs described in subchapter 1.4 *Procrastination*. To unambiguously recognize their purpose, the author will link every functionality to the strategy identifier it's supposed to fulfill:

1. Creating step-by-step to-do-list (1, 2, 3, 6, 8, 9, 10, 11, 12)
2. Saving dates into a calendar (2, 4, 5)
3. Making notes (3, 7, 8)
4. Notify about completed tasks (4, 12)
5. Support buddy (3, 4, 9, 11, 12, 13, 14, 15)

3.1.2. Generic functionalities that fulfill strategies for managing memory

Memory management encompasses a broad spectrum of techniques some of which were described in chapter *1.5 Memory management*. The following list presents generic functionalities that fulfill those presented strategies:

1. Creating step-by-step, visual to-do-list (5, 6)
2. Saving dates into a calendar (1, 3, 4)
3. Making notes (2, 4)
4. Notify about active or completed tasks (4, 6)
5. Support buddy (4)

3.1.3. Generic functionalities that fulfill strategies for managing attention

Attention management is another complex topic especially in the context of ADHD. Some techniques of managing attention were described in chapter *1.6 Attention management*. The following list presents generic functionalities that fulfill those presented strategies:

1. Creating step-by-step, easy-to-follow to-do-list (1, 2, 4, 6)
2. Making notes for remembering simple tasks (1, 2, 4)
3. Adding tasks into a calendar (1, 2, 3, 5)
4. Notify about active or completed tasks (3, 4, 6)
5. Support buddy (6)

3.2. Solution proposal to the mentioned problems

This subchapter serves as a brief introduction of the realized functionalities and the reasoning behind them. Detailed modeling, analysis and implementation will be described in later chapters.

3.2.1. Timer panel

Within the *Attention* category of the app there is a *Timer* panel which serves the obvious function of measuring time. The *Timer* panel is designed to fulfill the following strategies:

1. Set visual reminders (*Procrastination, Memory, Attention*) - notifies about the completed countdown with visual, affirming notification
2. Organize work surrounding (*Procrastination, Attention*) - *Timer* panel is designed to be minimal and reduce distractions by providing only the necessary information but at the same time be captivating and enjoyable to use
3. Minimize distractions (*Procrastination, Memory, Attention*) - *Timer* panel includes the necessary elements to perform countdown and nothing more
4. Reward perself for completing a task (*Procrastination, Attention*) - provides gratification mechanism for enduring the set time for finishing the task
5. Set time blocks (*Attention*) - *Timer* by its very nature sets a time block for completing a task
6. Practice mindfulness (*Procrastination*) - the intrinsic value of time countdown is mindful observation of time passage. *Timer* provides this by definition of its functionality

3.2.2. Notes panel

Within the *Attention* category of the app there is a *Notes* panel which serves the function of saving quick notes. The *Notes* panel is designed to fulfill the following strategies:

1. Prepare an organizational system (*Procrastination*) - *Notes* menu panel is currently designed to have a limited amount of slots to be used. This helps with overflowing the user with unnecessary, unremoved content and forces to delete the old content (may motivate to complete tasks)
2. Set visual reminders (*Procrastination, Memory, Attention*) - notifies about the successful operation of saving a note
3. Organize work surrounding (*Procrastination, Attention*) - *Notes* menu panel is designed to be minimal and reduce distractions by providing only the necessary information but at the same time be captivating and very easy to use
4. Minimize distractions (*Procrastination, Memory, Attention*) - *Notes* menu panel includes the necessary elements to save notes, display them and nothing more.
5. Keep a notebook (*Memory, Attention*) - the primary function of the *Notes* panel is to have instant access to *notes* and easily save them. Current version of the application

does not provide any special formatting and so the decision fatigue is reduced (future versions may provide special formatting while minimizing the mentioned decision fatigue)

6. Keep an organized list of tasks for the day (*Memory, Attention*) - *Notes* panel is designed to be an instant-access panel that allows for saving short-term reminders on activities. It may not provide long-term organization elements yet.
7. Practice mindfulness (*Procrastination*) - the intrinsic value of saving *notes* is focusing on the moment to appropriately store important information in the app.

3.2.3. Knowledge base panel & Challenges panel

Within the *Attention module* there are also *Knowledge base* and *Challenges* panels that serve a purpose of an accessible way for the user to get familiar with the prepared materials. The materials include *Educational* information on ADHD and *Challenges* panel that aims to propose a solution for the common daily struggles a person might face. Both the *Challenges* panel and the *Knowledge base* panel are designed to fulfill similar strategies and so the author decided to list the strategies that are met by the more general flashcard functionality, in context of the presented application of course. Both panels are functionally very similar and only really differ in their content and so their use case.

The *Flashcards* panel is designed to fulfill the following strategies:

1. Prepare an organizational system (*Procrastination, Attention*) - Both panels are currently designed to have pre-made flashcards, so that the information included is verified and reliable to study and face (in context of challenges).
2. Set visual reminders (*Procrastination, Memory, Attention*) - the action of drawing a random flashcard and flipping the flashcard to reveal the answer is very clear and visual so that it is unambiguous which flash card is active. Moreover, the frontside has white background color while the backside has green background color so it is clearly indicated which side is currently displayed.
3. Organize work surrounding (*Procrastination, Attention*) - Each panel is designed to be minimal and reduce distractions by providing only the necessary information but at the same time be captivating and very easy to use.
4. Minimize distractions (*Procrastination, Memory, Attention*) - Each panel includes only the necessary elements. There is only a “Random” button which draws a random flashcard from the stack and a flashcard itself that can be flipped to reveal the hidden side.
5. Keep a notebook (*Memory, Attention*) - the primary function of the panel is to have easy access to the information included and refer to it at any time. Current version of the application does not provide any customization options (adding custom flashcards or editing existing ones).
6. Gamification (*Procrastination, Memory, Attention*) - These panels are designed to be easy-access panels at any time. The intrinsic element of quizzing that is a part of flashcards meets the gamification requirements - the user either “wins or loses” by answering a question from the *Knowledge base panel* and is literally given a

challenge to face when choosing the *Challenging flashcards*. The user interface and functionality is tailored to be approachable and maximally easy to use.

3.2.4. Routines panel

Within the *Organization* category of the app there is a *Routines* panel which serves the function of creating a 4-step activities to follow which shall lead to fostering a new habit. The *Routines* panel is designed to fulfill the following strategies:

1. Break projects into actionable microtasks (*Procrastination, Memory, Attention*) - the intrinsic value of a *routine* is to break a complex task into atomic activities that are easy to follow, complete and remember.
2. Prepare an organizational system (*Procrastination*) - *Routines Menu* panel is currently designed to have a limited amount of slots to be used. This helps with overflowing the user with unnecessary, unremoved content and forces to delete the old content (may motivate to complete tasks)
3. Overcome perfectionism (*Procrastination*) - *Routine* is designed to be simple. It does not allow for elaborate description of the task, rather it encourages simple sentence commands or keywords, just enough to know what is there to do without the pressure to structure the text
4. Set visual reminders (*Procrastination, Memory, Attention*) - User gets notified about the successful operation of saving a *routine* when user does so and about a routine completion after marking all the steps.
5. Act according to set routines (*Procrastination*) - Primary purpose of a *routine* is to set positive reinforcement for the given habit.
6. Reward yourself for completing a task (*Procrastination*) - After marking all the steps of the routine, the user gets notified about a successful completion of the routine.
7. Organize work surrounding (*Procrastination, Attention*) - *Routines Menu* panel is designed to be minimal and reduce distractions by providing only the necessary information but at the same time be captivating and very easy to use
8. Minimize distractions (*Procrastination, Memory, Attention*) - *Routines Menu* panel includes the necessary elements to save *routines*, display them and track the completion of the habit with reinforcement gratification.
9. Keep a notebook (*Memory, Attention*) - *Routine* is designed to be a 4-step activity that ends with a gratifying notification of completion.
10. Keep an organized list of tasks for the day (*Memory, Attention*) - *Routines Menu* is designed to be an easy-access panel that allows for saving long-term stacks of activities. It may help in structuring new habits eg. morning hygiene.
11. Gamification (*Procrastination, Attention*) - *Routines* are designed to gratify routine completion. They serve a purpose of a basic to-do-list that are easy to follow. They are meant to encourage to complete an entire sequence of tasks and gratify on completion.

3.2.5. Forums panel

Within the *People* category of the app there is a *Forums* panel which serves the function of keeping the list of trustworthy and reliable communities dedicated to ADHD and mental health. The *Forums* panel is designed to fulfill the following strategies:

1. Prepare an organizational system (*Procrastination*) - *Forums Page* was purposefully given a separate panel - to reduce the clutter of a multi-purpose view, making it dedicated to communities (separation from helpline). *Forums Page* panel currently holds only a couple of communities, but is about to expand the list in the foreseeable future.
2. Organize work surrounding (*Procrastination, Attention*) - *Forums Page* panel is designed to be minimal and reduce distractions by providing only the necessary information but at the same time be captivating and very easy to use
3. Minimize distractions (*Procrastination, Memory, Attention*) - *Forums Page* panel includes the necessary elements to visit and discover the community.
4. Rest and keep the head sharp (*Procrastination*) - For when there is a crisis, it is helpful to have a community or professionals to reach out to.

3.2.6. Helpline panel

Within the *People* category of the app there is a *Helpline* panel which serves the function of keeping the list of trustworthy and reliable helplines dedicated to ADHD and mental health. The *Helpline* panel is designed to fulfill the following strategies:

1. Prepare an organizational system (*Procrastination*) - *Helpline Page* was purposefully given a separate panel - to reduce the clutter of a multi-purpose view, making it dedicated to helpline (separation from communities). The *Helpline Page* panel currently holds only a couple of contacts, but is about to expand the list in the foreseeable future.
2. Organize work surrounding (*Procrastination, Attention*) - *Helpline Page* panel is designed to be minimal and reduce distractions by providing only the necessary information but at the same time be captivating and very easy to use.
3. Minimize distractions (*Procrastination, Memory, Attention*) - *Helpline Page* panel includes the necessary elements to contact the helpline.
4. Rest and keep the head sharp (*Procrastination*) - For when there is a crisis, it is helpful to have professionals to reach out to in a secure and accessible way.

3.2.7. Affirmations panel

Within the *Mindfulness module* there is an *Affirmations* panel that serves a purpose of an accessible way for the user to get positive affirmation, primarily in the context of ADHD. The materials include positive reinforcement on ADHD for the common daily struggles a person might face.

The *Affirmations* panel is designed to fulfill the following strategies:

1. Prepare an organizational system (*Procrastination, Attention*) - *Affirmations* are currently designed to have pre-made flashcards, so that the information included is reliable to internalize.
2. Set visual reminders (*Procrastination, Memory, Attention*) - the action of drawing a random flashcard is very clear and visual so that it is unambiguous which flash card is active. *Affirmations* are designed to only have a frontside due to their intrinsic simple presentation. For the distinction from the other flashcards panels, the *affirmations* were given a different background color - pink that is believed to be fitting the context of *affirmations*.
3. Organize work surrounding (*Procrastination, Attention*) - *Affirmations* panel is designed to be minimal and reduce distractions by providing only the necessary information but at the same time be captivating and very easy to use.
4. Minimize distractions (*Procrastination, Memory, Attention*) - *Affirmations* panel includes only the necessary elements. There is only a “Random” button which draws a random flashcard from the stack and a flashcard itself that can be viewed
5. Keep a notebook (*Memory, Attention*) - the primary function of the *Affirmations* panel is to have easy access to the information included and refer to it at any time. Current version of the application does not provide any customization options (adding custom flashcards or editing existing ones).
6. Gamification (*Procrastination, Memory, Attention*) - *Affirmations* panel is designed to be an easy-access panel at any time. The intrinsic element of drawing a random *affirmation* meets the gamification requirement of an element of surprise.

3.2.8. Graph tasks module

There is a dedicated category within the application that originated from the idea of cooperating with procrastination. This category is called *Graph tasks* and serves a primary purpose of creating step-by-step to-do-lists with atomic activities to follow which also incorporates mindful procrastination.

Routines fulfill similar strategies as *Graph tasks* do (hence the list is similar) but they should be used for different purposes. While *routine* shall help in structuring long-term habit, *graph task* helps in breaking down complex tasks into atomic activities to be acted upon.

The *Graph task* panel is designed to fulfill the following strategies:

1. Break projects into actionable microtasks (*Procrastination, Memory, Attention*) - the intrinsic value of *graph task* is to break a complex task into atomic activities that are easy to follow, complete and remember.

2. Prepare an organizational system (*Procrastination*) - *Graph Task Menu* panel is currently designed to have a limited amount of slots to be used. This helps with overflowing the user with unnecessary, unremoved content and forces to delete the old content (may motivate to complete tasks)
3. Overcome perfectionism (*Procrastination*) - *graph task* is designed to be simple. It does not allow for elaborate description of the task, rather it encourages simple sentence commands or keywords, just enough to know what is there to do without the pressure to structure the text
4. Set visual reminders (*Procrastination, Memory, Attention*) - notifies about the successful operation of saving a *graph task*
5. Organize work surrounding (*Procrastination, Attention*) - *Graph Task Menu* panel is designed to be minimal and reduce distractions by providing only the necessary information but at the same time be captivating and very easy to use
6. Minimize distractions (*Procrastination, Memory, Attention*) - *Graph Task Menu* panel includes the necessary elements to save *graph tasks*, display them and nothing more
7. Keep a notebook (*Memory, Attention*) - *Graph task* serves as a multistep *note*. It's designed to be a 4-step activity that ends with the "You did it!" kind of gratification message
8. Keep an organized list of tasks for the day (*Memory, Attention*) - *Graph Task Menu* is designed to be an easy-access panel that allows for saving short-term stacks of activities. It may not fully meet the need for long-term planer yet but it can very well be used to structure new habits eg. reading a book before going to bed.
9. Practice mindfulness (*Procrastination*) - the intrinsic value of making a *graph task* is focusing on the moment to appropriately store important information in the app.
10. Visualizing the end result (*Procrastination, Memory*) - the built-in element of *graph task* is "the final activity" which shall mark the completion of the task.
11. Gamification (*Procrastination, Attention*) - *graph tasks* are designed to be graphs of activities. They serve a purpose of a basic to-do-list but are visually more than that. They were meant to resemble "badge finding" games which point to the next treasure. The literal incorporation of "arrows" to follow a direction meets that purpose.

3.2.9. Assessment test panel

The *assessment test* panel serves a radically different purpose. It is based on the medically approved test of the Adult ADHD Self-Report Scale (ASRS) v1.1 [40] and is meant to be an educational panel only. The *assessment test* panel included in the application is by no means an exhaustive nor official diagnostic tool. The *assessment test* within the app serves as the first steps to familiarize the user with ADHD and possibly spark an interest in this topic.

Test consists of 18 questions, somewhat divided by the common characteristics of either hyperactive/impulsive ADHD or the inattentive ADHD:

→ Hyperactivity/Impulsivity:

- ◆ How often do you fidget or squirm with your hands or feet when you have to sit down for a long time?
- ◆ How often do you feel overly active and compelled to do things, like you were driven by a motor?
- ◆ How often do you have problems remembering appointments or obligations?
- ◆ When you have a task that requires a lot of thought, how often do you avoid or delay getting started?
- ◆ How often do you leave your seat in meetings or other situations in which you are expected to remain seated?
- ◆ How often do you feel restless or fidgety?
- ◆ How often do you have difficulty unwinding and relaxing when you have time to yourself?
- ◆ How often do you find yourself talking too much when you are in social situations?
- ◆ When you're in a conversation, how often do you find yourself finishing the sentences of the people you are talking to, before they can finish them themselves?
- ◆ How often do you have difficulty waiting your turn in situations when turn taking is required?
- ◆ How often do you interrupt others when they are busy?

→ Inattention:

- ◆ How often do you have trouble wrapping up the final details of a project, once the challenging parts have been done?
- ◆ How often do you have difficulty getting things in order when you have to do a task that requires organization?
- ◆ How often do you have problems remembering appointments or obligations?
- ◆ When you have a task that requires a lot of thought, how often do you avoid or delay getting started?
- ◆ How often do you make careless mistakes when you have to work on a boring or difficult project?
- ◆ How often do you have difficulty keeping your attention when you are doing boring or repetitive work?

- ◆ How often do you have difficulty concentrating on what people say to you, even when they are speaking to you directly?
- ◆ How often do you misplace or have difficulty finding things at home or at work?
- ◆ How often are you distracted by activity or noise around you?

Explanation of the scale, scoring system and algorithm for calculating the result will be provided and described in chapter 7. *Implementation of the project*. The next chapter goes in detail about the analysis and modeling process of the described functionalities, their practical incorporation into functional mobile application and their premeditated use cases.

3.3. Specialist approved solution

The author had the pleasure to consult the work with psychologist, psychotherapist and psychoeducator Dorota Dot Okulicz [41]. The author presented the application and the idea behind it. Discussion was held concerning the attractiveness of user interface, accessibility of functionalities and a substantive value behind them including the research that was done and included in this paper. Dorota gave insights on how to improve the application and gave her honest opinion which reads as follows:

“The mobile application authored by Antoni Malinowski dedicated to people with ADHD is an interesting solution, which aggregates many potentially useful tools in one place, such as notes, timer, flashcards (with knowledge about this neurodevelopmental disorder), challenges, experiments, affirmations. User experience of the application is intuitive to me. The application does not contain any distracting elements, it is very easy to use and has a simple graphic design. I think it has a chance to support in working on procrastination and memory problems. It certainly still requires further development and functional enhancement, but in its current phase the idea seems interesting and worth developing.”

4. Modeling and analysis of the proposed solution

This chapter describes in greater detail the process of analysis of the presented solution as well as the modeling process of aligning the architectural diagrams to match the requirements so that it can later be practically implemented adhering to software engineering design patterns.

4.1. Requirements

The analysis begins by delineating the requirements into two categories: functional and non-functional. Functional requirements detail what the application must do, including tasks such as user authentication, data processing, and real-time notifications. Non-functional requirements focus on how the system performs these tasks, emphasizing attributes like scalability, security, and user experience.

4.1.1. Functional

Functional requirements describe what the application is expected to do. They define the core functionalities and features that the application must support. The presented tables contain functional requirements of the application for the following modules:

1. ADHD screening test module [42]
2. Attention module
 - 2.1. Timer
 - 2.2. Notes
 - 2.3. Knowledge base
 - 2.4. Challenges
3. Organization module
 - 3.1. Routine
4. People module
 - 4.1. Forums
 - 4.2. Helpline
5. Mindfulness module
 - 5.1. Affirmations
6. Graph Tasks module

Table 2 presents the functional requirements table for the ADHD screening test module.

No.	Requirement name	Description	Constraint
1	Take assessment test	User can take the ADHD assessment test as a part of the educational module of the application which is based on the Adult ADHD Self-Report Scale (ASRS) v1.1 [40]. The provided screening test is for user information only and it does not indicate a diagnosis of ADHD.	

Table 2. Functional requirements table for the ADHD screening test module (source: self)

Table 3 presents the functional requirements table for the timer panel included within the attention module.

No.	Requirement name	Description	Constraint
2.1.1	Set the timer	User can set timer for a arbitrary period of time ranging from 1 minute to 23 hours and 59 minutes in the increments of 1 minute	
2.1.2	Start the timer	User can start the timer by pressing the green “Start” button which triggers the countdown. When the countdown finishes a popup appears informing the user about the completed countdown and the device vibrates for 2 seconds.	Time must be first set, otherwise the button is disabled
2.1.3	Stop the timer	User can stop the timer by pressing the red “Stop” button which stops the countdown. When the button is triggered, the countdown stops and may be resumed by clicking on the green “Start” button.	Countdown must be first stopped, otherwise the button is disabled
2.1.4	Reset the timer	User can reset the timer by pressing the gray “Reset” button which resets the time. When the button is triggered, the time returns to the initially set by the user.	Countdown must be first stopped, otherwise the button is disabled

Table 3. Functional requirements table for the timer panel included within the attention module (source: self)

Table 4 presents the functional requirements table for the notes panel included within the attention module.

No.	Requirement name	Description	Constraint
2.2.1	Display Notes menu	User can display the Notes menu with five available slots for adding custom notes.	
2.2.2	Add new note	User navigates to EditNotePage from the notes menu view by clicking on the “Add new note” label if there is any empty note left or directly on the edit icon. By filling in the wanted fields and pressing the green “Save” button the user saves the note , gets notified about successful operation and is redirected back to the note menu.	There must be an empty slot left for adding a new note.
2.2.3	Edit note	User navigates to the Edit Note page from the notes menu view by clicking on the orange pencil icon. By filling in the wanted fields and pressing the green “Save” button the user updates the existing note, gets notified about successful operation and is redirected back to the notes menu.	There must exist some custom note. Otherwise, the operation that is being performed is actually “Add new note” (technically very similar)
2.2.4	Delete note	User can delete the custom note by clicking on the red trash icon.	There must exist a custom note.
2.2.5	Display note	User can display the custom note by clicking on any available note label (from the notes menu) as long as it has been created by the user.	There must exist a custom note.

Table 4. Functional requirements table for the notes panel included within the attention module (source: self)

Table 5 presents the functional requirements table for the Education flashcards panel included within the attention module.

No.	Requirement name	Description	Constraint
2.3.1	Display Education flashcards panel	User can display the Educational flashcards panel with premade flashcards presented one at a time	
2.3.2	Flip the flashcard	User can display the other side of the flashcard by tapping on the flashcard. It results in flipping the flashcard.	
2.3.3	Draw random flashcard	User can draw random flashcard from the premade stack of flashcards by clicking the “RANDOM” button.	

Table 5. Functional requirements table for the education flashcards included within the attention module
(source: self)

Table 6 presents the functional requirements table for the Challenging flashcards panel included within the attention module.

No.	Requirement name	Description	Constraint
2.4.1	Display Challenging flashcards panel	User can display the Challenging flashcards panel with premade flashcards presented one at a time	
2.4.2	Flip the flashcard	User can display the other side of the flashcard by tapping on the flashcard. It results in flipping the flashcard.	
2.4.3	Draw random flashcard	User can draw random flashcard from the premade stack of flashcards by clicking the “RANDOM” button.	

Table 6. Functional requirements table for the challenging flashcards included within the attention module
(source: self)

Table 7 presents the functional requirements table for the routine panel included within the organization module.

No.	Requirement name	Description	Constraint
3.1.1	Display Routine menu	User can display the Routine menu page with five available slots for adding custom routines.	
3.1.2	Add new routine	User navigates to the Edit Routine page from the Routine menu view by clicking directly on the orange pencil icon. By filling in the wanted fields and pressing the green “Save” button the user saves the routine, gets notified about successful operation and is redirected back to the Routine menu.	There must be an empty slot left for adding a new routine.
3.1.3	Edit routine	User navigates to the Edit Routine page from the routine menu view by clicking on the orange pencil icon. By filling in the wanted fields and pressing the green “Save” button the user updates the existing routine, gets notified about successful operation and is redirected back to the routine menu.	There must exist some custom routine. Otherwise, the operation that is being performed is actually “Add new routine” (technically very similar)
3.1.4	Delete routine	User can delete the custom routine by clicking on the red trash icon.	There must exist a custom routine.
3.1.5	Display routine	User can display the custom routine by clicking on any available routine label (from the notes menu) as long as it has been created by the user.	There must exist a custom routine.

Table 7. Functional requirements table for the routine panel included within the organization module (source: self)

Table 8 presents the functional requirements table for the forums panel included within the people module.

No.	Requirement name	Description	Constraint
4.1.1	Display Forums page	User can display the Forums page listing some of the reliable forums	
4.1.2	Visit the forum	User can navigate to the chosen forum's website by clicking the blue internet icon. This action navigates to the chosen destination outside of the application.	User must have and internet connection to access the forums

Table 8. Functional requirements table for the forums panel included within the people module (source: self)

presents the functional requirements table for the helpline panel included within the people module.

No.	Requirement name	Description	Constraint
4.2.1	Display Helpline page	User can display the Helpline page listing some of the trusted helplines.	
4.2.2	Call the helpline	User can call the helpline by clicking the green phone icon. This action navigates to the default phone app outside of the application.	

Table 9. Functional requirements table for the helpline panel included within the people module (source: self)

Table 10 presents the functional requirements table for the affirmations panel included within the mindfulness module.

No.	Requirement name	Description	Constraint
5.1.1	Display affirmations panel	User can display the affirmations panel with premade affirmation flashcards presented one at a time	
2.1.2	Draw random flashcard	User can draw random flashcard from the premade stack of flashcards by clicking the "RANDOM" button.	

Table 10. Functional requirements table for the affirmations panel included within the mindfulness module (source: self)

Table 11 presents the functional requirements table for the graph tasks panel of the graph tasks module.

No.	Requirement name	Description	Constraint
6.1	Display graph tasks menu/sample graph tasks menu	User can display the graph tasks menu with five available slots for adding custom graph tasks.	
6.2	Add new graph task	User navigates to the Edit Graph Task page from the graph tasks menu view by clicking directly the orange pencil icon. By filling in the wanted fields and pressing the green “Save” button the user saves the graph task, gets notified about successful operation and is redirected back to the graph task menu.	There must be an empty slot left for adding a new graph task.
6.3	Edit graph task	User navigates to the Edit Graph Task page from the graph tasks menu view by clicking directly the orange pencil icon. By filling in the wanted fields and pressing the green “Save” button the user updates the existing graph task, gets notified about successful operation and is redirected back to the graph task menu.	There must exist some custom graph tasks. Otherwise, the operation that is being performed is actually “Add new graph task” (technically very similar)
6.4	Display graph task	User can display the custom graph task/sample graph task by clicking on any available graph task label (from the graph tasks menu or the sample graph tasks menu) as long as it is either created by the user or it is a sample graph task.	There must exist custom graph task or sample graph task.

Table 11. Functional requirements table for the graph tasks panel within the graph tasks module (source: self)

4.1.2. Non-functional

Non-functional requirements (NFRs) define the standards, behaviors, and attributes that a system must exhibit to be considered effective, efficient, and satisfying to the user and other stakeholders. These requirements do not specify particular behaviors but instead describe how the system should behave and perform under various circumstances.

The author divided non-functional requirements into the following categories:

1. Accessibility
2. Security
3. Ethics
4. Performance
5. Maintainability

Table 12 presents the non-functional requirements table for accessibility within the application.

No.	Feature	Description	Priority
1.1	Color Contrast and Visual Design	Minimal white color palette ensures clean and welcoming user interface. Obvious color contrast makes it easy to navigate through all UI elements without being distracted and visually repulsed.	Must-have
1.2	Font size and style	The chosen font is sans-serif to ensure readability and universal appeal of the text. Font size should be big enough to fit relatively on all screen sizes yet small enough not to disturb the entirety of the layout.	Must-have
1.3	Uptime	At this stage of development the app is inherently off-line which makes it easier to ensure the uptime. All the updates to the app are expected to follow that rule for now, until a breaking change that introduces online features gets incorporated	Must-have
1.4	Visibility of system status	<i>“The design should always keep users informed about what is going on, through appropriate feedback within a reasonable amount of time.”</i> [43] For this very reason, every action involving modification of the app	Must-have

		content by the user eg. adding a new graph task is accompanied by the notification of whether the operation was successful or not.	
1.5	Cross-platform	Mobile app was written in .NET MAUI which inherently ensures multi-platform availability of the application - both for mobile and desktop. The app was initially designed to be an Android-first application, however there is a great potential in making a web version of it. Web version would allow for access across all devices without the need to download the app which aligns with the primary goal - accessibility and inclusivity.	Should-have
1.6	Language	The app is by default in English as at the initial stage of its release it is expected to target primarily English-speaking audience, however plans for further development expect the app to be multilingual on the basis of the country of the user which he choses in the app settings	Should-have

Table 12. Non-functional requirements table for accessibility within the application (source: self)

Table 13 presents the non-functional requirements table for security within the application.

No.	Feature	Description	Priority
2.1	Data-safety (Offline-first)	The current version of the app is offline-first which means that no access to the internet is required to use the app ⁸ . This goes in-hand with the personal use of the app - even if there is no personal data being shared the user should be guaranteed that he is the only one with the access to the app resources.	Must-have

Table 13. Non-functional requirements table for security within the application (source: self)

⁸ To visit both the assessment test source (from assessment test panel) and a given forum's website (from forums panel) the user has to have an internet connection. Accessing the panels themselves does not require internet connection.

Table 14 presents the non-functional requirements table for ethics concerning the application.

No.	Feature	Description	Priority
3.1	Free software/Libre software	Libre means “free” as in “free speech” not “free drink”. Because the author strongly believes in the free nature of development - that the user should have complete freedom on what to do with the software, the source code must be under GNU General Public License, version 3, which not only by its very nature guarantees that the code is public but also guarantees user’s freedom to share and change free software - to make sure the software is free for all its users. [44]	Must-have
3.2	Privacy	At the current stage of development no personal data is being collected by the author nor the software. As of the present version, there is no need to identify the user of the application - the functionality does not require any personal data to be collected.	Must-have
3.3	No personal data used for marketing purposes without users’ consent	From the stage of development when there will be an incentive to record personal data - eg. account creation, there will be actions taken to permit the use and manipulation of personal data for marketing purposes against the will of the users.	Must-have

Table 14. Non-functional requirements table for ethics concerning the application (source: self)

Table 15 presents the non-functional requirements table for performance within the application.

No.	Feature	Description	Priority
4.1	The “3-Click Rule”	<p><i>“The 3-click rule is a persistent, unofficial heuristic that says that no page should take more than 3 clicks (or taps on a touchscreen) to access.” [45].</i></p> <p>Although this rule has no official backing and actually taking care of practicality is more important than how fast can the user navigate to some screen, the author considers it a good-enough rule-of-thumb to keep the access to the most important sections of the app within those “3 clicks”.</p>	Must-have
4.2	Response time	<p>To ensure smooth and enjoyable usage of the app it is recommended to minimize the load time for various actions within the app eg. navigating between screens. <i>“There are 3 main time limits (which are determined by human perceptual abilities) to keep in mind when optimizing web and application performance.”</i> - 1 second can be considered as the appropriate threshold for the user’s follow of thought to stay uninterrupted and so that is the maximum response time for the author’s application to follow [46].</p>	Must-have

Table 15. Non-functional requirements table for performance within the application (source: self)

Table 16 presents the non-functional requirements table for maintainability within the application.

No.	Feature	Description	Priority
5.1	Cross-platform	For the same reason that .NET MAUI inherently ensures multi-platform availability of the application it eases the maintainability as there is a single-code base that is responsible for delivering solutions to multiple platforms. At this stage of development there is no need to native-first applications and so C# as a language is sufficient to operate on the codebase.	Must-have
5.2	Open repository ⁹ code	Source code is kept in the open repository on github. It is expected to allow anyone to introduce changes and collaborate on the project. The repository is being actively maintained to minimize bottlenecks of legacy code-base.	Must-have

Table 16. Non-functional requirements table for maintainability within the application (source: self)

⁹ The author uses the term “open code repository” to describe the literal practice of making the code repository open and publicly available for display and collaboration. The term “open-source code repository” is deliberately not used because “open-source” is associated with mainly practical advantages and does not campaign for principles of free code [47]. Since the author bases himself on the idea of freedom for the users of computing and not solely on the practicality of “open-source code”, the author will use “open code” to draw a distinction from the “open-source code”. In practice, both “open code” and “open-source code” result in making the code publicly available, but because the latter does not advocate a movement for freedom and justice, the term will not be used.

4.2. System actors

Next, we identify the system actors, pivotal entities interacting with the application, such as users, administrators, and external services. This identification aids in the systematic development of use-cases, which illustrate the interactions between actors and the system to achieve a functional goal. This results in a comprehensive use-case diagram that provides a high-level overview of all possible interactions. For this initial version of the application there exists a need for only two actors:

- user
- subsystem of time (responsible for handling time measurement and periodical events)

The same safety feature of “offline-first” that secures data in its place allows for minimizing the unnecessary clutter of too many system users. This goes with the foreseeable vision for the application that is expected to be continuously improving and have new features added alongside scaling of the system and its user base.

Figure 7 presents the system actors within the application.

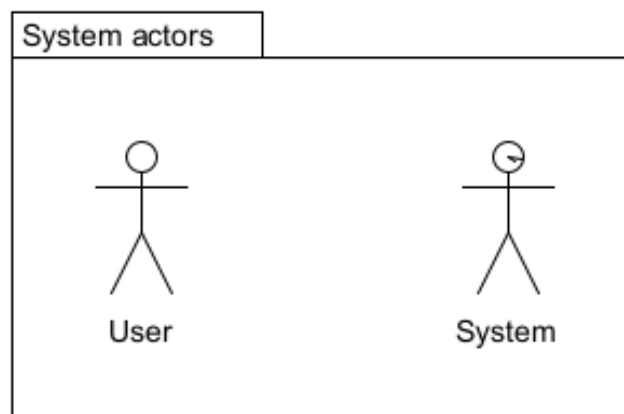


Figure 7. Diagram of system actors present the application (tool: UMLet [48], source: self)

4.3. Diagrams

The modeling section includes two types of critical diagrams. A use case diagram helps visualize the system's functional requirements by illustrating the interactions between actors and the system's use cases. This diagram provides a high-level overview of what the system does from the user's perspective.

An activity diagram is used to depict the workflow of the system, showing the sequence of activities and the decision points that guide the flow from one activity to another. It provides a detailed view of the processes within the system, making it easier to understand how various actions are coordinated and how decisions are made throughout the user's interaction with the system.

4.3.1. Use case diagrams

To describe all the capabilities of the system and what the user can do the author will present use case diagrams of all the actions that can be performed within each of the application modules. To avoid chaos, the use case diagrams presented are divided into 6 subchapters, one for each module of the application.

The author also provides a clear distinction between the functionalities provisioned by the technology used and functionalities that were implemented by the author. The distinction is important because while some of the functionalities (mainly rendering the user interface) are built into the technology used, some are mandatory to be implemented by the author or the application will lack the essential functionality (mainly adding, editing and deleting custom elements or responding to user gestures).

4.3.1.1. Screening test module

Figure 8 presents a use case diagram for the functionalities of the screening test module.

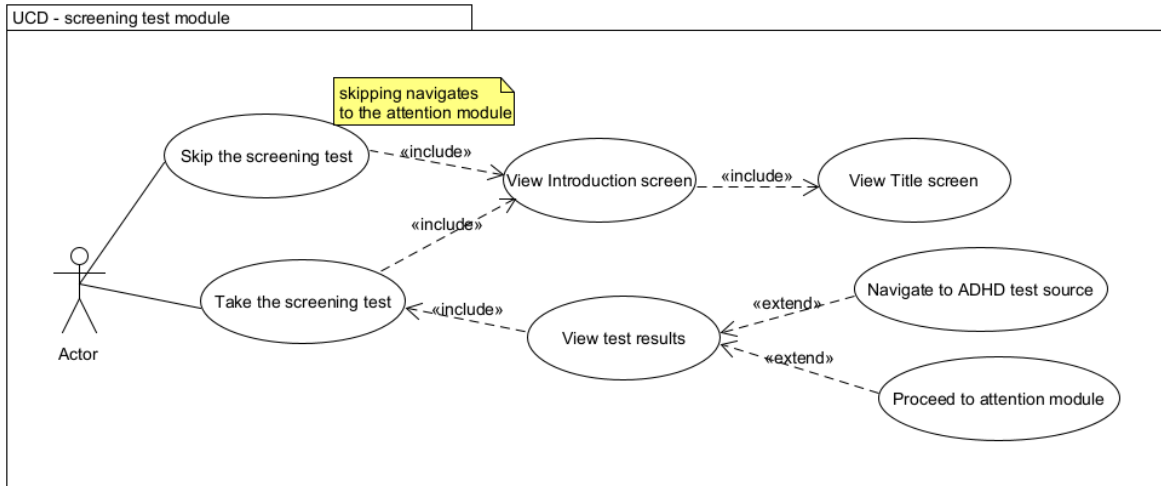


Figure 8. Use case diagram for the screening test module (tool: UMLet [48], source: self)

As stated by the author: “*Application that has no underlying implementation of crucial functionality has no practical use for users.*”[49]. Presented below figure 24 marks what has been implemented by the author to make the application work.

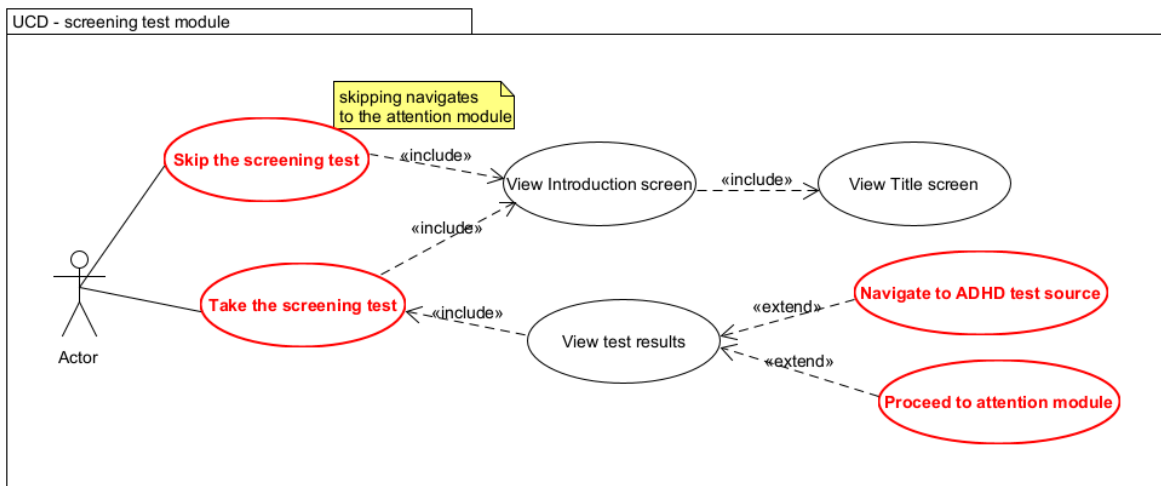


Figure 9. Use case diagram for the screening test module, marking what has been implemented by the author (tool: UMLet [48], source: self)

4.3.1.2. Attention module

Figure 10 presents a use case diagram for the functionalities of the panels within the attention module.

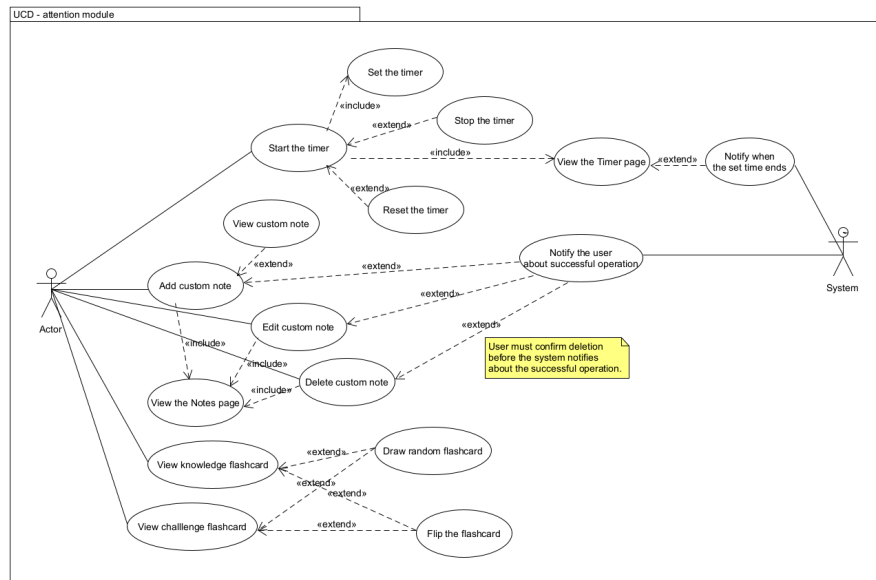


Figure 10. Use case diagram for the panels within the attention module (tool: UMLet [48], source: self)

Presented below figure 11 marks what has been implemented by the author to make the application work.

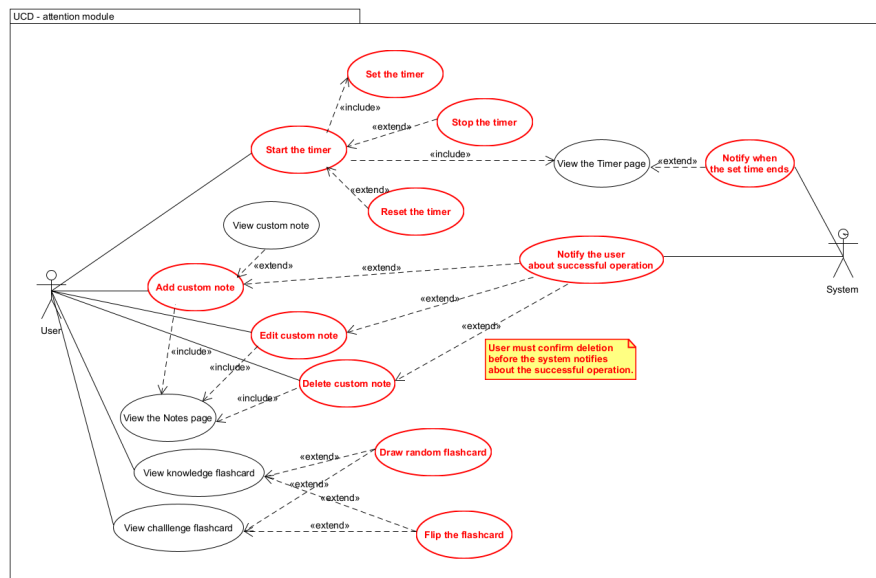


Figure 11. Use case diagram for the panels within the attention module, marking what has been implemented by the author (tool: UMLet [48], source: self)

Functionalities of displaying content on the user interface can be easily handled with XAML. However, all interaction that is performed within the application that involves modifying both its content, the database records and notifications that the system displays to the user must be implemented as it is later described.

4.3.1.3. Organization module

Figure 12 presents a use case diagram for the functionalities of the panels within the organization module.

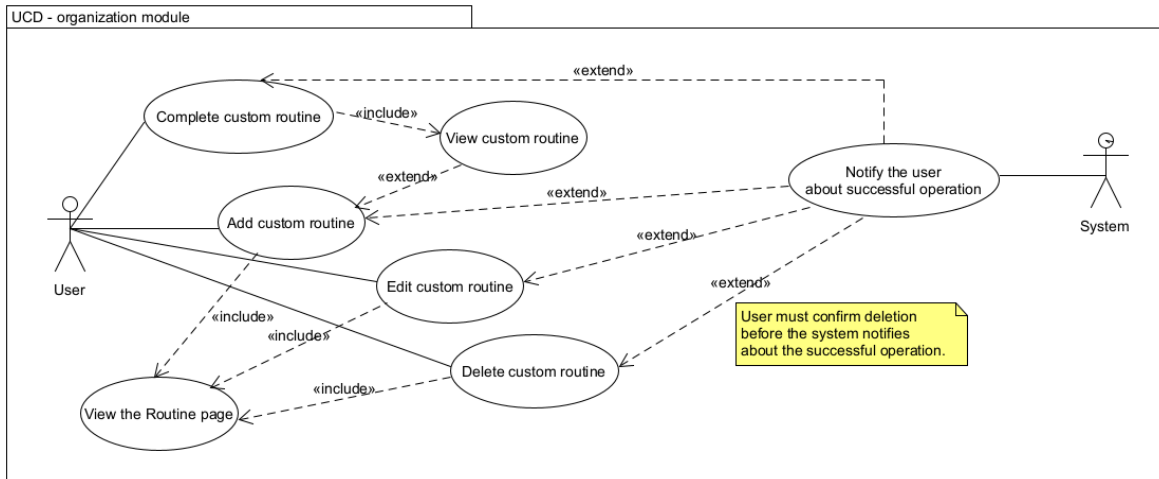


Figure 12. Use case diagram for the panels within the organization module (tool: UMLet [48], source: self)

Presented below figure 13 marks what has been implemented by the author to make the application work.

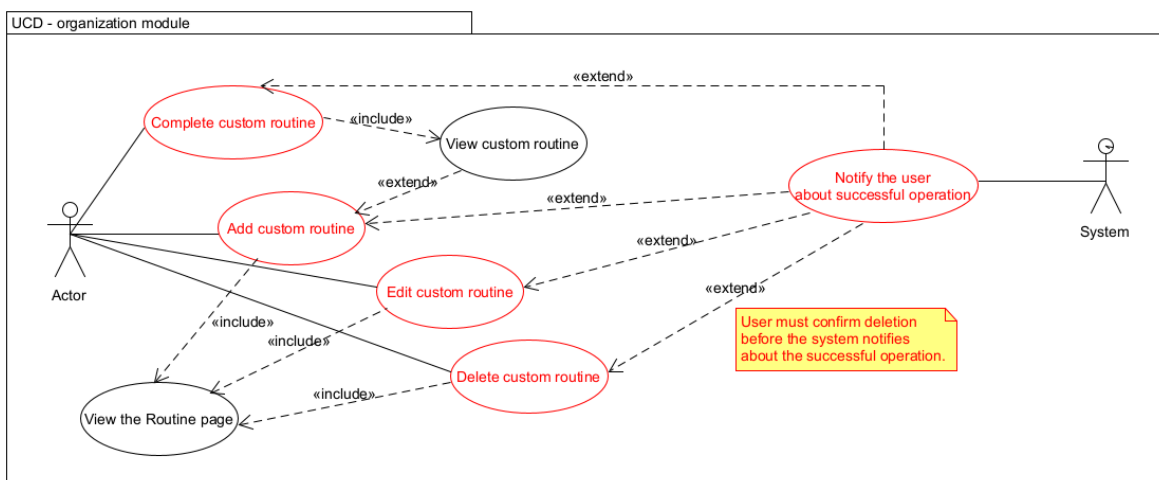


Figure 13. Use case diagram for the panels within the organization module, marking what has been implemented by the author (tool: UMLet [48], source: self)

4.3.1.4. People module

Figure 14 presents a use case diagram for the functionalities of the panels within the people module.

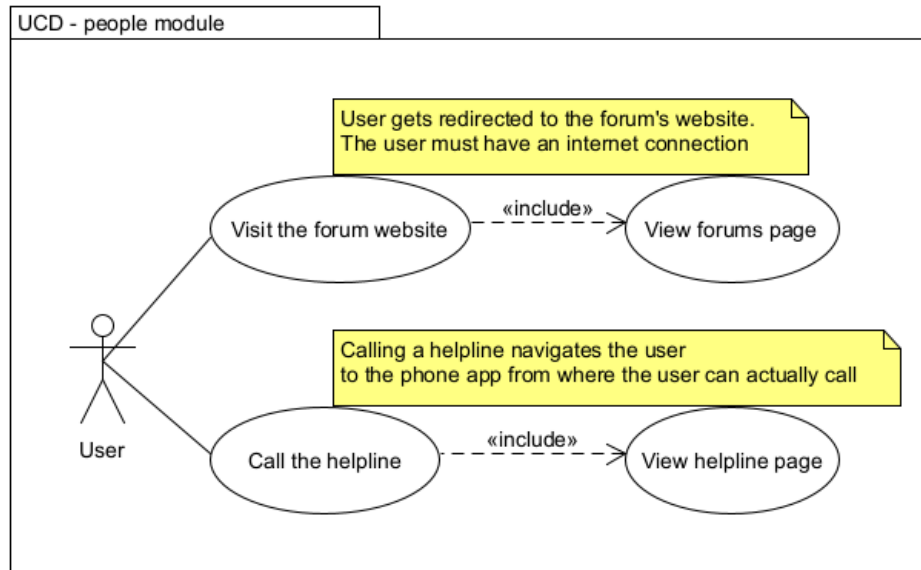


Figure 14. Use case diagram for the panels within the people module (tool: UMLet [48], source: self)

Presented below figure 15 marks what has been implemented by the author to make the application work.

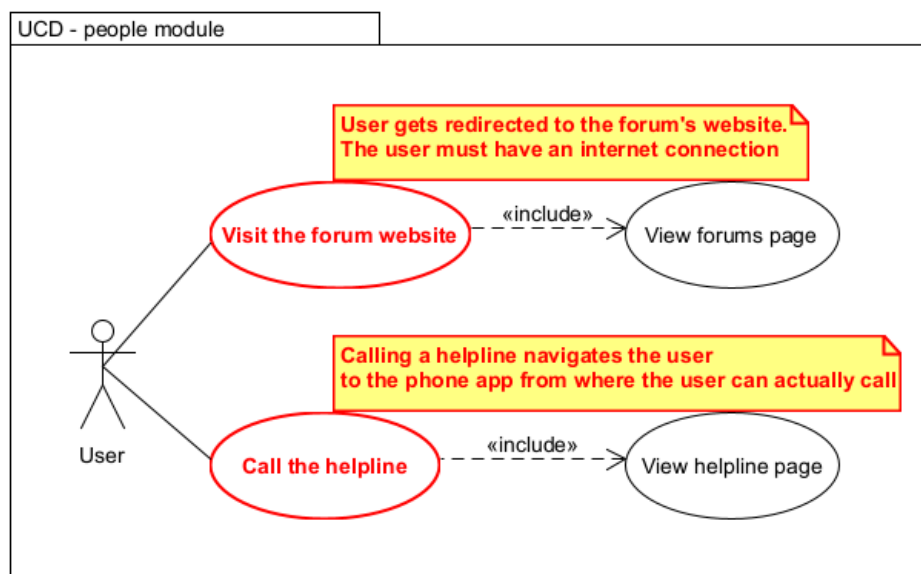


Figure 15. Use case diagram for the panels within the people module, marking what has been implemented by the author (tool: UMLet [48], source: self)

4.3.1.5. Mindfulness module

Figure 16 presents a use case diagram for the functionalities of the panels within the mindfulness module.

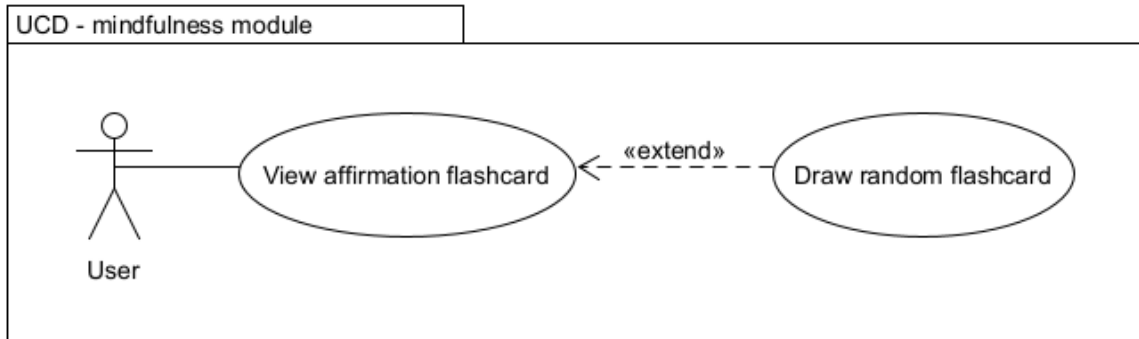


Figure 16. Use case diagram for the panels within the mindfulness module (tool: UMLet [48], source: self)

Presented below figure 17 marks what has been implemented by the author to make the application work.

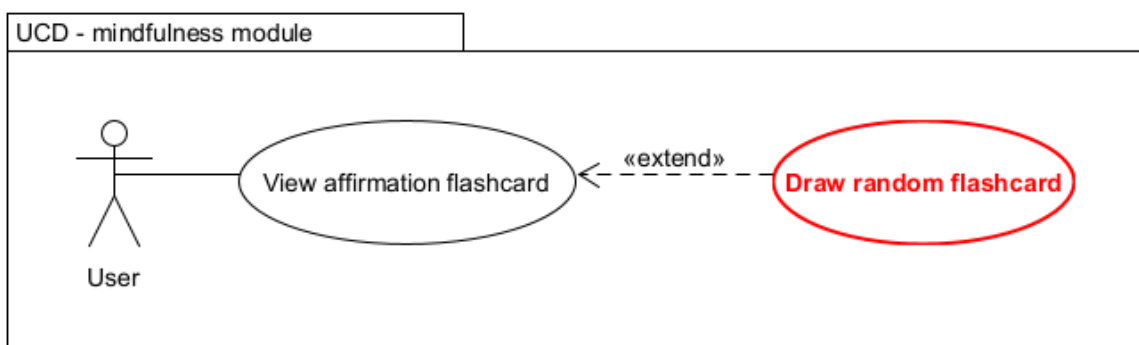


Figure 17. Use case diagram for the panels within the mindfulness module, marking what has been implemented by the author (tool: UMLet [48], source: self)

4.3.1.6. Graph tasks module

Figure 18 presents a use case diagram for the functionalities of the panels within the graph tasks module.

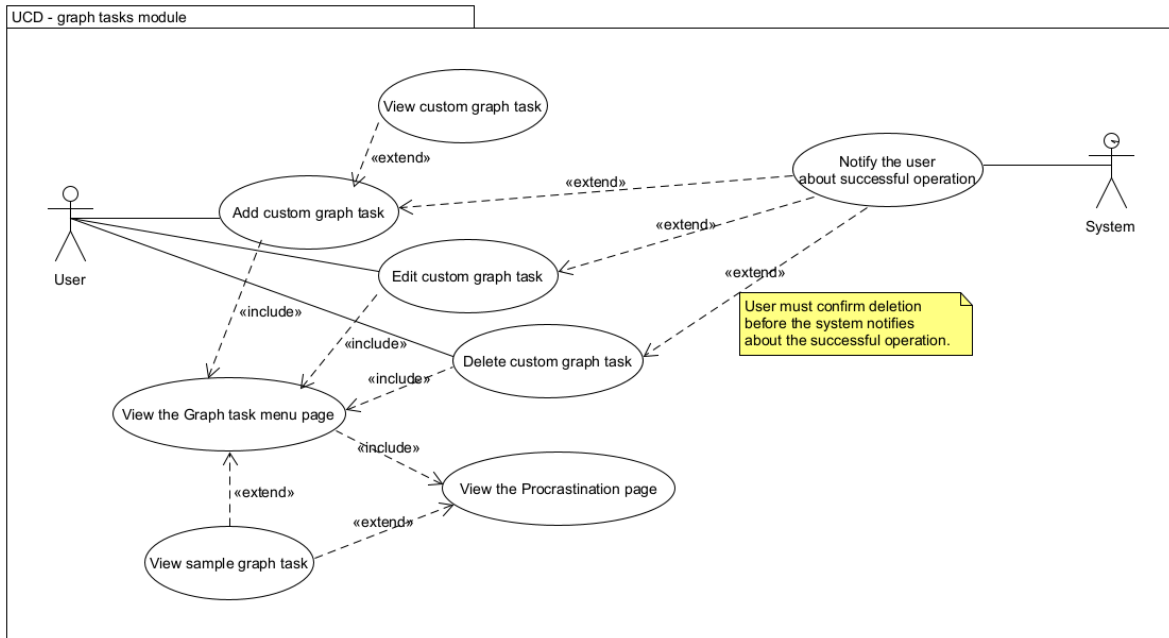


Figure 18. Use case diagram for the panels within the graph tasks module (tool: UMLet [48], source: self)

Presented below figure 19 marks what has been implemented by the author to make the application work.

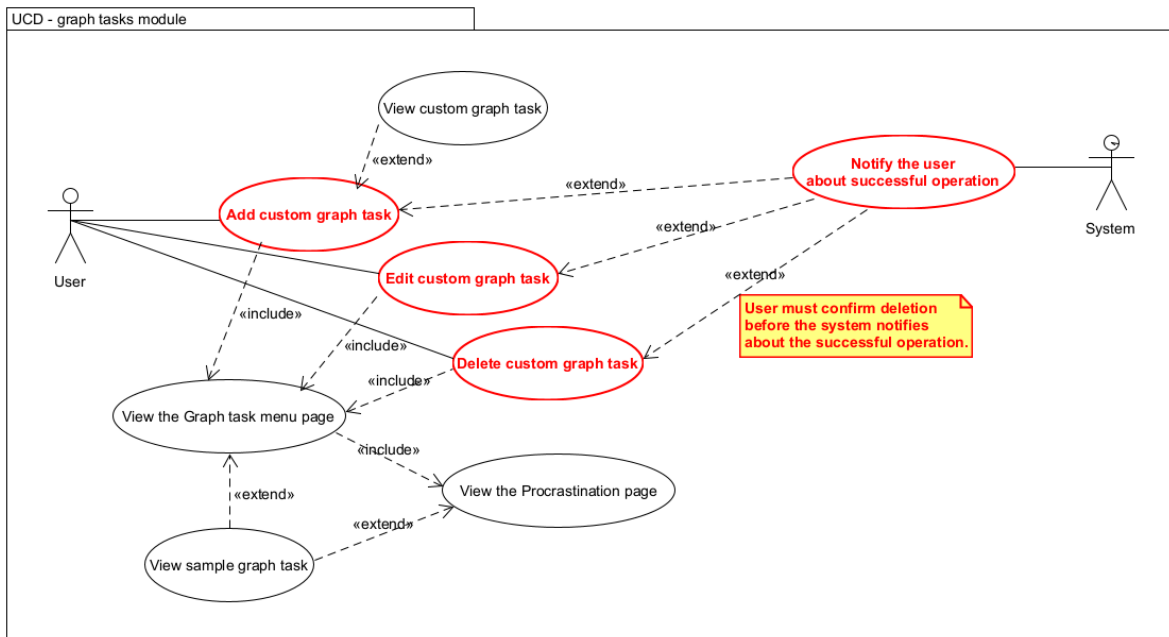


Figure 19. Use case diagram for the panels within the graph tasks module, marking what has been implemented by the author (tool: UMLet [48], source: self)

4.3.2. General activity diagram

Presented below figure 20 shows an activity diagram depicting the general flow of navigation starting from the user opening the application all the way through navigating to the attention module (acting as the home page).

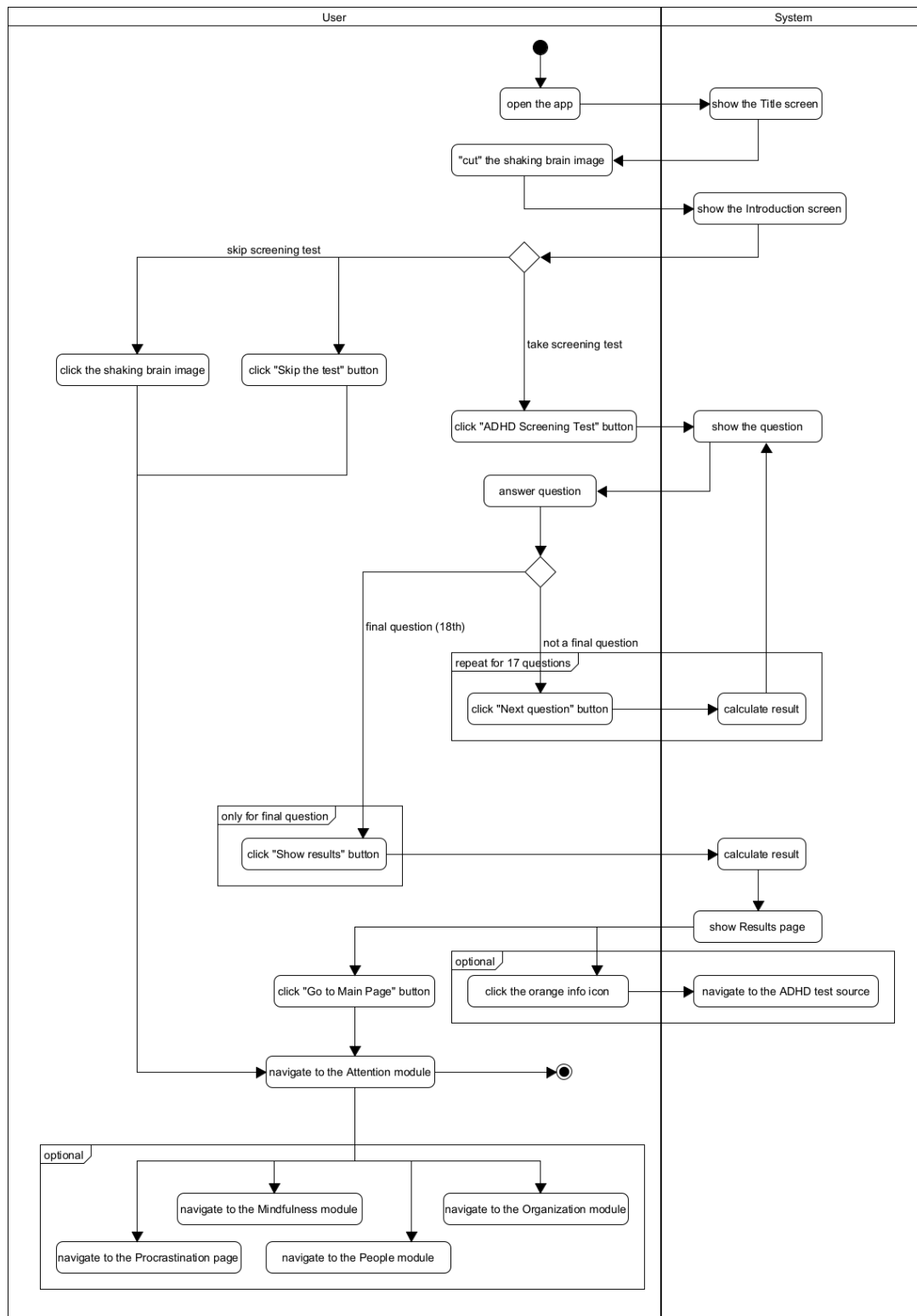


Figure 20. Use case diagram flow of the navigation (tool: UMLet [48], source: self)

Presented below figure 21 marks what has been implemented by the author to make the application work.

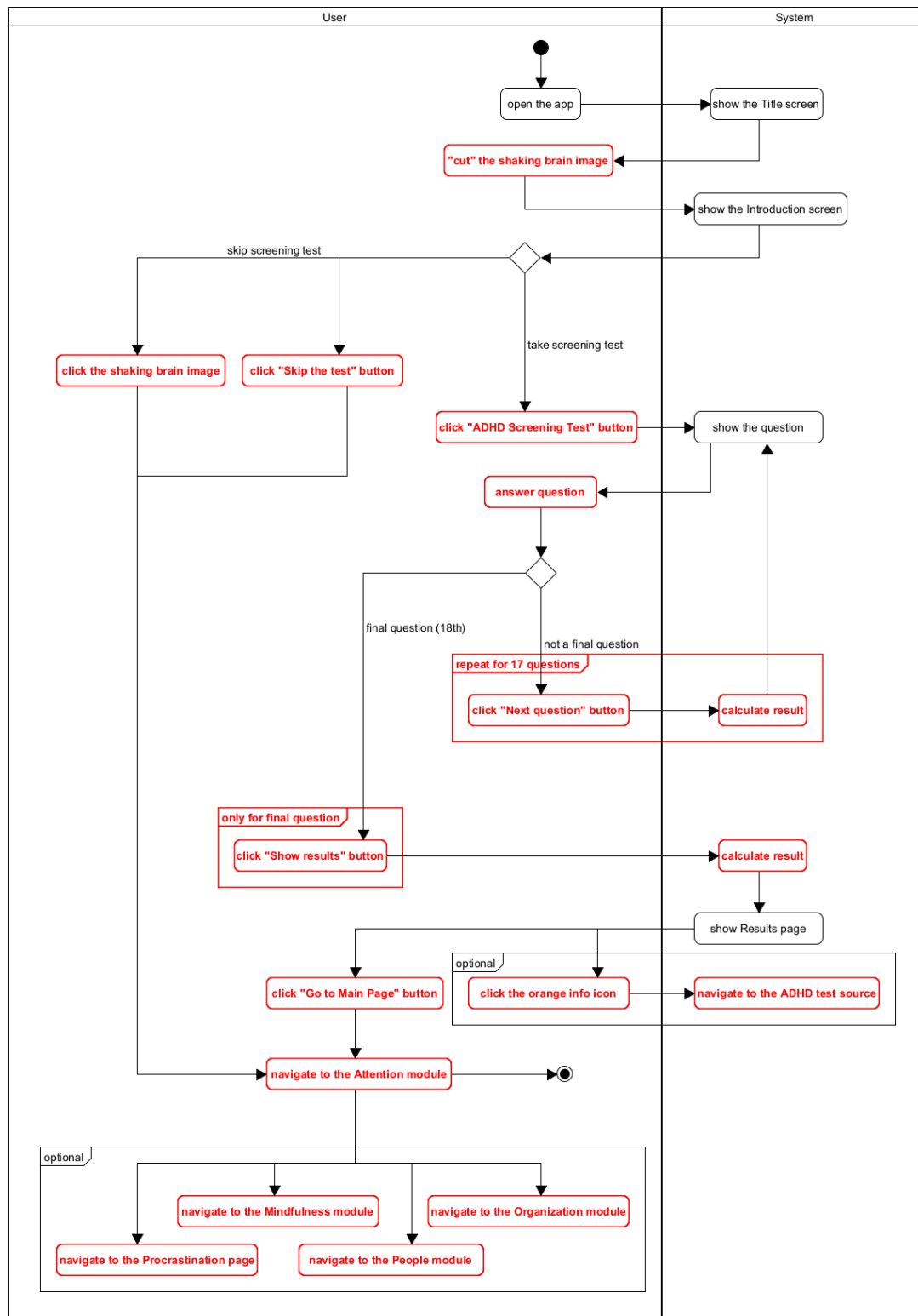


Figure 21. Use case diagram flow of the navigation, marking what has been implemented by the author (tool: UMLet [48], source: self)

5. Design of the proposed solution

This chapter is dedicated to describe the process of designing the application. The subchapters present the application's alignment with Nielsen's usability heuristics [50], initial hand-drawn ideas for the design and the digital design prototypes.

5.1. Initial design prototype

There were many iterations of the design. To showcase the milestones the author decided to present the highlights of the process. Examples of the interface elements of this application are presented below demonstrating the most important screens from the users' point of view as well as aesthetics and ergonomics and repeatability of the solutions used.

5.1.1. Paper prototype

The initial prototype was done on paper and includes some of the screens that are already a part of the application and also features the ones that are to be added in the foreseeable future (more on that in chapter 9. *Consideration of the future improvements*). As seen on figures 22 and 23 the paper prototype is only a mock design serving a purpose of visualizing the design and presenting the overall concept of the screens. Paper prototype was necessary to discuss what may be the essential elements that need to be included in the digital prototype. It also aids in getting the author's point across during the initial talks about the features of the application.

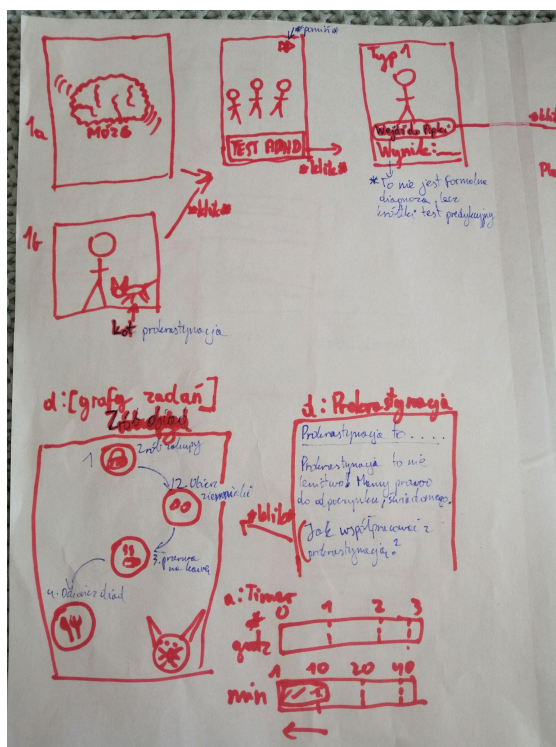


Figure 22. Paper prototype of the title screens
(source: self)

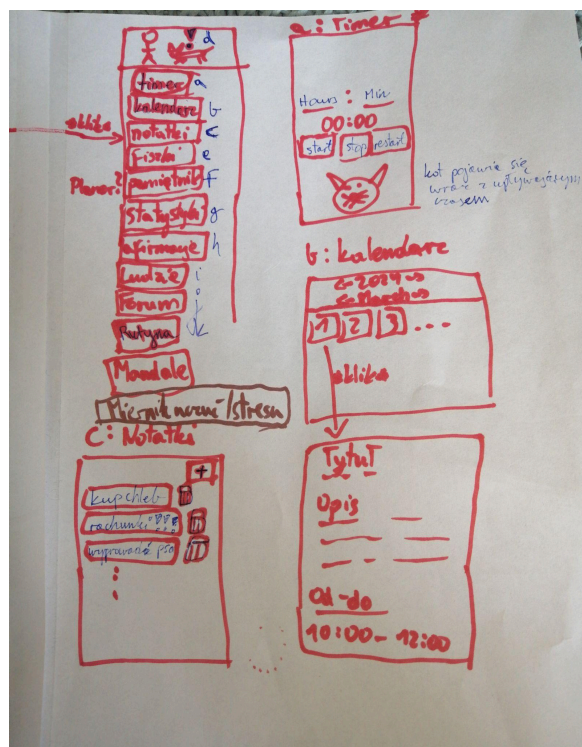


Figure 23. Paper prototype of the main screens
(source: self)

5.1.2. Digital prototype

Digital prototype was done using Marvel. As seen on figure 24 the title screens remained almost unchanged in concept to the paper prototype. The title screen (on the left) showcases the application references themselves. The introduction screen (on the right) showcases the 3 types of ADHD with a button prompting to take the ADHD screening test or skip it by going directly into the home screen.

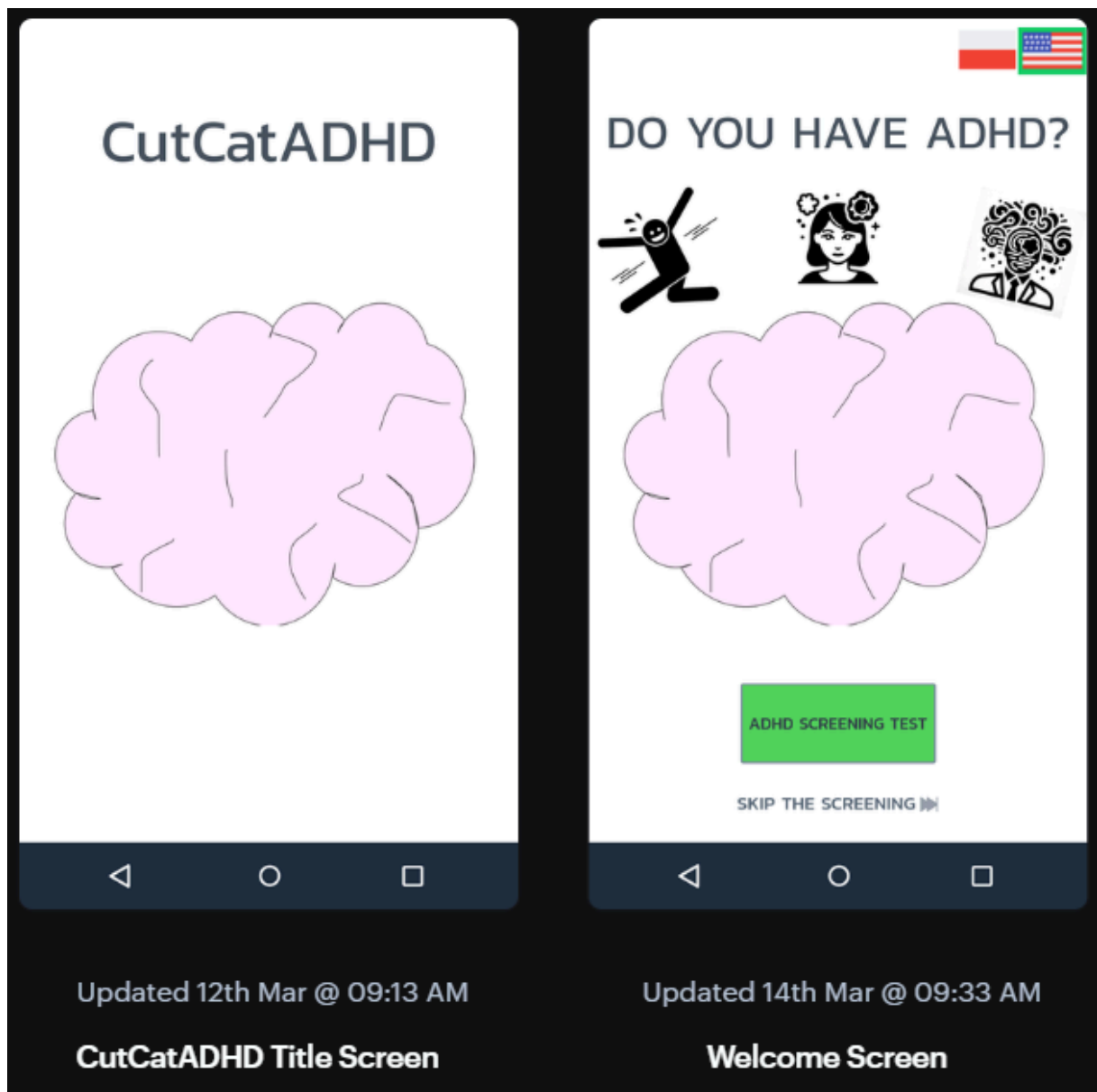


Figure 24. Digital prototype of title screen (on the left) and introduction screen (on the right) (tool: Marvel [51], source: self)

As seen on figure 25 the home screen shows 4 categories - Attention (Uwaga), Organization (Organizacja), People (Ludzie), Entertainment (Rozrywka). Each of these categories are suited for different needs and contain distinct functionality. Categorization makes it easier to organize these functionalities by their use cases. Additionally, the top banner shows one of the ADHD type avatars with a cat - referencing the title “CutCatADHD” itself and a play of words that cats, just like ADHDers, travel in their own direction. During the digital prototype phase, each of the categories had its own distinct color palette as seen on figure 26 - Attention homepage screen having reddish/brownish panel colors.



Figure 25. Digital prototype of the home screen (tool: Marvel [51], source: self)



Figure 26. Digital prototype of the Attention (Uwaga) homepage screen (tool: Marvel [51], source: self)

As seen on figure 27, the Timer page was very minimal but may have been still too distracting - too many elements with ambiguous functionality may lead to overwhelm. Even though the timer functionality was straightforward, the user interface was unfulfilling due to the actual Timer taking up only about half of the screen. The rest of the screen was filled with “CutCat” references even though they served no actual purpose. But the overall concept was there, and it gave the design idea for the later implementation.

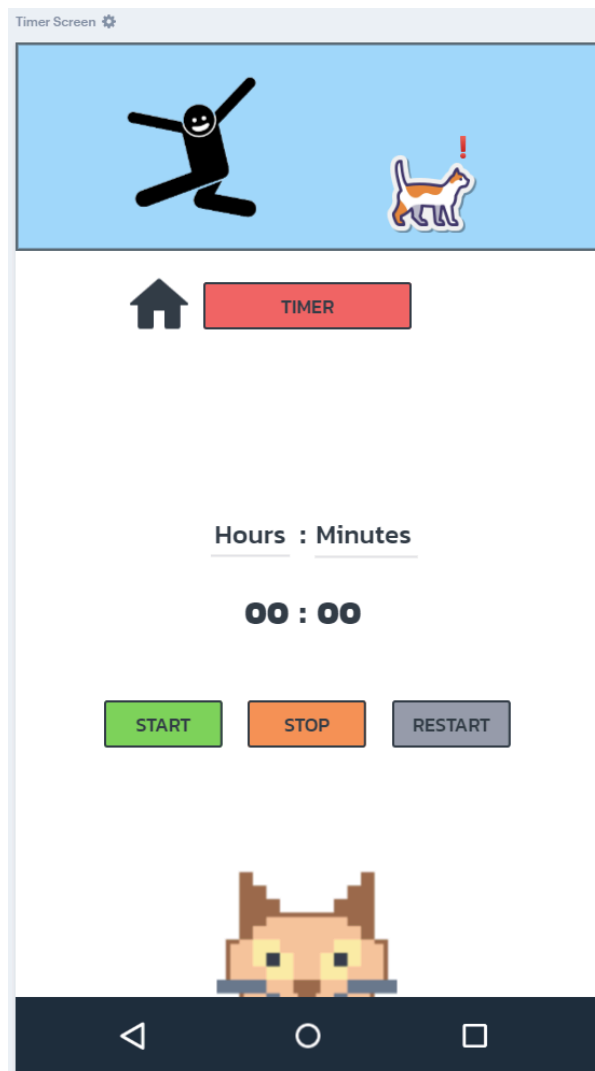


Figure 27. Digital prototype of the Timer page (tool: Marvel [51], source: self)

5.2. On-device design

After the satisfactory design prototype was done in the design software, the screens were implemented on device. In the process of developing the application many iterations have been made to meet the usability standards of actual usage as well as the software capabilities. This chapter presents some of the screens implemented in the process of making the project.

5.2.1. UI wireframe implementation

One of the first iterations of UI implementation, high fidelity wireframe, is shown on figure 28 presenting the Timer page.

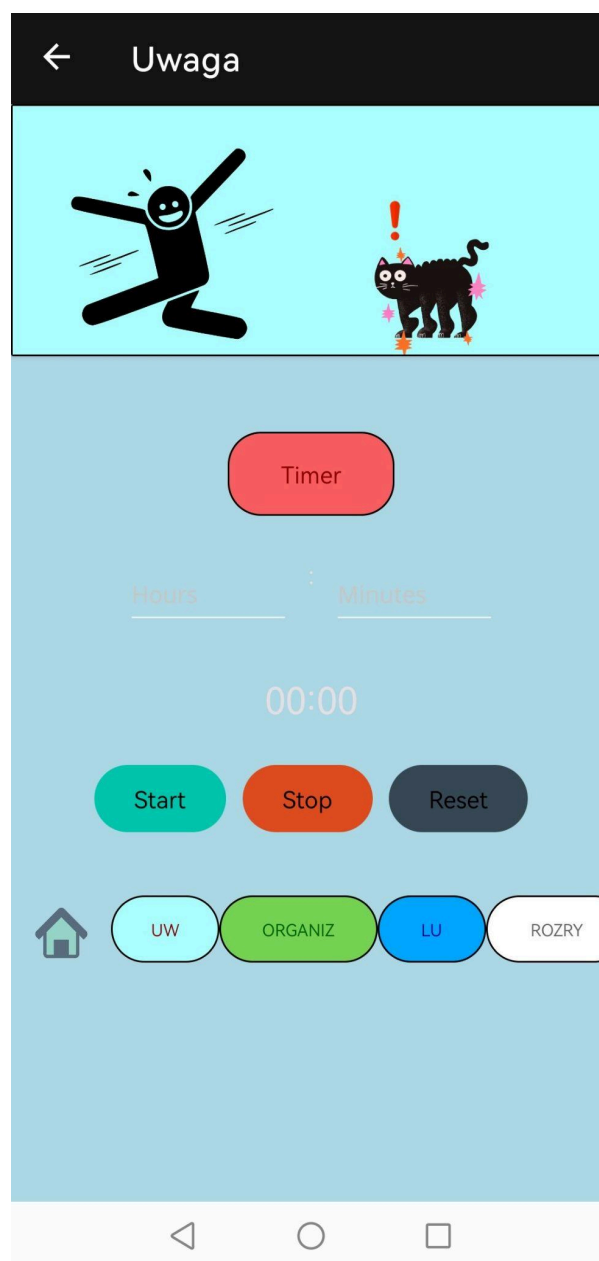


Figure 28. Wireframe implementation of the Timer page (source: self)

5.2.2. Final design iterations

After many iterations of the design, the UI ended up looking more minimal and clean. The main color palette became based on white colors while the action elements (eg. buttons) were given a distinct blueish-grayish color palette.

Screens presented in the following subchapters will concern the tasks that were given to test users during the testing phase of the application. The test results will be discussed later in chapter 8. *Conducting user tests*. Presented screens describe the following use cases:

1. Navigating to the Attention module (considered as task 0)
2. Creating custom routine
3. Completing the custom routine (displaying and checking all the routine steps)
4. Creating custom note (reminding about deleting the routine)
5. Visiting one of the forums
6. Deleting created earlier custom routine

The device used for all the user tests and presented on the screenshots is the Pixel 33 emulator running Android 13.0 operating system (API 33).

5.2.2.1. Navigating to the Attention module (considered as task 0)

As a new user, the action starts from the Title screen as shown on figure 29. From here, the user navigates to the Introduction page by following the call-to-action displayed on the Title page - “CUT it!”. The next screen is the Introduction screen, shown on figure 30. From here the user can take the screening test that redirects him to the Screening Test page consisting of 18 questions, or skip the test and navigate directly to the Attention page by clicking either on the brain image or the “Skip the test” label.



Do you have ADHD?

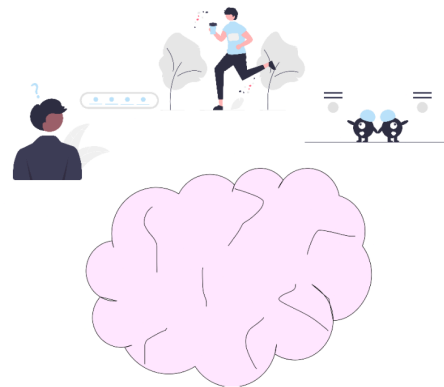


Figure 29. Current final design of the Title screen
(tool: Pixel 33 emulator, source: self)



Figure 30. Current final design of the Introduction
screen (tool: Pixel 33 emulator, source: self)

The Attention module as seen on figure 31 contains 4 panels - Timer, Notes, Knowledge base and Challenges.

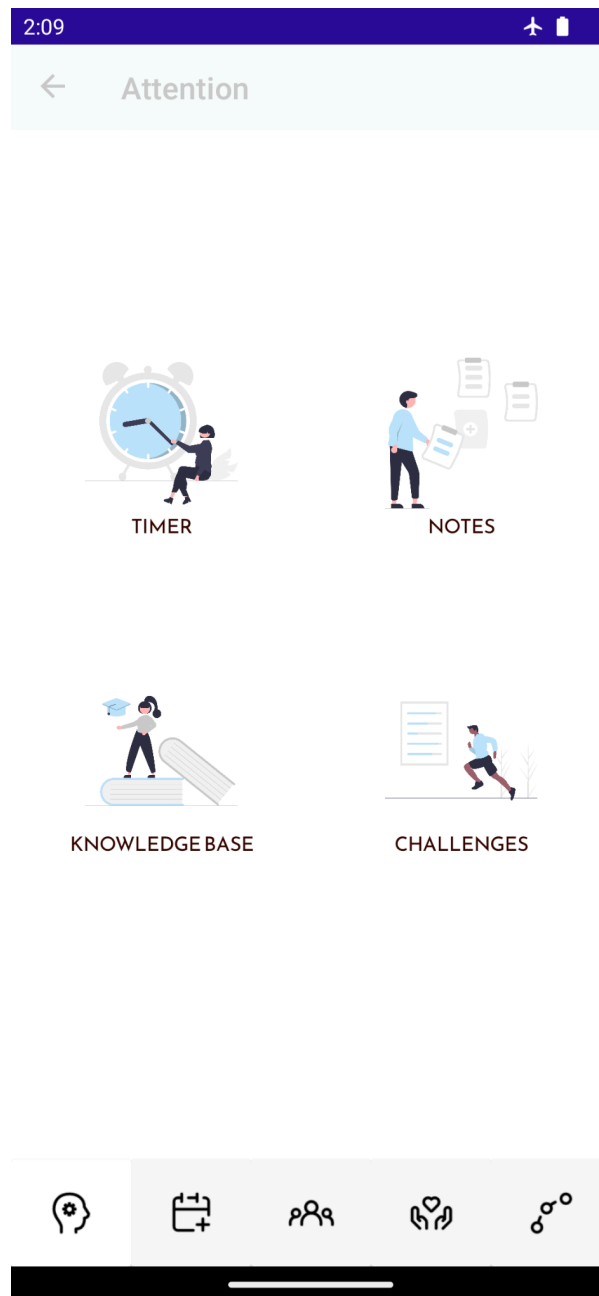


Figure 31. Current final design of the Attention page (tool: Pixel 33 emulator, source: self)

5.2.2.2. Creating custom routine

From the Attention module, the user clicks the "calendar +" icon located on the bottom navigation bar. This action navigates the user to the Organization module page as seen on figure 32. User then clicks the “ROUTINES” panel and navigates to the Routine Menu Page view as seen on figure 33. The Routine Menu Page contains 5 slots available for creating custom routines.

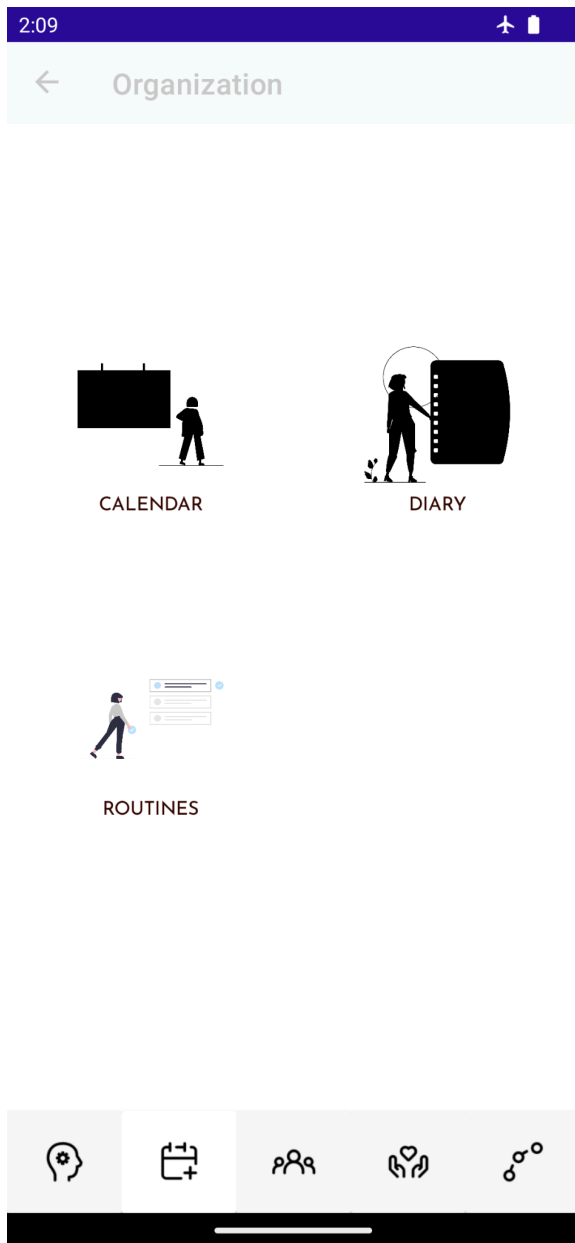


Figure 32. Current final design of the Organization module page (tool: Pixel 33 emulator, source: self)

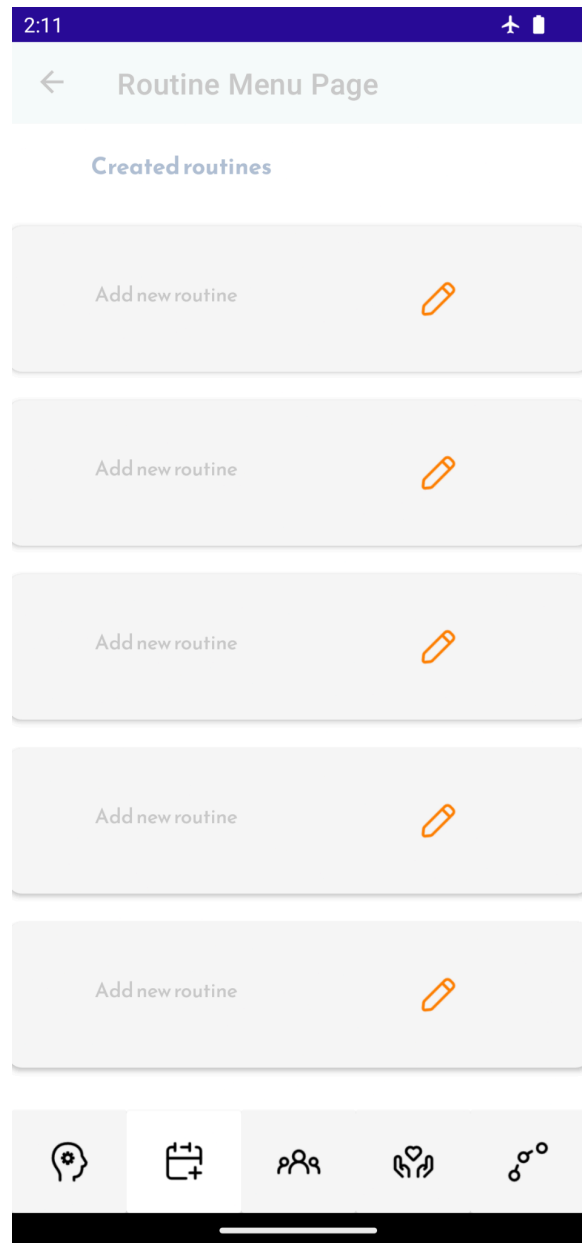


Figure 33. Current final design of the Routine Menu Page (tool: Pixel 33 emulator, source: self)

User clicks the "orange pencil" edit icon button next to the "Add new routine" label which navigates the user to the Edit Routine view as seen on figure 34. User fills in the routine title, description, 4 routine steps and submits the form by clicking the green "SAVE" button. System then notifies the user about a successful operation as seen on figure 35.

2:12

← Edit Routine

Enter routine name

Below you can describe the routine precisely

Enter note description

1. Enter step 1 of the routine

2. Enter step 2 of the routine

3. Enter step 3 of the routine

4. Enter step 4 of the routine

SAVE

CANCEL

Icons: Gear, Calendar, Group of people, Heart, Link

Figure 34. Current final design of the Edit Routine view (tool: Pixel 33 emulator, source: self)

2:13

← Edit Routine

routine 1

Below you can describe the routine precisely

sample description

1. first step

Success

Routine 'routine 1' saved successfully

OK

SAVE

CANCEL

Icons: Gear, Calendar, Group of people, Heart, Link

Figure 35. System notification about a successful routine creation (tool: Pixel 33 emulator, source: self)

By clicking the “OK” button the user gets navigated back to the Routine Menu Page. This time the newly created routine is available in the menu as seen on figure 36.

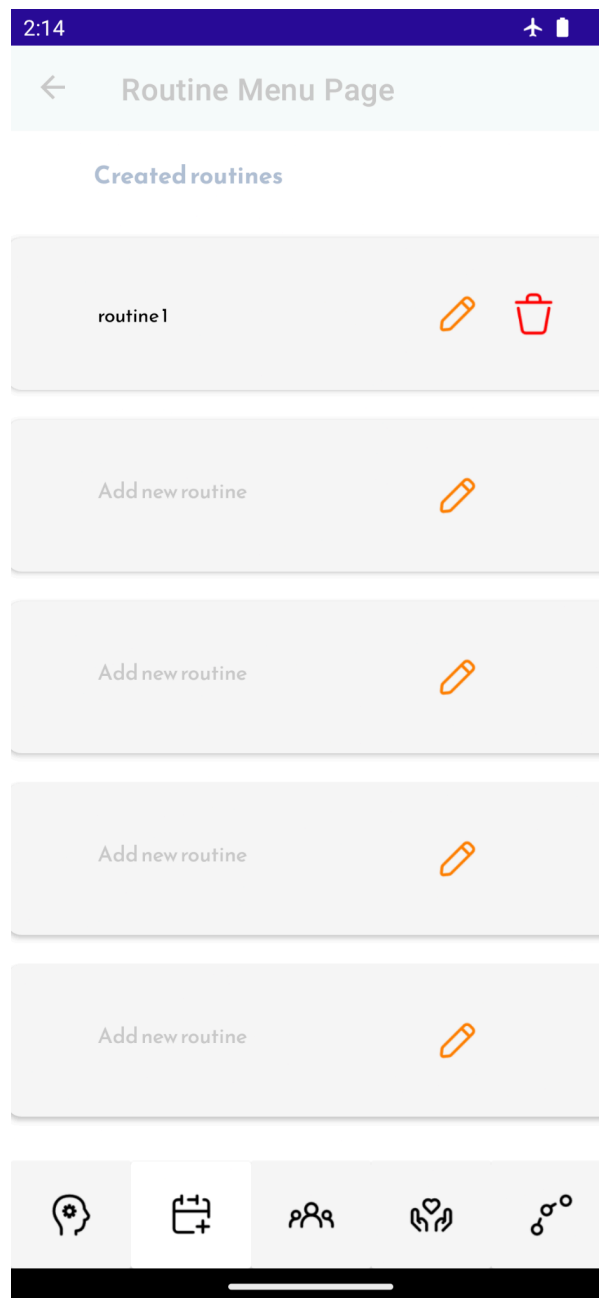


Figure 36. Routine Menu Page view with newly created custom routine (tool: Pixel 33 emulator, source: self)

5.2.2.3. Complete the custom routine

To complete the custom routine, starting from the Attention module page user shall first navigate to the Routines panel within the Organization module. The updated Routine Menu Page view as previously seen on figure 36 allows for accessing the custom routine by clicking its label (presented in black font). This navigates the user to the template page that is populated with the data of the chosen routine as seen on figure 37 on the next page (more on the data population in chapter 7. *Implementation of the project*).



Figure 37. Routine template page view populated with the data of the chosen routine (tool: Pixel 33 emulator, source: self)

User checks the steps of the routine by clicking the green checkboxes as seen on figure 38. When the user checks all the routine steps and so completes it the system shows a notification about a successful completion as seen on figure 39.



Figure 38. Custom routine view with some of the steps checked by the user (tool: Pixel 33 emulator, source: self)

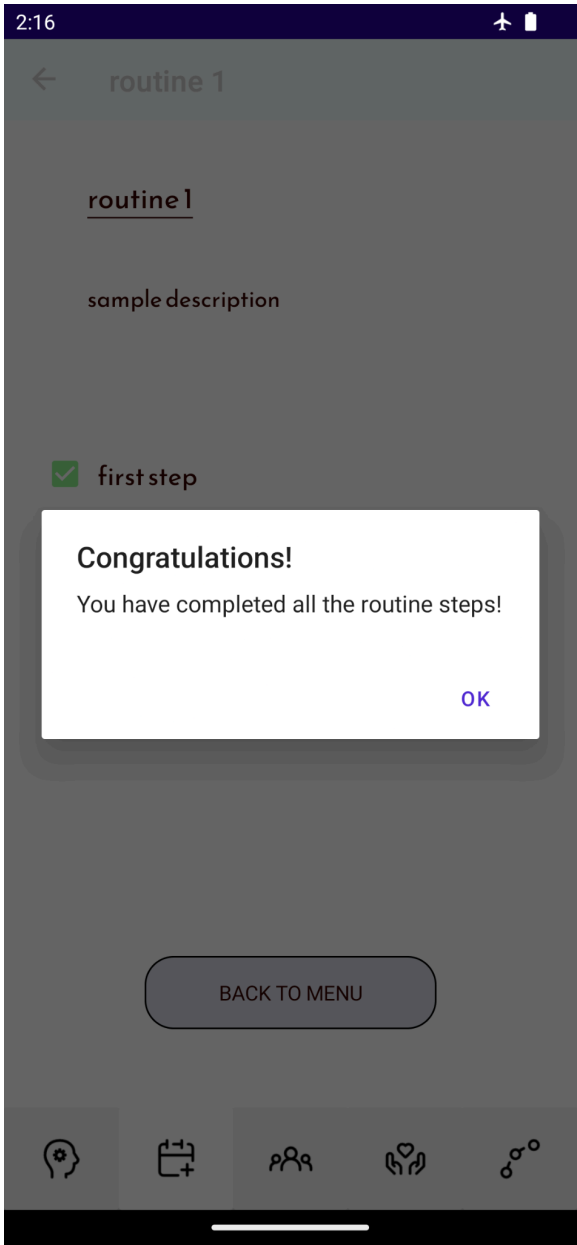


Figure 39. System notification about a successful routine completion (tool: Pixel 33 emulator, source: self)

5.2.2.4. Creating custom note (reminding about deleting the routine)

To create a custom note and remind perself about deleting the previously made routine user shall navigate to the “NOTES” panel available on the Attention module page as seen on figure 40. As seen on figure 41 the Notes Menu page consists of 5 slots for saving custom notes. To create a new note user shall click the "orange pencil" edit icon button next to the "Add new note" label which navigates the user to the Edit Note view.

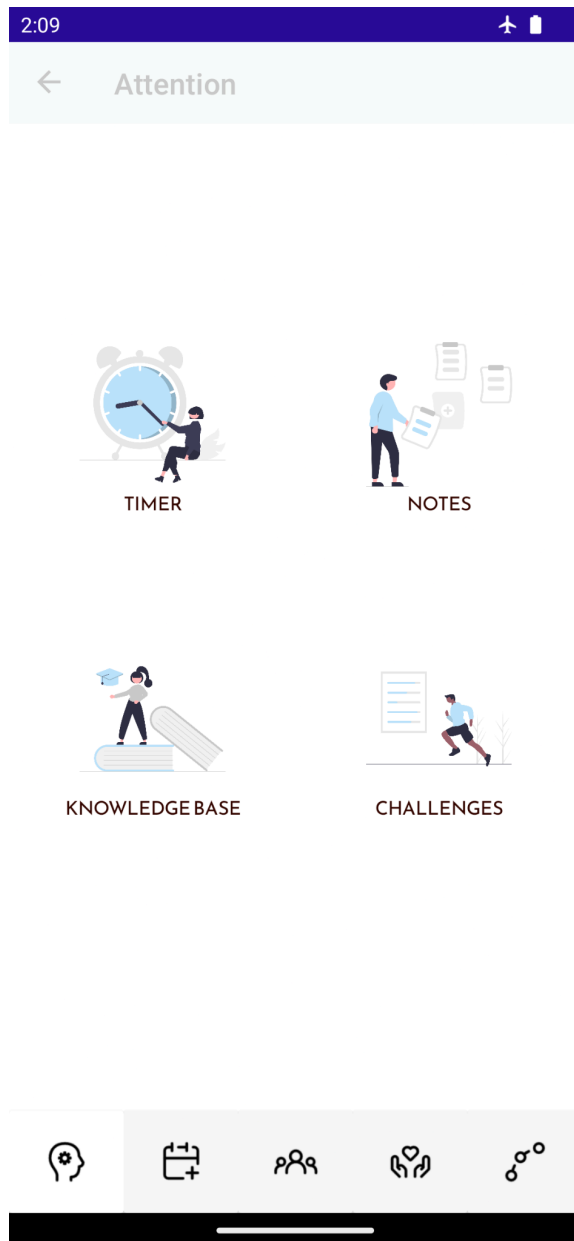


Figure 40. Current final design of the Attention module page (tool: Pixel 33 emulator, source: self)

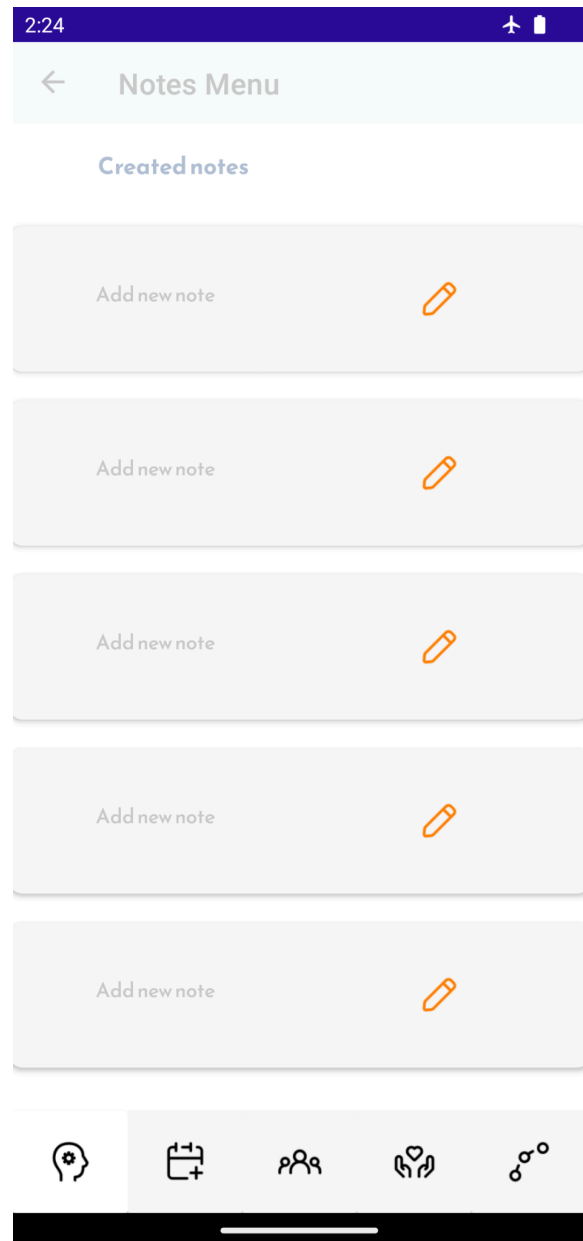


Figure 41. Current final design of the Notes Menu page (tool: Pixel 33 emulator, source: self)

Figure 42 shows the Edit Note page for the about to be created note. On the Edit Note page the user can add a note headline and description. After clicking the green “SAVE” button, the system saves the custom note and notifies the user about a successful creation as seen on figure 43.

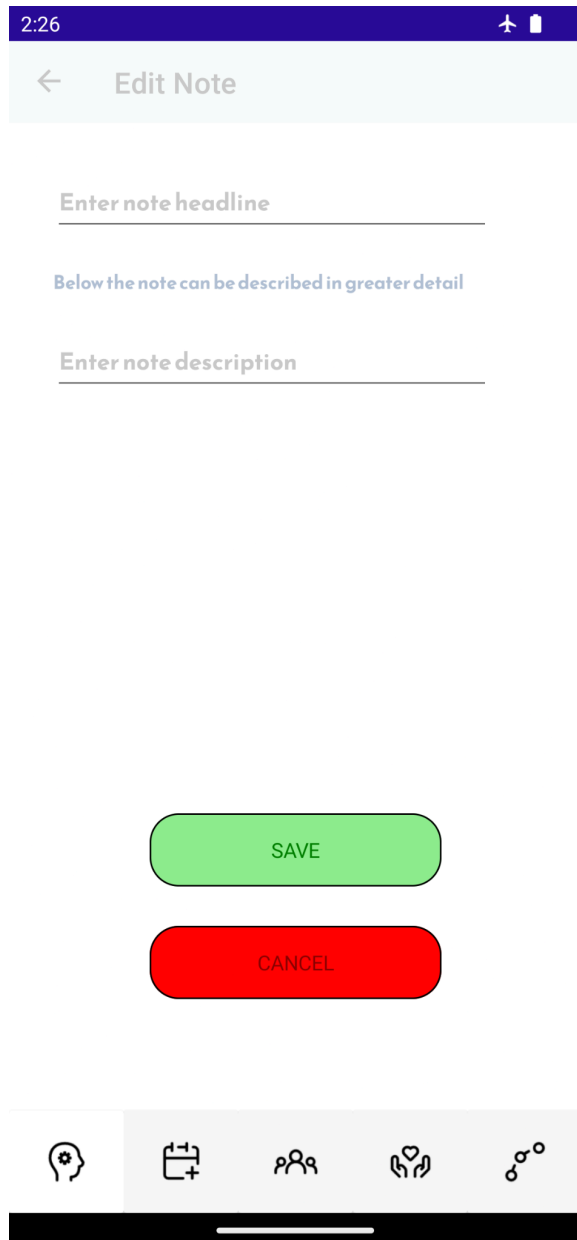


Figure 42. Current final design of the Edit Note view (tool: Pixel 33 emulator, source: self)

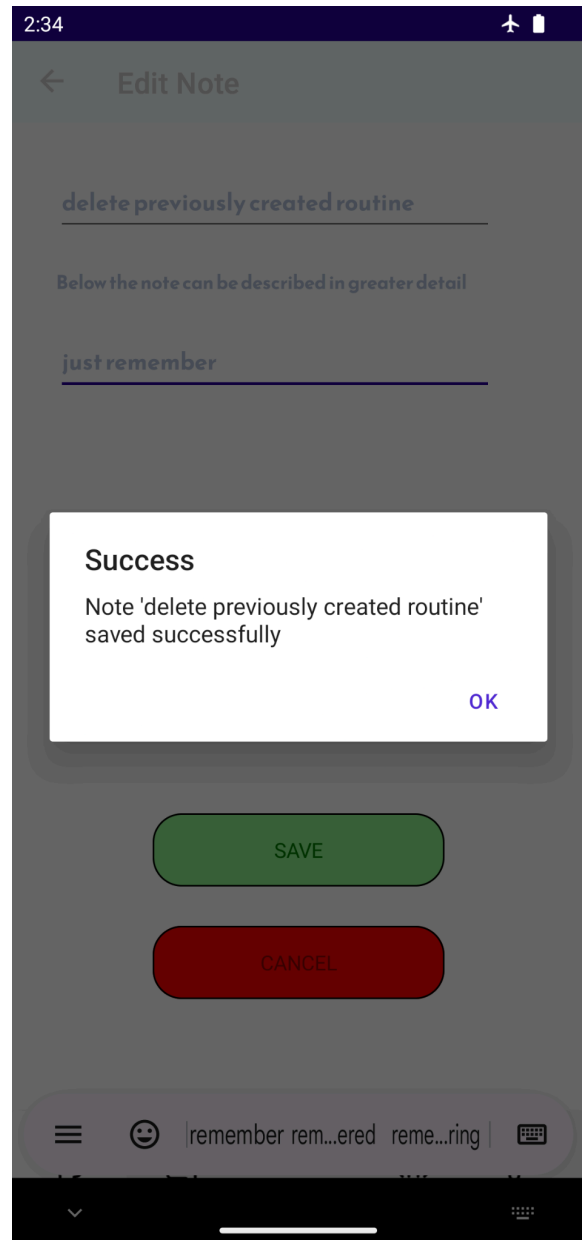


Figure 43. System notification about a successful note creation (tool: Pixel 33 emulator, source: self)

5.2.2.5. Visiting one of the forums

By utilizing the bottom navigation bar the user can navigate to any of the main modules by clicking the appropriate icon. To navigate to the People module user shall click the “people” icon. Figure 44 shows the People module page. To navigate and display Forums Page users shall click the “FORUMS” panel. Forums Page lists some of the reliable internet forums¹⁰ as seen on figure 45. To navigate to the given forum user shall click the blue “internet” icon next to the forum’s label. This action takes the user outside the application and opens up the forum’s website in a default browser. Important thing to notice is that the Forums Page works offline (indicated by the airplane mode turned on as in the top right corner of figure 45) just like any other page in the application. However to access the external resources like the forum’s website the user must have an internet connection.

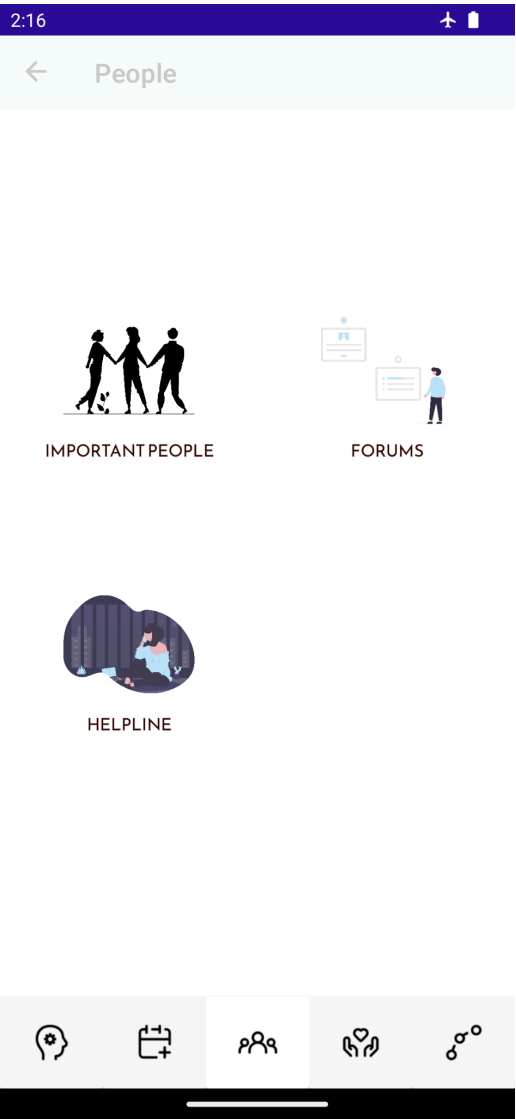


Figure 44. Current final design of the People module page (tool: Pixel 33 emulator, source: self)

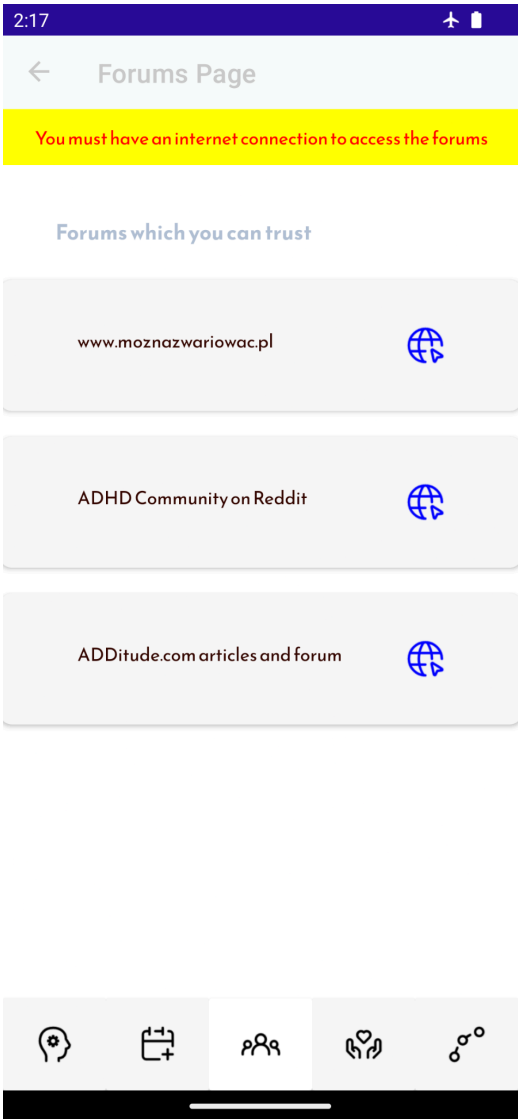


Figure 45. Forums Page view (tool: Pixel 33 emulator, source: self)

¹⁰ The current listing of the forums may change in the future iterations of the application.

5.2.2.6. Deleting created earlier custom routine

As mentioned earlier, by utilizing the bottom navigation bar the user can navigate to any of the main modules by clicking the appropriate icon. To navigate back to the Routine Menu Page the user shall click the “calendar +” icon and then the “ROUTINES” panel itself. To delete the routine, the user shall click the red “trash” delete icon next to the routine’s label. This triggers the system to first ask the user for deletion confirmation as shown on figure 46. Pressing “Yes” irreversibly deletes the routine. For confirmation, the system notifies the user about a successful delete operation as seen on figure 47. The system “refreshes” the page (details of the implementation are described in chapter 7. *Implementation of the project*) and shows the Routine Menu Page without the just-deleted routine.

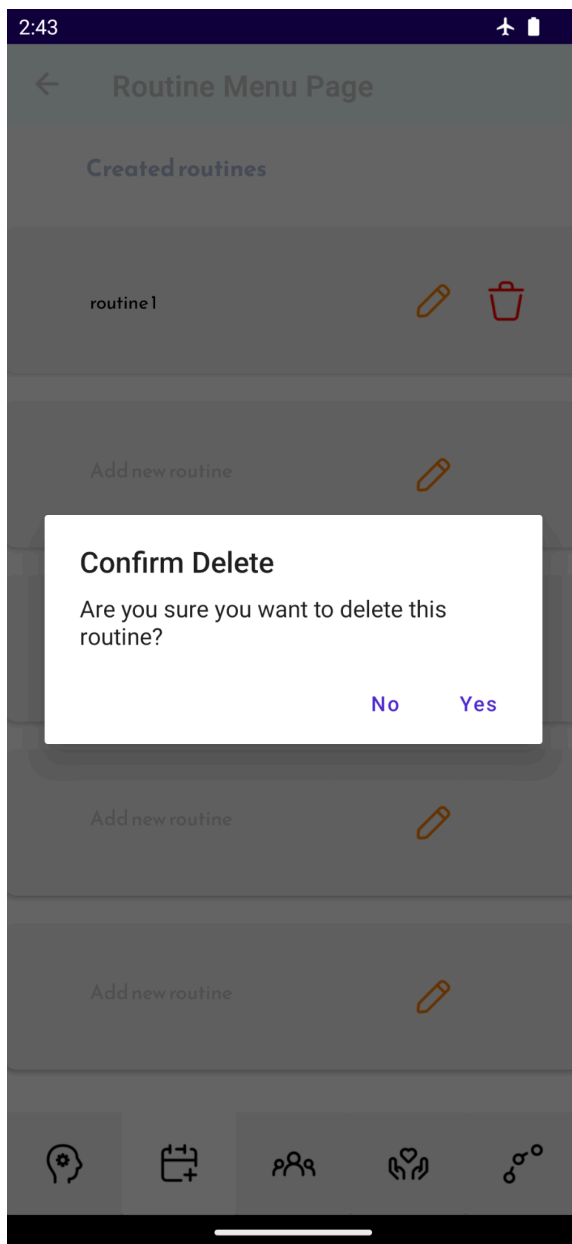


Figure 46. System prompts the user for a deletion confirmation (tool: Pixel 33 emulator, source: self)

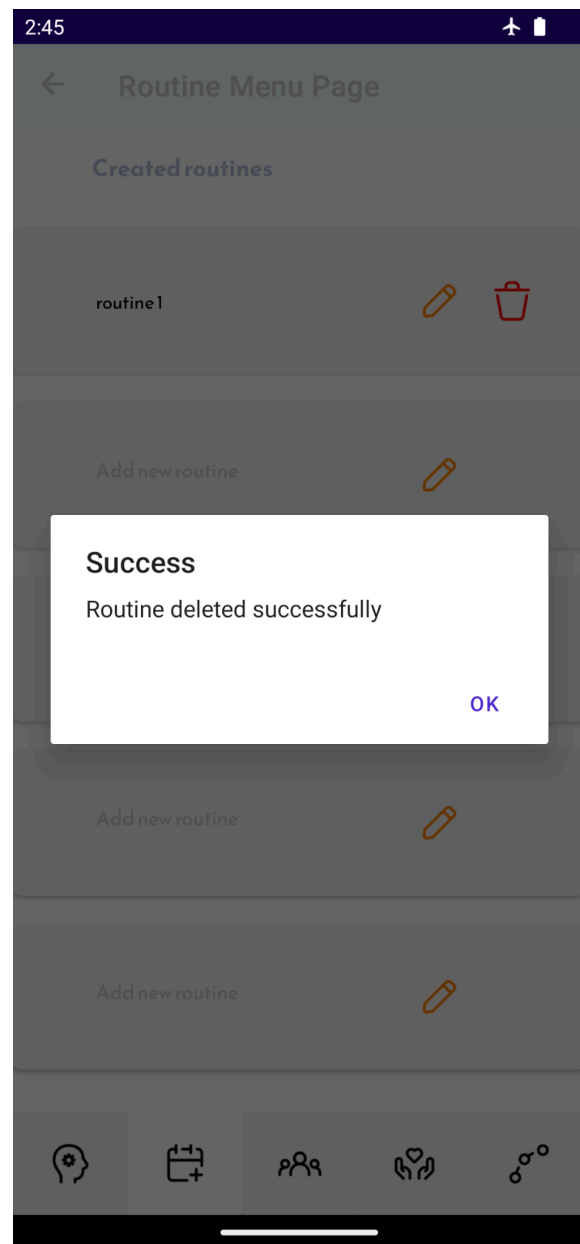


Figure 47. System notification about a successful routine deletion (tool: Pixel 33 emulator, source: self)

6. Technologies used for the implementation of the project

This chapter will present all the major technologies used for creation of the "CutCatADHD" mobile application as well as the documentation. Because the author wants to cherish the libre software movement but also be blunt in the way he uses software, this chapter is divided into subchapters categorizing whether the used technology is libre, open-source or proprietary.

6.1. Libre/Free software

“Free software” means software that respects users' freedom and community. Roughly, it means that the users have the freedom to run, copy, distribute, study, change and improve the software. Thus, “free software” is a matter of liberty, not price. To understand the concept, you should think of “free” as in “free speech,” not as in “free beer.” We sometimes call it “libre software,” borrowing the French or Spanish word for “free” as in freedom, to show we do not mean the software is gratis.”[52]

As seen on figure 21 for a program to be free it must guarantee its users the four essentials freedoms:

- *“The freedom to run the program as you wish, for any purpose (freedom 0).*
- *The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.*
- *The freedom to redistribute copies so you can help others (freedom 2).*
- *The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.”[53]*

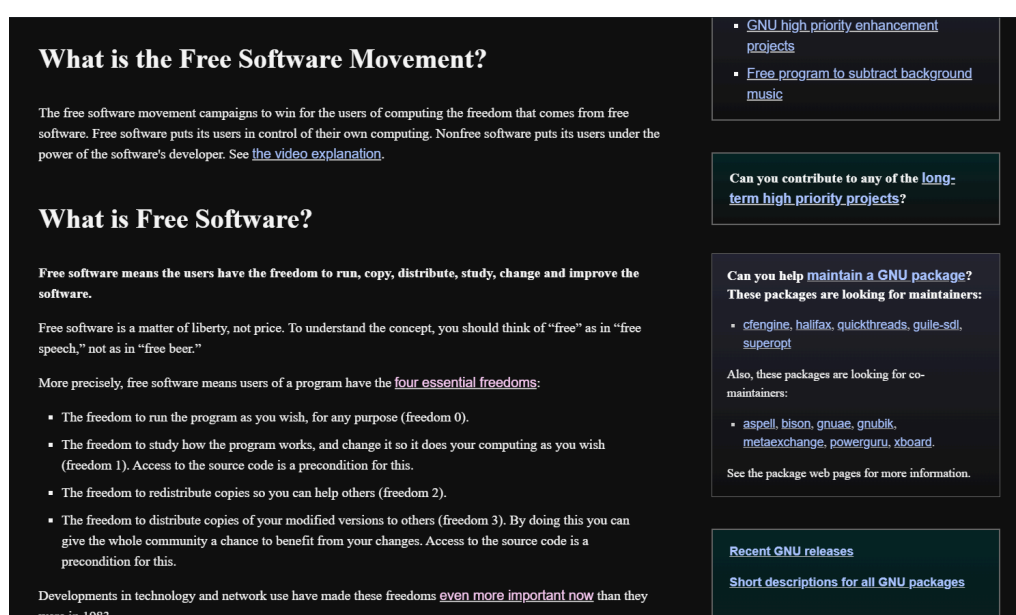


Figure 48. Homepage of GNU Operating System website (source: [54])

6.1.1. UMLet

6.1.1.1. Introduction

“UMLet is a free, open-source UML tool with a simple user interface: draw UML diagrams fast, create sequence and activity diagrams from plain text, share via exports to eps, pdf, jpg, svg, and clipboard, and develop new, custom UML elements.”[55]

6.1.1.2. History

UMLet was created as an open-source, Java-based UML tool designed for teaching the Unified Modeling Language (UML) and quickly creating UML diagrams as seen on figure 69. It was developed with the goal of providing a lightweight and user-friendly alternative to more complex and cumbersome UML modeling tools. [55]

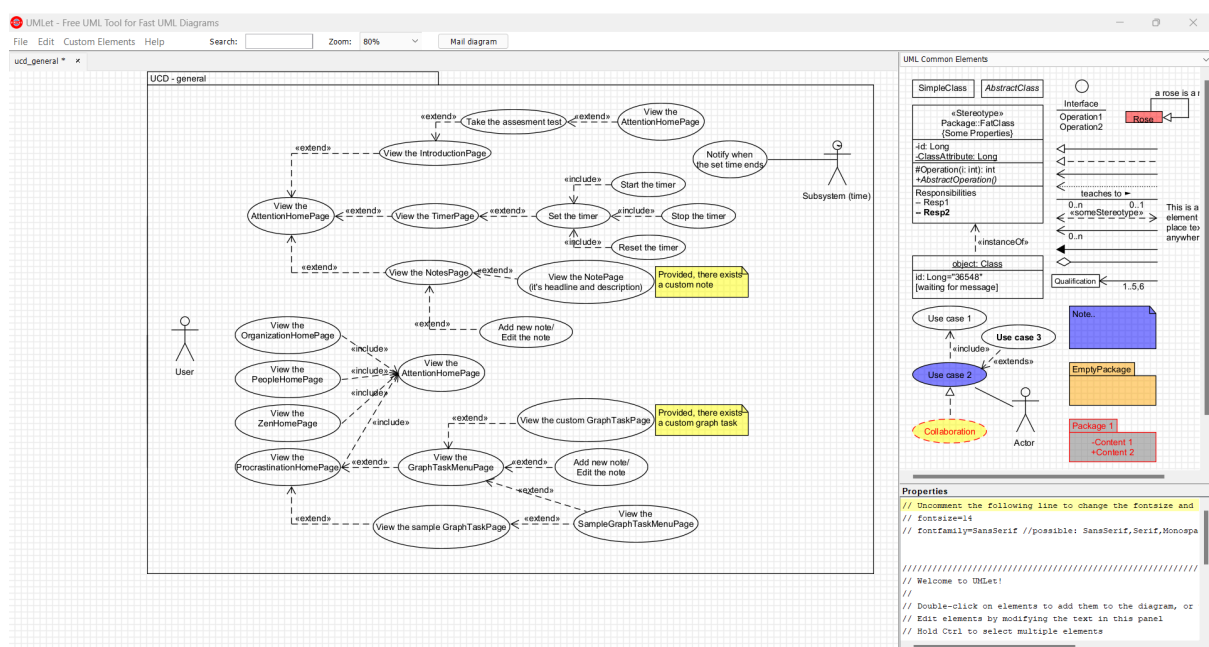


Figure 49. General use case diagram for the “CutCatADHD” application made with UMLet (tool: UMLet [48], source:self)

The key ideas behind UMLet's design were outlined in a paper titled "Flyweight UML Modelling Tool for Software Development" by Martin Auer, Thomas Tschurtschenthaler, and Stefan Biffl, published in the proceedings of the 29th EUROMICRO Conference. This paper described UMLet's approach of using a simple text-based markup language for modifying UML elements, rather than relying on complex dialogs or icons.

UMLet appears to have been available and actively developed since the early 2000s. The project's GitHub repository traces its history back to 2001, when it was released under the GNU General Public License (GPL) [56]. Over the years, UMLet has undergone several releases and updates, introducing new features and improvements. Some notable milestones include:

- **Version 10.3** (around 2010): Updates to the user interface.
- **Version 11.0** (around 2011): Introduction of a list of recently opened files, drag-and-drop support for UXF files (UMLet's native file format), and updates to the file format.
- **Version 13.0** (around 2013): Internal refactoring and the addition of context-sensitive help.
- **Version 15.0** (around 2015): Introduction of web features like zoom, lasso, export, dark mode, high-resolution export, and improved startup performance.

In addition to the standalone application, UMLet has been adapted for other platforms and environments, such as an Eclipse plugin and a web-based version called UMLetino. More recently, it has been integrated as an extension for Visual Studio Code.

Throughout its history, UMLet has maintained its focus on providing a simple and efficient tool for creating UML diagrams, while also allowing users to customize and extend the software through features like custom UML elements and batch processing capabilities.

6.1.2. Git

6.1.2.1. Introduction

Git is a distributed version control system that allows developers to track changes to their codebase over time. It enables multiple developers to work on the same codebase simultaneously, merge their changes, and maintain a complete history of all modifications. Git operates by creating “snapshots” of the codebase at different points in time, rather than tracking individual file changes [57]. Git is free and open code (in fact it is free software) as shown on figure 50.

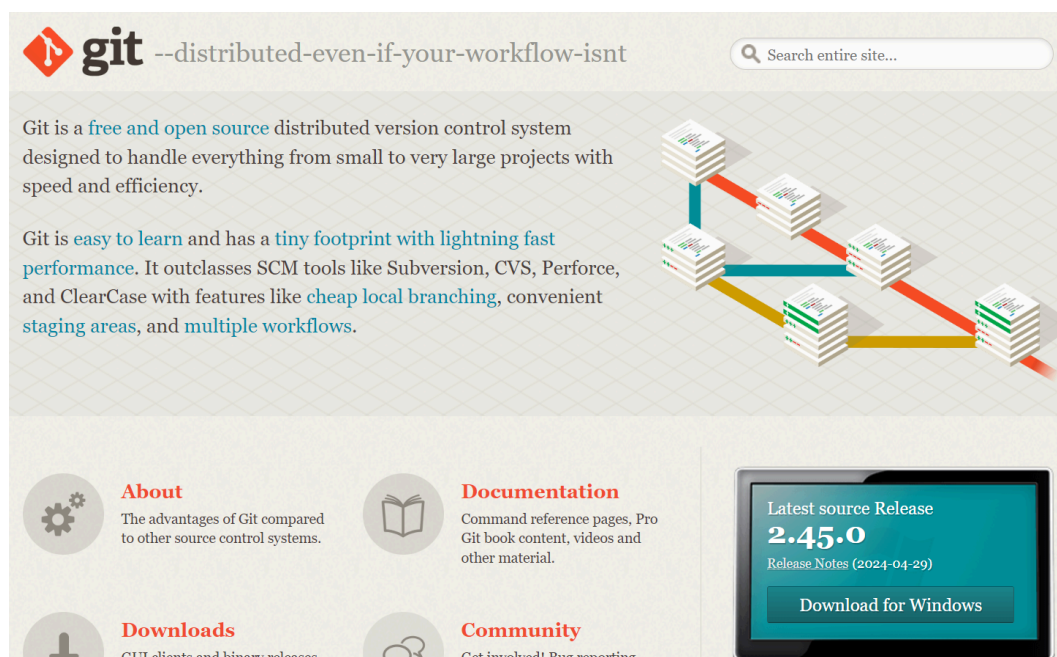


Figure 50. Homepage of GNU Operating System website (source: git homepage [58])

6.1.2.2. History

To understand the usefulness and importance of git, a short introduction to the Linux kernel is needed. The Linux kernel is an open source software project of fairly large scope. During the early years of the Linux kernel maintenance (1991–2002), changes to the software were passed around as patches and archived files. In 2002, the Linux kernel project began using a proprietary distributed version control system called BitKeeper [59]. Some, including GNU Project founder Richard Stallman, expressed concern about proprietary tools being used on a flagship free project. [60]

It was initially offered for free to the Linux community with BitKeeper's main claim to fame that it offered a distributed system, whereby whole repositories could be forked and merged easily. This was the key. However, BitMover (the company that developed BitKeeper) put significant restrictions on the Linux community in exchange for the non-pay license. First, Linux developers would not be allowed to work on competing revision control projects while using BitKeeper. And second, BitMover would control certain metadata related to the kernel project, in order to notice any abuse of the license.

Without access to that metadata, kernel developers would be unable to compare past kernel versions—an important standard feature of other revision control systems.

Existing projects like CVS and Subversion could do forks and merges only as major, time-consuming operations. With BitKeeper, it became a trivial operation which was the main reason to keep using it despite it being proprietary software in the heart of libre Linux kernel. Many kernel developers were not happy and the controversy did not die down, although Linus Torvalds continued to rely on BitKeeper for years.

In 2005, Andrew Morton tried to reverse-engineer the BitKeeper networking protocols in order to create a free software alternative. BitMover had warned the Linux developers that it would pull the plug if anyone tried this, and that's exactly what happened. Suddenly, BitKeeper could no longer be used for kernel development. The entire development toolchain, and all the developer culture that had sprung up around distributed version control, was thrown into uncertainty. [61]

This prompted the Linux development community (and in particular Linus Torvalds, the creator of Linux) to develop their own tool based on some of the lessons they learned while using BitKeeper. Some of the goals of the new system were as follows:

- Speed
- Simple design
- Strong support for non-linear development (thousands of parallel branches)
- Fully distributed
- Able to handle large projects like the Linux kernel efficiently (speed and data size)

Torvalds took inspiration from existing version control systems like BitKeeper, but also drew ideas from the Monolithic Version Control System used for the Linux kernel previously.

The result was Git - a free and open source distributed version control system. Git was initially designed as a low-level version control toolset on which others could write higher-level user interfaces and tools. However, it quickly gained popularity within the open source community. By 2008, several Linux distributions had adopted Git.

The name "Git" was derived from the British slang word "git" meaning "uncompromising person".

Some key milestones in Git's version history include:

- **Initial Release** (April 2005): The first self-hosted version of git
- **Git 1.5.0** (February 24, 2007): This version marked a significant step towards making Git more user-friendly. Key improvements included the better organization of commands into high-level ("porcelain") and low-level ("plumbing") commands, which made the system more approachable for new users. The introduction of features like 'git stash' and 'git svn' allowed users to save local modifications temporarily without committing them and provided a bridge to Subversion, respectively.
- **GitHub Launch** (April 10, 2008): While not a Git software update, the launch of GitHub was pivotal in Git's history. GitHub provided a user-friendly web interface for hosting Git repositories and simplified many Git operations. This platform significantly lowered the barrier to entry for using Git, making it accessible to developers who were less comfortable with command-line tools.
- **Git 1.6.0** (September 17, 2008): This release focused on refining the user interface and improving documentation. The introduction of a standalone 'git remote' command helped users manage connections to remote repositories more easily. Enhanced configuration options for 'git fetch' and 'git pull' improved the handling of updates from remote sources, making these commands more intuitive.
- **Git 1.7.0** (March 17, 2010): This version introduced organizational changes that made Git commands more logical and grouped related functionality together. Improvements in network protocols and configuration made Git more flexible and easier to integrate with varying workflows.
- **Git 2.0.0** (May 28, 2014): One of the major changes was the modification of the default push behavior to 'simple', which helped prevent unintended updates to remote branches. This release also focused on improving performance and scalability, addressing issues related to handling large repositories.
- **Git 2.9.0** (June 13, 2016): This release brought several performance enhancements, particularly in how Git handled large numbers of submodules and improvements in the diff algorithm. These changes were crucial for developers working on complex projects with many components.
- **Git 2.26** (March 22, 2020): Responding to a broader conversation about terminology and inclusivity in tech, Git began to shift the default branch name from 'master' to 'main'. This change aimed to make the tech community more welcoming and to reconsider the impact of terminology used in software development. The release also included performance optimizations, particularly benefiting operations in large-scale repositories.

The latest stable version of Git as of 2024 is Git 2.39, released in March 2024. Git continues to receive regular updates and improvements from its maintainers and the open-source community. Today, Git has become one of the most widely adopted version control systems in the world. It is used by millions of developers across companies, open source projects, and individual developers for tracking changes in computer files.

6.1.3. SQLite

6.1.3.1. Introduction

SQLite is a self-contained, serverless, zero-configuration, transactional SQL database engine as seen on figure 51. Unlike traditional client-server database management systems, SQLite is an embedded database, meaning that the entire database is stored in a single cross-platform file on the host device.

SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file. The database file format is cross-platform - you can freely copy a database between 32-bit and 64-bit systems or between big-endian and little-endian architectures. [62]



Figure 51. Homepage of SQLite website (source: SQLite homepage [63])

6.1.3.2. History

Throughout its history, SQLite has continuously evolved, adding new features and capabilities while maintaining its core principles of being a self-contained, serverless, and zero-configuration SQL database engine. These milestones highlight some of the most significant enhancements that have contributed to SQLite's widespread adoption and popularity.

- ➔ **SQLite 1.0** (2000) - The first public release of SQLite, a self-contained, serverless, zero-configuration, transactional SQL database engine.
- ➔ **SQLite 2.8.0** (2004) - Introduced the VACUUM command to rebuild the entire database file by defragmenting it.
- ➔ **SQLite 3.0.0** (2004) - A major release that added support for UTF-8 and UTF-16 encoding, as well as improved concurrency control.
- ➔ **SQLite 3.3.6** (2006) - Introduced the Write-Ahead Logging (WAL) feature, which significantly improved performance for workloads with many concurrent writers.
- ➔ **SQLite 3.6.19** (2009) - Added support for Recursive Triggers, allowing triggers to modify or insert into the same table that caused the trigger to fire.
- ➔ **SQLite 3.7.0** (2010) - Introduced the PRAGMA optimize command, which can rebuild the database to better utilize available space and improve query performance.
- ➔ **SQLite 3.8.0** (2013) - Added support for Foreign Key Constraints, allowing developers to enforce referential integrity rules.
- ➔ **SQLite 3.25.0** (2018) - Introduced Window Functions, providing analytical capabilities similar to those found in other database systems.
- ➔ **SQLite 3.31.0** (2020) - Added support for Recursive Common Table Expressions (CTEs), enabling more complex queries and data manipulation.
- ➔ **SQLite 3.35.0** (2021) - Introduced the ability to create Temporary Triggers, which are automatically dropped when the session ends.
- ➔ **SQLite 3.39.0** (2022) - Added support for SQL Scalar Subqueries, allowing subqueries to be used in more contexts, such as in the SELECT list or in expressions.

6.1.3.3. Public domain license

SQLite is in the public domain which means that it can be used for any purpose, including proprietary uses, without obligations [64]. It is technically less restrictive than the famous section 7 of the GPLv2 license, so called “Liberty or Death” clause as seen on figure 52, which states that licensees may distribute a GPL-covered work only if they can satisfy all the license's terms, regardless of any other legal constraints they may be subject to [65]. GPLv2 license is considered a strong copyleft because it strictly forces the user to make the software and its modifications libre.

Public domain license on the other hand, gives the user complete freedom over the software and its modifications, including informally speaking not caring about the license at all.

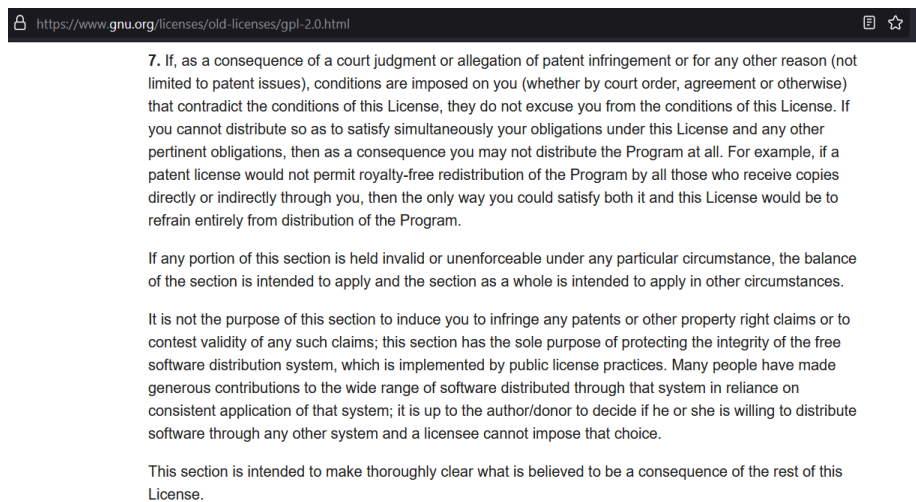


Figure 52. Section 7 of the GPLv2 license, the so-called "liberty or death" clause¹¹ (source: GNU GPL v2 license [66])

6.1.3.4. Reasons to choose SQLite

The author chose SQLite primarily because it is libre software. Secondly, it has an incredibly small footprint with the entire library being less than 750 KB in size, making it one of the most compact database management systems available [67]. Everything in this database lives in a single file (with a .sqlite extension) that can be put anywhere in the filesystem [68].

Thirdly, SQLite is serverless and self-contained, requiring no installation or configuration. It can be used by simply accessing the disk file directly. Most relational database engines are implemented as a server process in which programs communicate with the host server through an interprocess communication that relays requests. In contrast, SQLite allows any process that accesses the database to read and write to the database disk file directly. This simplifies SQLite's setup process, since it eliminates any need to configure a server process. Likewise, there's no configuration necessary for programs that will use the SQLite database: all they need is access to the disk [69].

¹¹ The “Liberty or Death” clause was introduced in GPL version 2 (section 7) and then continued in GPL version 3 (section 12). In GPLv3, it was renamed from “Liberty or Death” to “No Surrender of Others’ Freedom”, but its core purpose remained the same [44].

6.2. Open-source software

“The terms “free software” and “open source” stand for almost the same range of programs. However, they say deeply different things about those programs, based on different values. The free software movement campaigns for freedom for the users of computing; it is a movement for freedom and justice. By contrast, the open source idea values mainly practical advantage and does not campaign for principles. This is why we do not agree with open source, and do not use that term. The obvious meaning for the expression “open source software” is “You can look at the source code.” Indeed, most people seem to misunderstand “open source software” that way. (The clear term for that meaning is “source available.”) That criterion is much weaker than the free software definition, much weaker also than the official definition of open source. It includes many programs that are neither free nor open source.” [47]

6.2.1. .NET MAUI

6.2.1.1. Introduction

.NET Multi-platform App UI (.NET MAUI) is a cross-platform framework from Microsoft for creating native mobile and desktop applications using C# and XAML. It allows developers to build apps that can run on multiple platforms including Windows, Android, iOS, and macOS from a single shared codebase and project file as seen on figure 53.

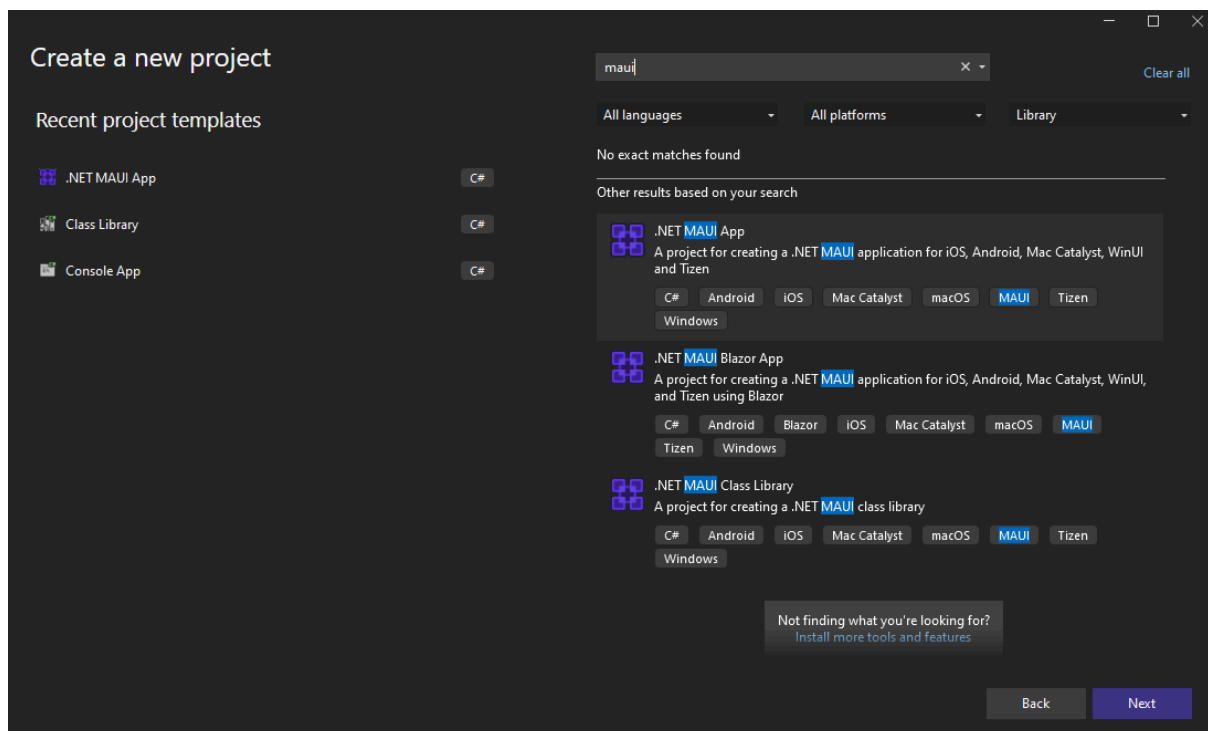


Figure 53. Visual Studio starter screen presenting the option to create a .NET MAUI App (tool: Visual Studio IDE [70], source: self)

.NET MAUI is open-source and is the evolution of Xamarin.Forms, with UI controls rebuilt from the ground up for performance and extensibility. Using .NET MAUI, developers can create multi-platform apps using a single project, but can add platform-specific source code and resources if necessary. One of the key aims of .NET MAUI is to enable developers to implement as much of the app logic and UI layout as possible in a single code-base. [71]

6.2.1.2. History

Xamarin.Forms was introduced in 2014 as part of the Xamarin platform, which Microsoft acquired in 2016. Xamarin.Forms allowed developers to build cross-platform mobile apps using C# and XAML, with a shared codebase for iOS, Android, and Windows platforms. While Xamarin.Forms was a significant step forward in cross-platform development, it had limitations in terms of performance, platform integration, and keeping up with the latest platform advancements.

As mobile and desktop platforms evolved, Microsoft recognized the need for a more modern, efficient, and unified solution for building apps across multiple platforms. This led to the development of .NET MAUI, which aimed to address the shortcomings of Xamarin.Forms and provide a more cohesive and future-proof approach to cross-platform app development.

Some key milestones in .NET MAUI version history include:

- ➔ **.NET MAUI Preview 1** (January 2021): The first preview of .NET MAUI introduced the initial implementation of the new cross-platform UI toolkit. It allowed developers to create apps targeting Windows, Android, iOS, and macOS using a single project and shared UI and logic code. This preview focused on establishing the core architecture and enabling basic UI development capabilities.
- ➔ **.NET MAUI Preview 4** (April 2021): This release brought significant improvements to the developer experience, including enhanced XAML Hot Reload functionality and better integration with Visual Studio and Visual Studio Code. It also introduced support for additional controls and layouts, improving the overall feature set for building rich user interfaces.
- ➔ **.NET MAUI Preview 7** (August 2021): Preview 7 focused on performance optimizations, particularly in areas such as startup time and rendering. It also introduced support for additional platform-specific features, allowing developers to leverage native capabilities more effectively. This release aimed to improve the overall app quality and user experience.
- ➔ **.NET MAUI GA** (November 2021): The General Availability (GA) release of .NET MAUI marked a major milestone, signifying its production-ready status. This version included stability improvements, bug fixes, and enhanced tooling support. It also introduced new features like accessibility improvements and better support for desktop platforms.
- ➔ **.NET MAUI 6.0.200** (March 2022): This update brought further performance enhancements, particularly in areas like memory usage and startup time. It also introduced new controls and layouts, expanding the UI toolkit's capabilities. Additionally, this release improved the integration with other .NET technologies, such as Blazor and gRPC.

→ **.NET MAUI 7.0** (November 2022): The .NET 7 release of .NET MAUI focused on improving developer productivity and app quality. It introduced new features like improved XAML compilation, better support for desktop platforms, and enhanced accessibility features. This release also included performance optimizations and bug fixes based on feedback from the developer community.

6.2.1.3. Open-Source, but Not Libre

.NET MAUI is an open-source project, which means that its source code is publicly available and can be inspected, modified, and distributed by anyone. The .NET MAUI project is hosted on GitHub as seen on figure 54, and the community is encouraged to contribute to its development through code contributions, bug reports, and feature requests.

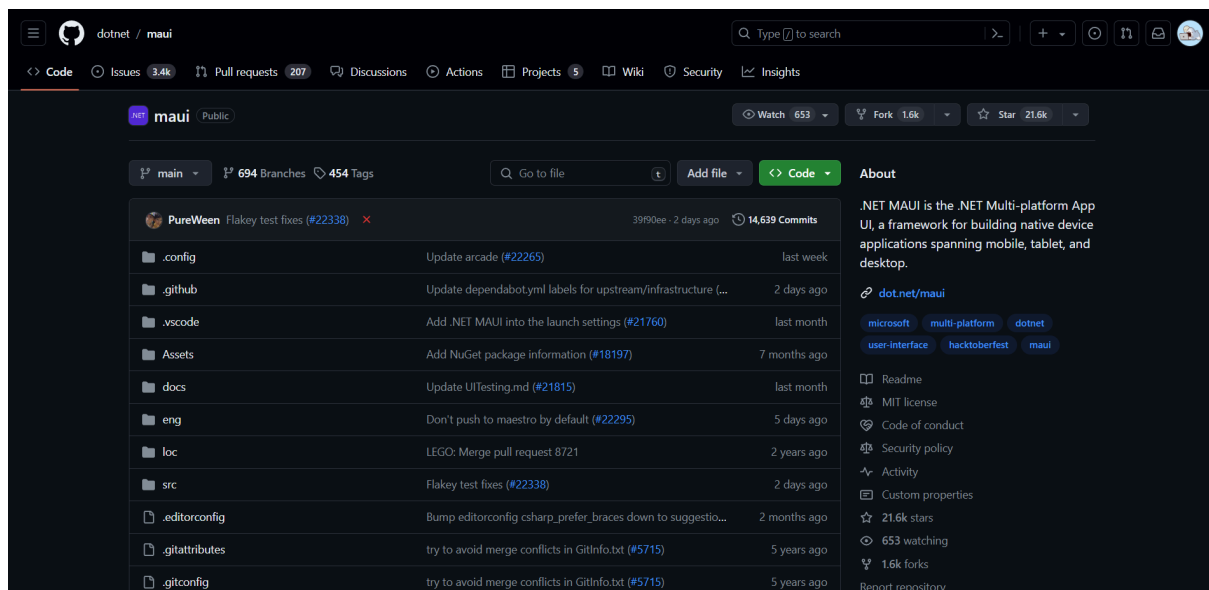


Figure 54. Github repository of .NET MAUI (source: GitHub/dotnet/maui [72])

However, despite being open-source, .NET MAUI is licensed under the MIT License, which is a permissive license that allows for commercial use, modification, distribution, and private use.

While the MIT License grants certain freedoms, it does not meet the criteria for libre software as defined by the Free Software Foundation (FSF). The FSF defines libre software as software that respects the user's freedom and provides the following essential freedoms:

- “The freedom to run the program as you wish, for any purpose (freedom 0).
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help others (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.”[73]

While the MIT License allows for the first three freedoms, *it does not explicitly grant the fourth freedom, which is the freedom to distribute modified versions of the software with access to the source code being a precondition for this*. This means that while developers can modify .NET MAUI for their own use, they may not be able to distribute those modifications without additional permissions or licensing agreements as explained on figure 55.

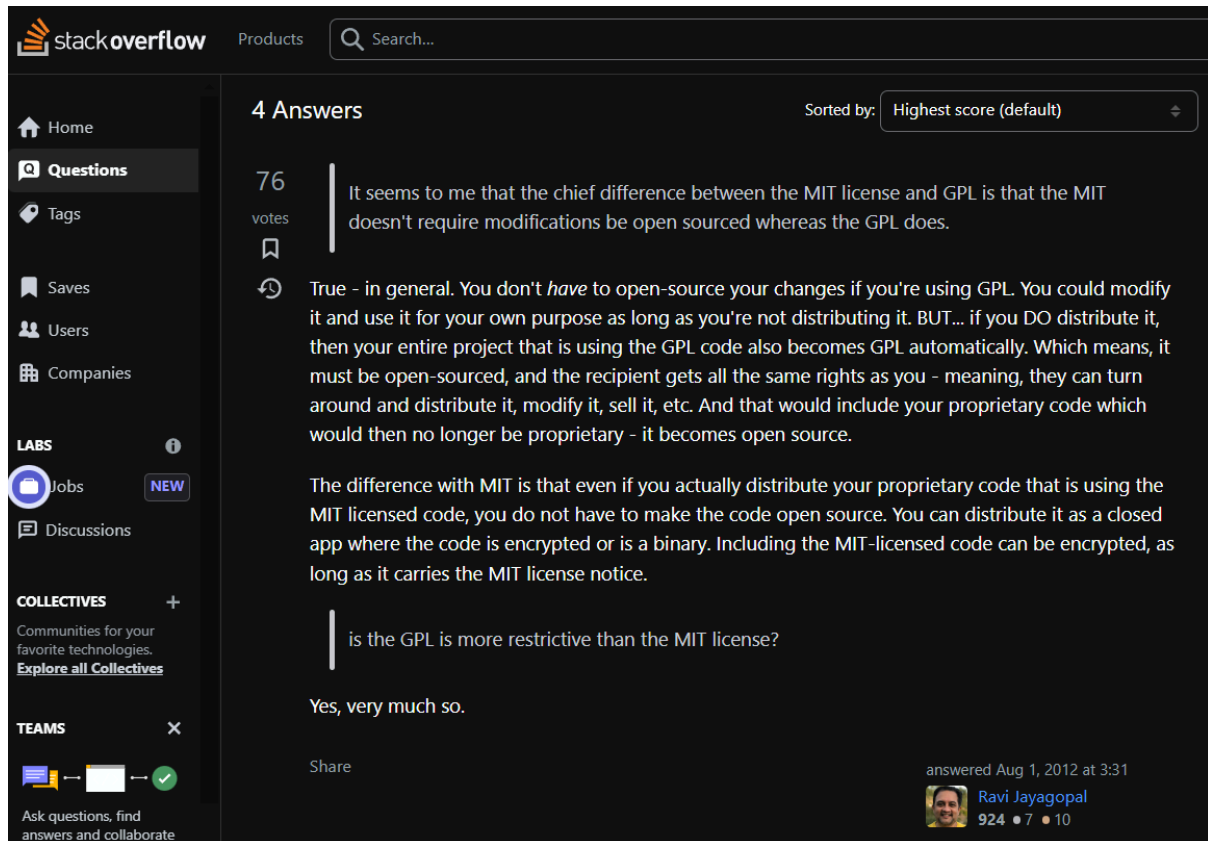


Figure 55. Concise explanation of the chief difference between MIT and GPLv2 licenses (source: StackOverflow [73])

Furthermore, .NET MAUI is part of Microsoft's broader .NET ecosystem, which includes proprietary components and technologies. While .NET MAUI itself is open-source, its integration with other Microsoft products and services may introduce dependencies on proprietary software or services, potentially limiting the user's freedom in certain scenarios.

6.2.1.4. Cross-Platform Architecture of .NET MAUI

.NET MAUI is designed as a unified solution for developing cross-platform applications, allowing developers to create apps that can run on multiple platforms, including Windows, Android, iOS, and macOS, using a single codebase. The key to this cross-platform capability lies in its architecture and the way it leverages platform-specific components while sharing a common base.

6.2.1.4.1. .NET Base Class Library (BCL)

At the core of .NET MAUI is the .NET Base Class Library (BCL), which is shared across all supported platforms. The BCL provides a common foundation for .NET applications, abstracting away platform-specific details and enabling developers to write code that can be shared across different operating systems.

6.2.1.4.2. Platform-Specific UI Frameworks

While the BCL provides a shared base, .NET MAUI leverages platform-specific UI frameworks to render the user interface natively on each platform while having a single framework for building the UIs for mobile and desktop apps that a developer can build with. For example:

1. **Windows:** .NET MAUI uses the Windows UI 3 (WinUI 3) library to render the UI on Windows desktop applications.
2. **Android:** On Android, .NET MAUI utilizes the Android UI toolkit and rendering engine.
3. **iOS:** For iOS applications, .NET MAUI integrates with the UIKit framework to render native iOS user interfaces.
4. **macOS:** On macOS, .NET MAUI employs Mac Catalyst, an Apple technology that ports iOS apps to the desktop while enhancing them with additional AppKit and platform APIs.

By leveraging these platform-specific UI frameworks, .NET MAUI ensures that applications have a truly native look and feel, while still allowing developers to share a significant portion of their codebase across platforms.

6.2.1.4.3. Native App Packages

As seen on figure 56, in a .NET MAUI app, the developer writes code that primarily interacts with the .NET MAUI API (1). .NET MAUI then directly consumes the native platform APIs (3). In addition, app code may directly exercise platform APIs (2), if required. When building applications with .NET MAUI, the C# code is compiled into platform-specific native app packages:

1. **Android:** C# code is translated into an intermediate language (IL), which is then JIT-compiled into native ARM assembly code at runtime.
2. **iOS:** C# code is directly compiled into native ARM assembly code.
3. **macOS:** .NET MAUI apps leverage Mac Catalyst to port the iOS app to the desktop, enhancing it with additional AppKit and platform APIs.
4. **Windows:** Native Windows desktop applications are created using the WinUI 3 library.

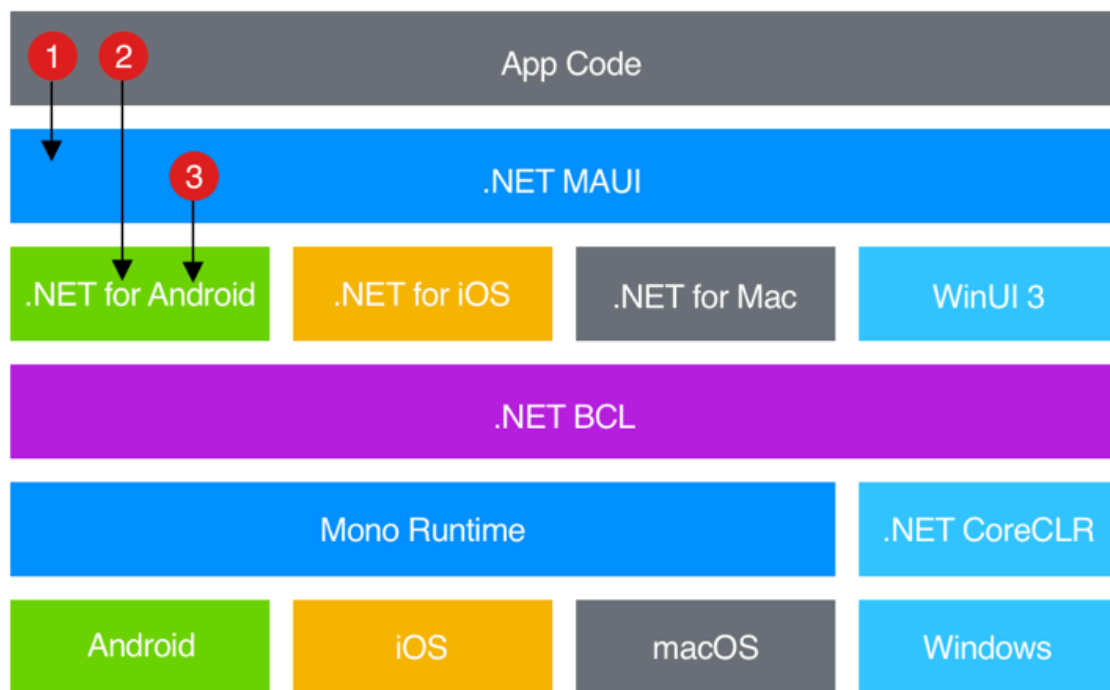


Figure 56. .NET MAUI architecture diagram (source: [74])

This approach ensures that .NET MAUI applications run as truly native apps on each platform, providing optimal performance and a seamless user experience.

6.2.1.4.4. C# language

C# is the primary programming language used for developing applications with .NET MAUI. As a core component of the .NET ecosystem, C# plays a crucial role in building cross-platform applications with MAUI.

.NET MAUI is built on top of the .NET runtime, which provides a robust and efficient execution environment for C# code. This combination of C# and .NET allows developers to write high-performance, scalable, and secure applications that can run on multiple platforms, including Windows, Android, iOS, and macOS.

6.2.1.4.5. XAML

XAML (Extensible Application Markup Language) plays a crucial role in building user interfaces for .NET MAUI applications. While C# is used for writing the application logic and business rules, XAML is used for defining user interfaces. XAML provides a declarative way to describe the layout and appearance of UI elements, separating the presentation layer from the underlying code.

XAML allows developers to define user interfaces in .NET MAUI apps using markup rather than code. XAML is not required in a .NET MAUI app, but it is the recommended approach to developing UI because it's often more succinct, more visually coherent, and has tooling support.

C# and XAML work together seamlessly in .NET MAUI, allowing developers to bind UI elements to C# code and handle user interactions and events. This separation of concerns between UI and logic promotes code organization, maintainability, and testability.

6.2.1.4.5.1. XAML Hot Reload and Live Visual Tree

.NET MAUI includes powerful tooling features that enhance the development experience when working with XAML. XAML Hot Reload enables developers to see changes made to XAML files in real-time, without the need for a full application restart. This feature significantly improves the iterative development process and productivity. Additionally, the Live Visual Tree and Live Property Explorer tools provide a visual representation of the UI hierarchy and allow developers to inspect and modify properties of UI elements at runtime, further streamlining the debugging and development process.

6.2.2. unDraw

6.2.2.1. Introduction & history

“A constantly updated collection of MIT licensed illustrations you can use on your designs, websites, apps and any project really!” [75]

unDraw is an open-source design project that offers a vast collection of high-quality illustrations for any idea you can imagine. Launched in 2017 by Katerina Limpitsouni with the goal of contributing beautiful design assets to the open-source community, unDraw has grown into a valuable resource for designers, developers, and content creators worldwide. The platform provides a library of customizable illustrations that can be seamlessly integrated into websites, products, and applications as seen on figure 57.

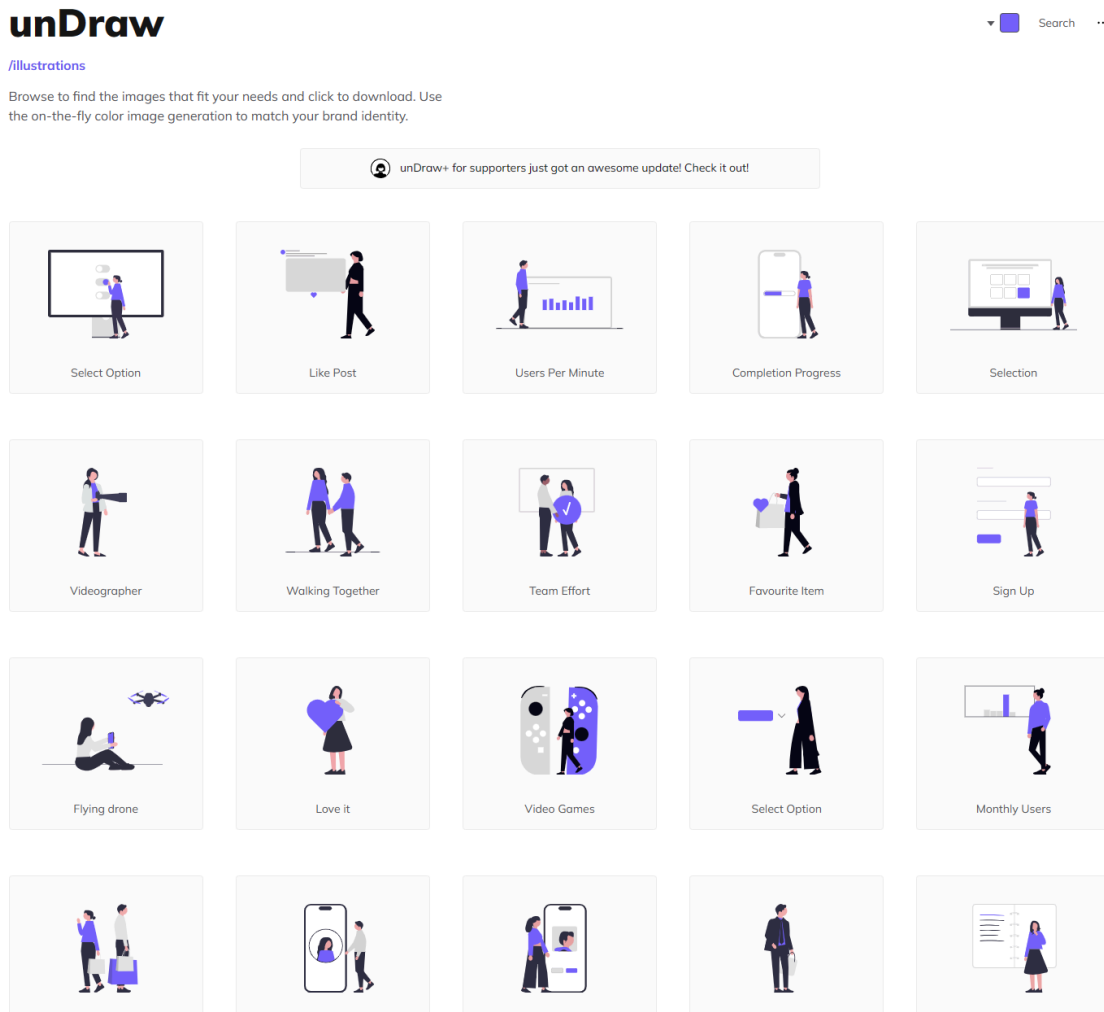


Figure 57. Illustrations browsing page of unDraw (source: unDraw [76])

6.2.2.2. Open-Source, but Not Libre

unDraw is an open-source project, but it is not considered libre or free software as defined by the principles of software freedom outlined by the Free Software Foundation (FSF) because as seen on figure 58 and quoted from the unDraw’s license subpage: “[...] *This license does not include the right to compile assets, vectors or images from unDraw to replicate a similar or competing service, in any form or distribute the assets in packs or otherwise. This extends to automated and non-automated ways to link, embed, scrape, search or download the assets included on the website without our consent.*” [77]

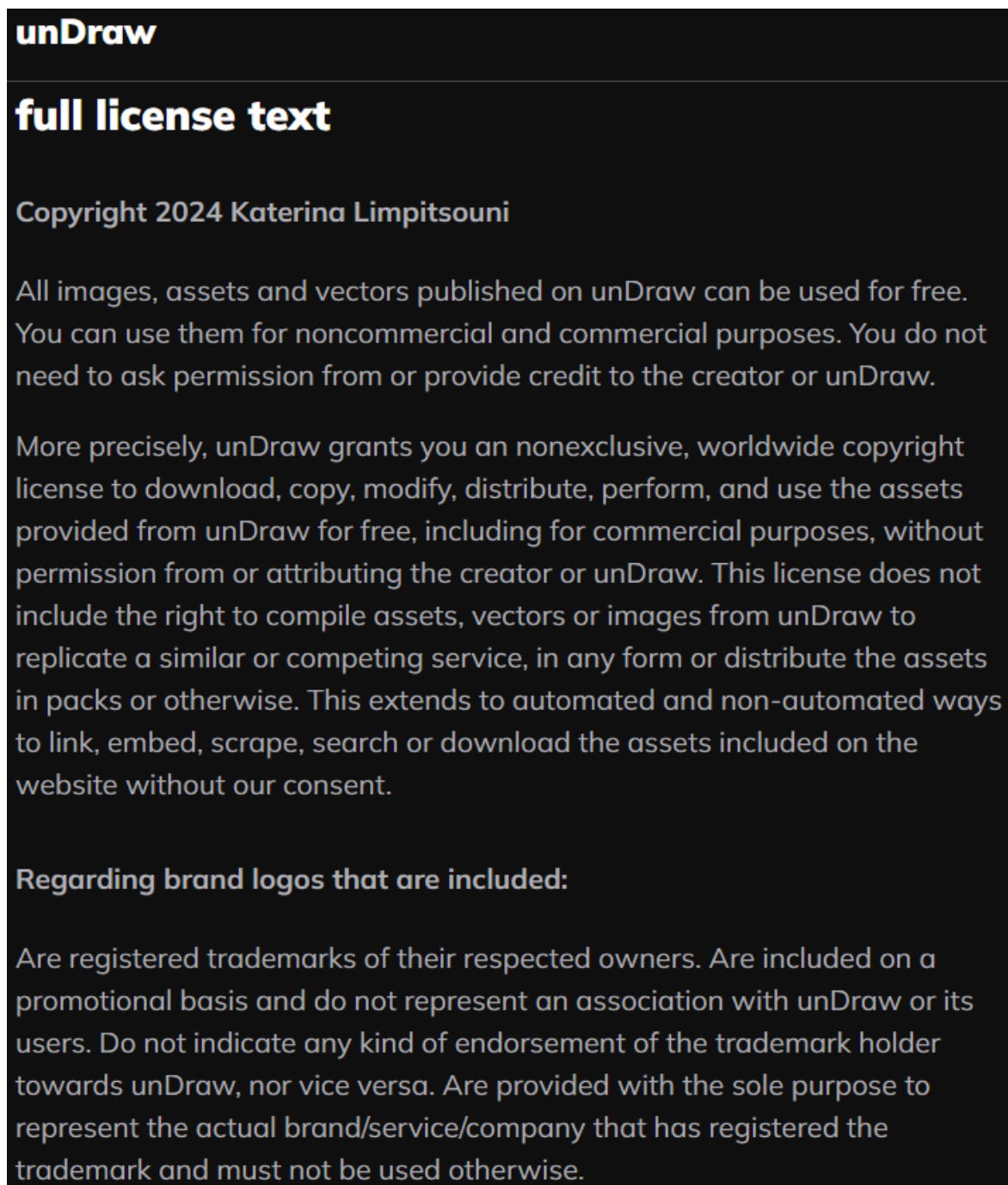


Figure 58. Full license text of unDraw (source: [77])

6.2.3. Iconoir icons

Some of the icons used for the application come from Iconoir [78]. Iconoir is an open-source library with 1500+ unique SVG icons, designed on a 24x24 pixels grid as seen on figure 59.

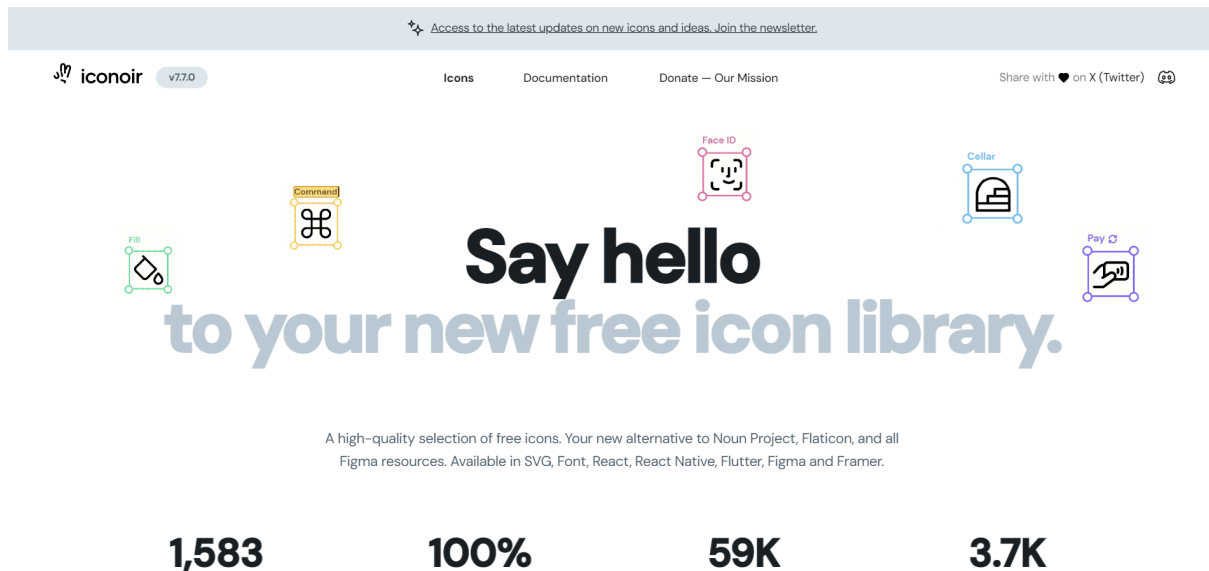


Figure 59. Iconoir homepage (source: Iconoir [77])

6.2.3.1. Open-Source, but Not Libre

Iconoir is an open-source project, but it is not considered libre or free software as defined by the principles of software freedom outlined by the Free Software Foundation (FSF) because as seen on figure 60 it is licensed under the MIT license which is not copyleft.

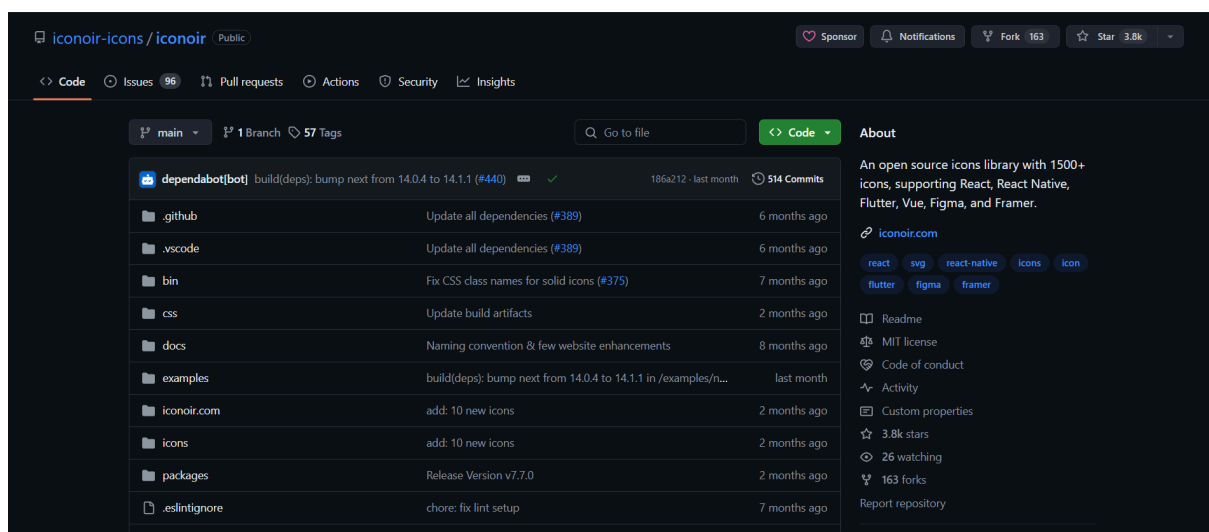


Figure 60. Github repository of Iconoir (source: [79])

6.3. Proprietary software

Proprietary or commercial software is a type of computer software that belongs to and is distributed by a specific company or organization. It cannot be freely modified, redistributed, or accessed like open-source software. Instead, it is distributed under a license that restricts its usage and often requires a fee or subscription to be used.

Proprietary software is also known as closed-source software due to its closed nature, as the source code is kept confidential and proprietary to the company that created it. This means that end-users cannot access the underlying code and cannot modify or customize the software according to their specific needs. The development and maintenance of proprietary software are solely controlled by software companies, giving them complete authority over its features, updates, and distribution. [80]

6.3.1. Microsoft Visual Studio IDE

6.3.1.1. Introduction

Microsoft Visual Studio as seen on figure 61 is a powerful and comprehensive integrated development environment (IDE) primarily used for developing applications targeting the Microsoft ecosystem, including Windows, web, cloud, and mobile platforms. It is a core component of the .NET ecosystem and is widely adopted by developers working with C#, Visual Basic, F#, and C++ programming languages.

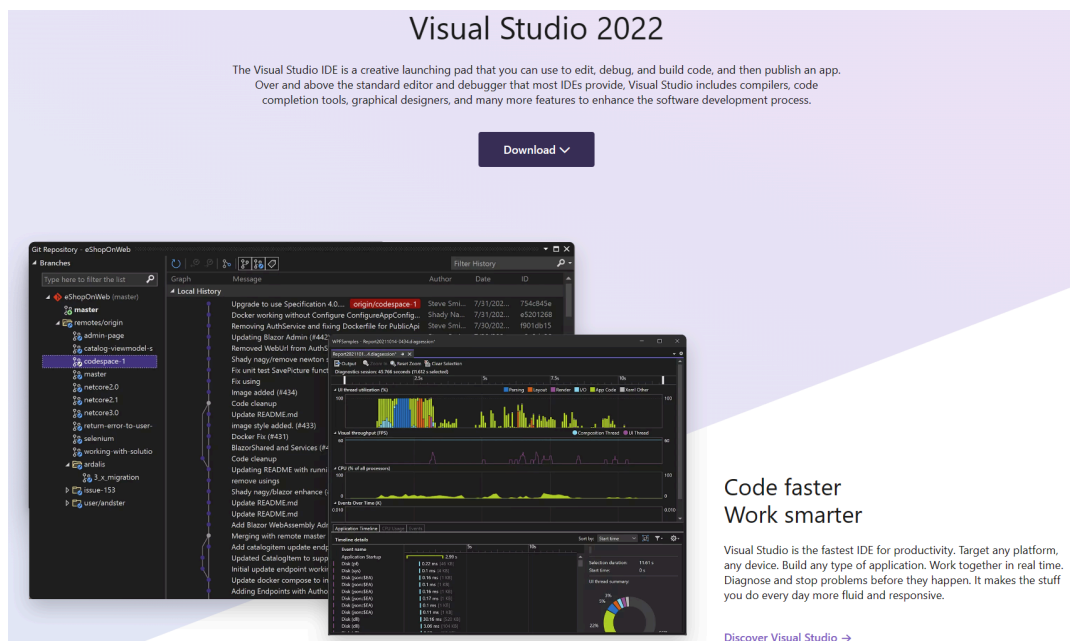


Figure 61. Homepage of Visual Studio website (source: Visual Studio webpage [70])

6.3.1.2. History

Microsoft Visual Studio has a rich history that spans over two decades, evolving from a collection of separate development tools into a comprehensive and integrated development environment (IDE).

- **Visual Studio 97** - Marked the beginning of the Visual Studio product line, bringing together various development tools into a single integrated development environment (IDE)
- **Visual Studio .NET (2002)** - Introduced support for the .NET Framework and managed code development using languages like C#, Visual Basic, and Managed C++
- **Visual Studio .NET 2003** - Improved performance and added support for mobile development; introduced the Visual Studio .NET Compact Framework.
- **Visual Studio 2005** - Introduced support for developing applications using Windows Presentation Foundation (WPF) and the .NET Framework 2.0
- **Visual Studio 2008** - Added support for LINQ (Language Integrated Query); introduced the Visual Studio Gallery for extensions and add-ons; enabled multi-targeting for different .NET Framework versions.
- **Visual Studio 2010** - Improved performance and productivity features; introduced support for the F# programming language.
- **Visual Studio 2012** - Brought enhancements in performance, debugging, and diagnostics; introduced asynchronous programming using `async` and `await` keywords.
- **Visual Studio 2013** - Focused on improving developer productivity and team collaboration features; introduced support for Git version control system.
- **Visual Studio 2015** - Improved performance and added support for cross-platform mobile development with Xamarin; introduced support for open-source development and cross-platform .NET Core.
- **Visual Studio 2017** - Introduced a new lightweight and modular installation experience; improved support for cloud development and Azure services.
- **Visual Studio 2019** - Focused on improving performance, reliability, and productivity; introduced a new start window and enhanced search capabilities.
- **Visual Studio 2022 (Preview 2021)** - Announced in April 2021; first version to run as a 64-bit process, allowing access to more than 4 GB of memory; preview releases throughout 2021
- **Visual Studio 2022 (General Availability November 8, 2021)** - Official release of Visual Studio 2022; supports Windows 10, Windows Server 2016 or later, and Windows Server 2022
- **Visual Studio 2022 17.3 (August 9, 2022)** - Added support for targeting the .NET Framework 4.8.1

6.3.1.3. Proprietary nature

The core Visual Studio IDE itself is proprietary software owned and licensed by Microsoft. The source code for the main Visual Studio IDE is not open-source or publicly available. Visual Studio includes some open-source components and integrates with open-source frameworks and tools, such as .NET Core, which is open-source. However, the Visual Studio product as a whole is not open-source due to the inclusion of proprietary components and closed-source extensions.

6.3.1.4. Core features of Visual Studio

While Microsoft Visual Studio offers a wide range of features and capabilities, three key features stand out as the core strengths of this integrated development environment.

6.3.1.4.1. Integrated Development Environment

Visual Studio's primary strength lies in its ability to provide a comprehensive and integrated development environment. It brings together various tools and features essential for software development, streamlining the entire development lifecycle within a single interface. This integration includes:

- ➔ **Code Editor:** Visual Studio's code editor is packed with advanced features like IntelliSense, code refactoring, and syntax highlighting, making it easier to write, navigate, and maintain code.
- ➔ **Debugging Tools:** Visual Studio offers powerful debugging tools that allow developers to diagnose and fix issues in their code with ease. Features like breakpoints, watch windows, and call stacks help developers understand the execution flow and identify problems quickly.
- ➔ **Build Automation:** Visual Studio integrates with build automation tools, enabling developers to automate the process of compiling, packaging, and deploying their applications, saving time and reducing errors.
- ➔ **Source Control Integration:** Visual Studio seamlessly integrates with popular source control systems like Git and Team Foundation Version Control (TFVC), enabling developers to collaborate effectively and manage code changes efficiently.

6.3.1.4.2. Cross-Platform Development

Visual Studio has evolved to support cross-platform development, allowing developers to build applications for various platforms and devices using a single codebase. This cross-platform support includes:

- **Windows Applications:** Visual Studio has long been the go-to IDE for developing Windows desktop applications using technologies like Windows Forms, WPF, and UWP.
- **Web Applications:** With support for ASP.NET, ASP.NET Core, and modern web technologies like JavaScript and TypeScript, Visual Studio enables developers to build robust and scalable web applications.
- **Mobile Applications:** Through integration with Xamarin and .NET MAUI, Visual Studio empowers developers to create native and cross-platform mobile applications for Android, iOS, and Windows devices.
- **Cloud Applications:** Visual Studio provides seamless integration with Microsoft Azure, enabling developers to build, deploy, and manage cloud-based applications and services with ease.

6.3.1.4.3. Extensibility and Customization

One of the key strengths of Visual Studio is its extensibility and customization capabilities. Visual Studio offers a vast ecosystem of extensions and add-ons, as seen on figure 62, developed by Microsoft, third-party vendors, and the community. These extensions can enhance the IDE's functionality, add support for new programming languages, frameworks, and tools, and even customize the user interface to suit individual preferences.

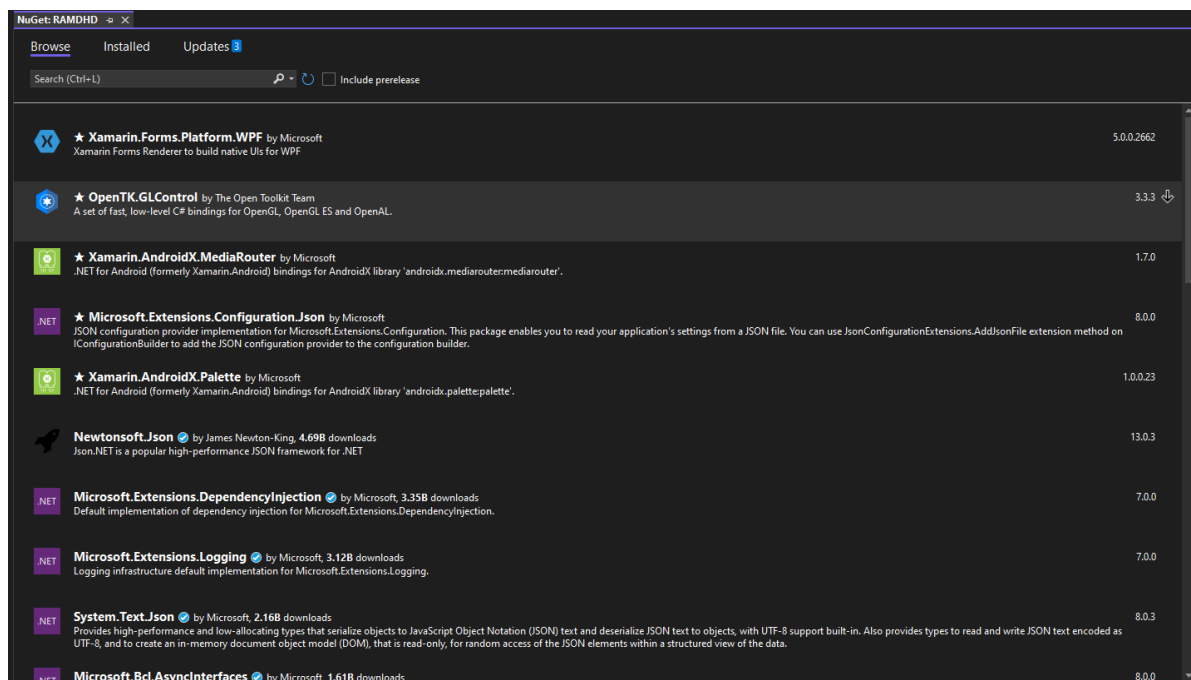


Figure 62. NuGet Package Manager window inside MS Visual Studio IDE (tool: Visual Studio IDE [70], source: self)

7. Implementation of the project

This chapter is dedicated to describing the process of implementing the application, providing the source code behind the final version of the project. The chapter is divided into subchapters according to the tasks that were given to users during the application testing phase:

1. Creating custom routine
2. Completing the custom routine (displaying and checking all the routine steps)
3. Creating custom note (reminding about deleting the routine)
4. Visiting one of the forums
5. Deleting created earlier custom routine

However, before describing the implementation behind mentioned tasks it is important to understand the fundamental navigation within the app. Hence the first subchapters will explain:

1. Navigating to the Attention module
2. Navigating between modules

7.1. Navigating to the Attention module

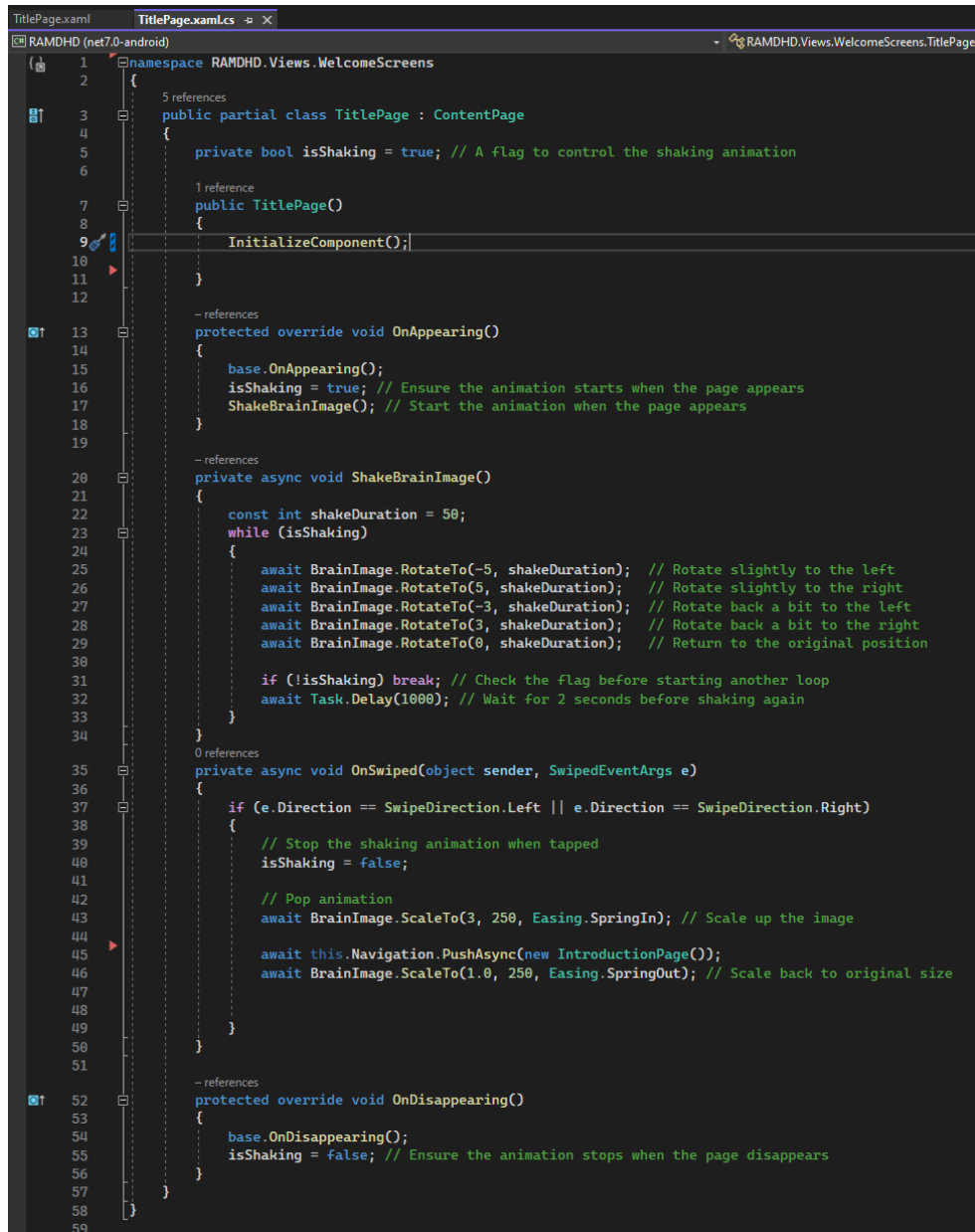
The Title screen consists of the “CutCatADHD” title and a “shaking” brain image that needs to be “cut” by the user (perform horizontal gesture) in order to navigate to the next screen. Figure 63 presents the XAML frontend code of the Title screen.



Figure 63. Source code of the TitlePage.xaml (tool: Visual Studio IDE [70], source: self)

The animation of the “shaking” brain, gestures recognition and navigation is handled in the code-behind as seen on figure 64. The .xaml.cs file is a code-behind file that is associated with a corresponding XAML file in a Xamarin.Forms application. The XAML file defines the user interface and layout of the application, while the .xaml.cs file contains the code that implements the logic and behavior of the application.

As specified in the OnSwiped function, when the user “cuts” the brain horizontally the animation pops (by scaling up) and navigates to the Introduction screen.



```

1 namespace RAMDHD.Views.WelcomeScreens
2 {
3     5 references
4     public partial class TitlePage : ContentPage
5     {
6         private bool isShaking = true; // A flag to control the shaking animation
7
8         1 reference
9         public TitlePage()
10        {
11            InitializeComponent();
12        }
13
14        - references
15        protected override void OnAppearing()
16        {
17            base.OnAppearing();
18            isShaking = true; // Ensure the animation starts when the page appears
19            ShakeBrainImage(); // Start the animation when the page appears
20        }
21
22        - references
23        private async void ShakeBrainImage()
24        {
25            const int shakeDuration = 50;
26            while (isShaking)
27            {
28                await BrainImage.RotateTo(-5, shakeDuration); // Rotate slightly to the left
29                await BrainImage.RotateTo(5, shakeDuration); // Rotate slightly to the right
30                await BrainImage.RotateTo(-3, shakeDuration); // Rotate back a bit to the left
31                await BrainImage.RotateTo(3, shakeDuration); // Rotate back a bit to the right
32                await BrainImage.RotateTo(0, shakeDuration); // Return to the original position
33
34                if (!isShaking) break; // Check the flag before starting another loop
35                await Task.Delay(1000); // Wait for 2 seconds before shaking again
36            }
37        }
38
39        0 references
40        private async void OnSwiped(object sender, SwipedEventArgs e)
41        {
42            if (e.Direction == SwipeDirection.Left || e.Direction == SwipeDirection.Right)
43            {
44                // Stop the shaking animation when tapped
45                isShaking = false;
46
47                // Pop animation
48                await BrainImage.ScaleTo(3, 250, Easing.SpringIn); // Scale up the image
49
50                await this.Navigation.PushAsync(new IntroductionPage());
51                await BrainImage.ScaleTo(1.0, 250, Easing.SpringOut); // Scale back to original size
52            }
53        }
54
55        - references
56        protected override void OnDisappearing()
57        {
58            base.OnDisappearing();
59            isShaking = false; // Ensure the animation stops when the page disappears
60        }
61    }
62 }

```

Figure 64. Source code of the TitlePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

On the Introduction screen there are several elements. The shaking brain image as presented before, 3 hidden images representing the 3 ADHD types and navigation elements as seen on figure 65.

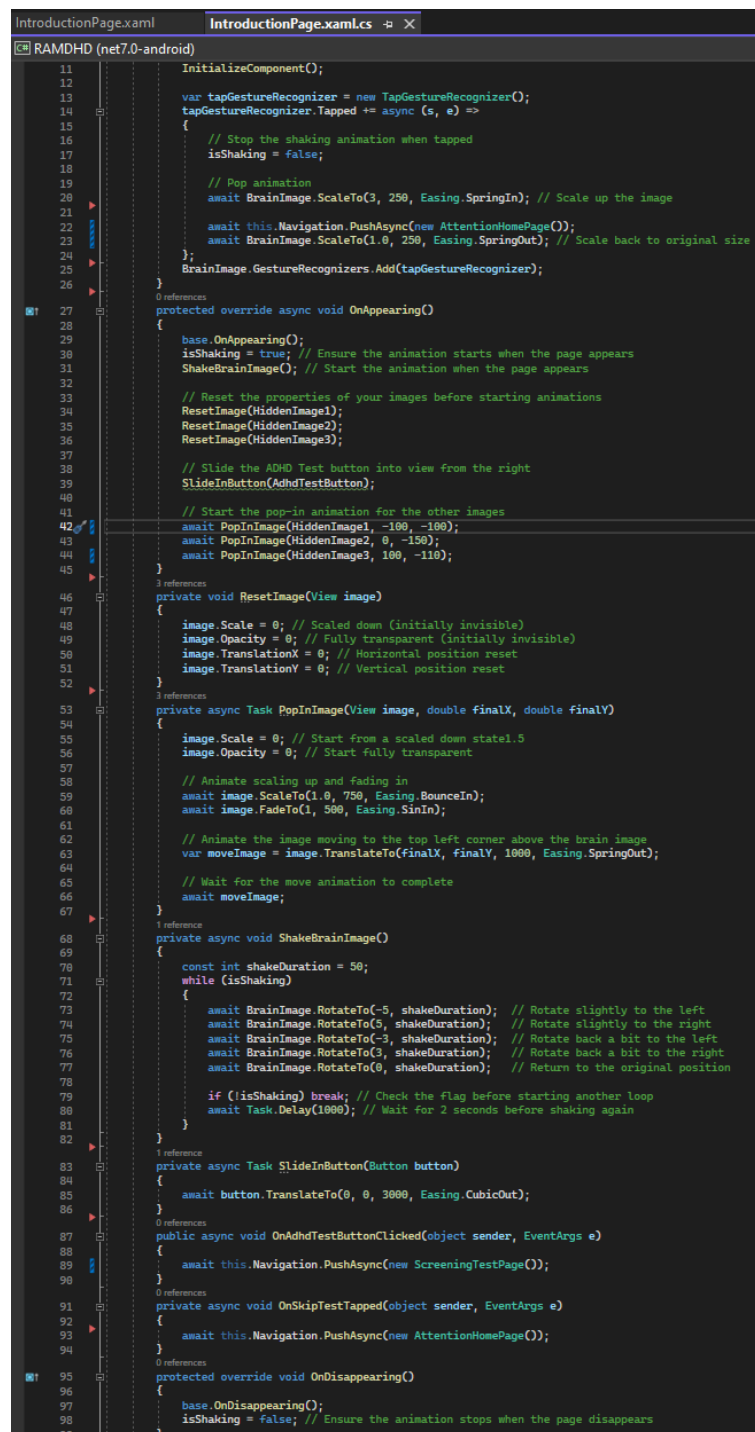
```

IntroductionPage.xaml IntroductionPage.xaml.cs
RAMDHD (net7.0-android)
1 <ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
2   xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
3   xmlns:material="http://schemas.enisn-projects.io/dotnet/maui/uraniumui/material"
4   x:Class="RAMDHD.Views.WelcomeScreens.IntroductionPage"
5   BackgroundColor="{StaticResource White}">
6
7   <!-- Use a Grid instead of VerticalStackLayout for more precise control over layout -->
8   <Grid VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand">
9     <Grid.RowDefinitions>
10      <RowDefinition Height="100"/>
11      <!-- For the title -->
12      <RowDefinition Height="*" />
13      <!-- For the image -->
14      <RowDefinition Height="Auto"/>
15      <!-- For the bottom spacer -->
16      <RowDefinition Height="Auto"/>
17    </Grid.RowDefinitions>
18
19    <!-- Title Text -->
20    <Label Grid.Row="0"
21      Text="Do you have ADHD?"
22      FontSize="31"
23      FontFamily="ReemKufiFunBold"
24      TextColor="{StaticResource Tertiary}"
25      HorizontalOptions="Center"
26      VerticalOptions="End" />
27
28    <!-- Brain Image centered in the grid -->
29    <Grid Grid.Row="1" HorizontalOptions="Center" VerticalOptions="Center">
30
31      <!-- Brain Image centered in the grid -->
32      <Image x:Name="BrainImage"
33        Source="brain.svg"
34        WidthRequest="200"
35        HeightRequest="200"
36        HorizontalOptions="Center"
37        VerticalOptions="Center">
38        <Image.GestureRecognizers>
39          <TapGestureRecognizer
40            NumberOfTapsRequired="1" />
41        </Image.GestureRecognizers>
42      </Image>
43
44      <Image x:Name="HiddenImage1"
45        Source="undraw_forgot_password_re_hxmm.svg"
46        WidthRequest="100"
47        HeightRequest="100"
48        Opacity="0" />
49
50      <Image x:Name="HiddenImage2"
51        Source="undraw_jogging_re_k28i.svg"
52        WidthRequest="100"
53        HeightRequest="100"
54        Opacity="0" />
55
56      <Image x:Name="HiddenImage3"
57        Source="undraw_decide_re_ixfw.svg"
58        WidthRequest="100"
59        HeightRequest="100"
60        Opacity="0" />
61    </Grid>
62
63    <!-- Button for ADHD Screening Test -->
64    <Button
65      Grid.Row="3"
66      x:Name="AdhdTestButton"
67      Text="ADHD SCREENING TEST"
68      TextColor="{StaticResource SharcBlack}"
69      FontFamily="ReemKufiFunRegular"
70      BackgroundColor="{StaticResource Tertiary}"
71      BorderWidth="1"
72      BorderColor="{StaticResource Secondary}"
73      HorizontalOptions="Center"
74      Margin="0,0,0,100"
75      VerticalOptions="End"
76      TranslationX="1000"
77      Clicked="OnAdhdTestButtonClicked"
78    />
79
80    <!-- Skip the Test Label -->
81    <Label Grid.Row="3"
82      Text="Skip the test"
83      Margin="0,60,0,0"
84      HorizontalOptions="Center"
85      VerticalOptions="Start"
86      TextColor="{StaticResource SharcBlack}"
87      FontFamily="ReemKufiFunRegular"
88      FontAttributes="Italic">
89      <Label.GestureRecognizers>
90        <TapGestureRecognizer Tapped="OnSkipTestTapped" />
91      </Label.GestureRecognizers>
92    </Label>
93  </Grid>
94 </ContentPage>
95

```

Figure 65. Source code of the IntroductionPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

The code-behind handles the animation of the brain and images, and the navigation as seen on figure 66. On initialization the gesture recognition is added to the brain image. If the brain is tapped the user gets navigated to the Attention module (acting as the main screen). When the screen gets loaded the brain starts shaking, the 3 images “pop out” from behind the brain one by one and the navigation button slides in. To navigate to the screening test user clicks the AdhdTestButton. To navigate to the Attention module user clicks either on the brain or the “Skip the test” label.



```

IntroductionPage.xaml IntroductionPage.xaml.cs
RAMDHD (net7.0-android)
11 InitializeComponent();
12
13 var tapGestureRecognizer = new TapGestureRecognizer();
14 tapGestureRecognizer.Tapped += async (s, e) =>
15 {
16     // Stop the shaking animation when tapped
17     isShaking = false;
18
19     // Pop animation
20     await BrainImage.ScaleTo(3, 250, Easing.SpringIn); // Scale up the image
21
22     await this.Navigation.PushAsync(new AttentionHomePage());
23     await BrainImage.ScaleTo(1.0, 250, Easing.SpringOut); // Scale back to original size
24 };
25 BrainImage.GestureRecognizers.Add(tapGestureRecognizer);
26
27 0 references
28 protected override async void OnAppearing()
29 {
30     base.OnAppearing();
31     isShaking = true; // Ensure the animation starts when the page appears
32     ShakeBrainImage(); // Start the animation when the page appears
33
34     // Reset the properties of your images before starting animations
35     ResetImage(HiddenImage1);
36     ResetImage(HiddenImage2);
37     ResetImage(HiddenImage3);
38
39     // Slide the ADHD Test button into view from the right
40     SlideInButton(AdhdTestButton);
41
42     // Start the pop-in animation for the other images
43     await PopInImage(HiddenImage1, -100, -100);
44     await PopInImage(HiddenImage2, 0, -150);
45     await PopInImage(HiddenImage3, 100, -110);
46 }
47 3 references
48 private void ResetImage(View image)
49 {
50     image.Scale = 0; // Scaled down (initially invisible)
51     image.Opacity = 0; // Fully transparent (initially invisible)
52     image.TranslationX = 0; // Horizontal position reset
53     image.TranslationY = 0; // Vertical position reset
54 }
55 3 references
56 private async Task PopInImage(View image, double finalX, double finalY)
57 {
58     image.Scale = 0; // Start from a scaled down state 1.5
59     image.Opacity = 0; // Start fully transparent
60
61     // Animate scaling up and fading in
62     await image.ScaleTo(1.0, 750, Easing.BounceIn);
63     await image.FadeTo(1, 500, Easing.SinIn);
64
65     // Animate the image moving to the top left corner above the brain image
66     var moveImage = image.TranslateTo(finalX, finalY, 1000, Easing.SpringOut);
67     // Wait for the move animation to complete
68     await moveImage;
69 }
70 1 reference
71 private async void ShakeBrainImage()
72 {
73     const int shakeDuration = 50;
74     while (isShaking)
75     {
76         await BrainImage.RotateTo(-5, shakeDuration); // Rotate slightly to the left
77         await BrainImage.RotateTo(5, shakeDuration); // Rotate slightly to the right
78         await BrainImage.RotateTo(-3, shakeDuration); // Rotate back a bit to the left
79         await BrainImage.RotateTo(3, shakeDuration); // Rotate back a bit to the right
80         await BrainImage.RotateTo(0, shakeDuration); // Return to the original position
81
82         if (!isShaking) break; // Check the flag before starting another loop
83         await Task.Delay(1000); // Wait for 2 seconds before shaking again
84     }
85 }
86 1 reference
87 private async Task SlideInButton(Button button)
88 {
89     await button.TranslateTo(0, 0, 3000, Easing.CubicOut);
90 }
91 0 references
92 public async void OnAdhdTestButtonClicked(object sender, EventArgs e)
93 {
94     await this.Navigation.PushAsync(new ScreeningTestPage());
95 }
96 0 references
97 private async void OnSkipTestTapped(object sender, EventArgs e)
98 {
99     await this.Navigation.PushAsync(new AttentionHomePage());
100 }
101 0 references
102 protected override void OnDisappearing()
103 {
104     base.OnDisappearing();
105     isShaking = false; // Ensure the animation stops when the page disappears
106 }

```

Figure 66. Source code of the IntroductionPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

The Attention module acts as a starting page. Each of the 4 modules (except the Graph task module) are structured very similarly. They contain from 2 to 4 navigation panels that include the main functionalities. As seen on figure 67 the Attention module contains 4 panels - Timer, Notes, Knowledge base and Challenges.

```

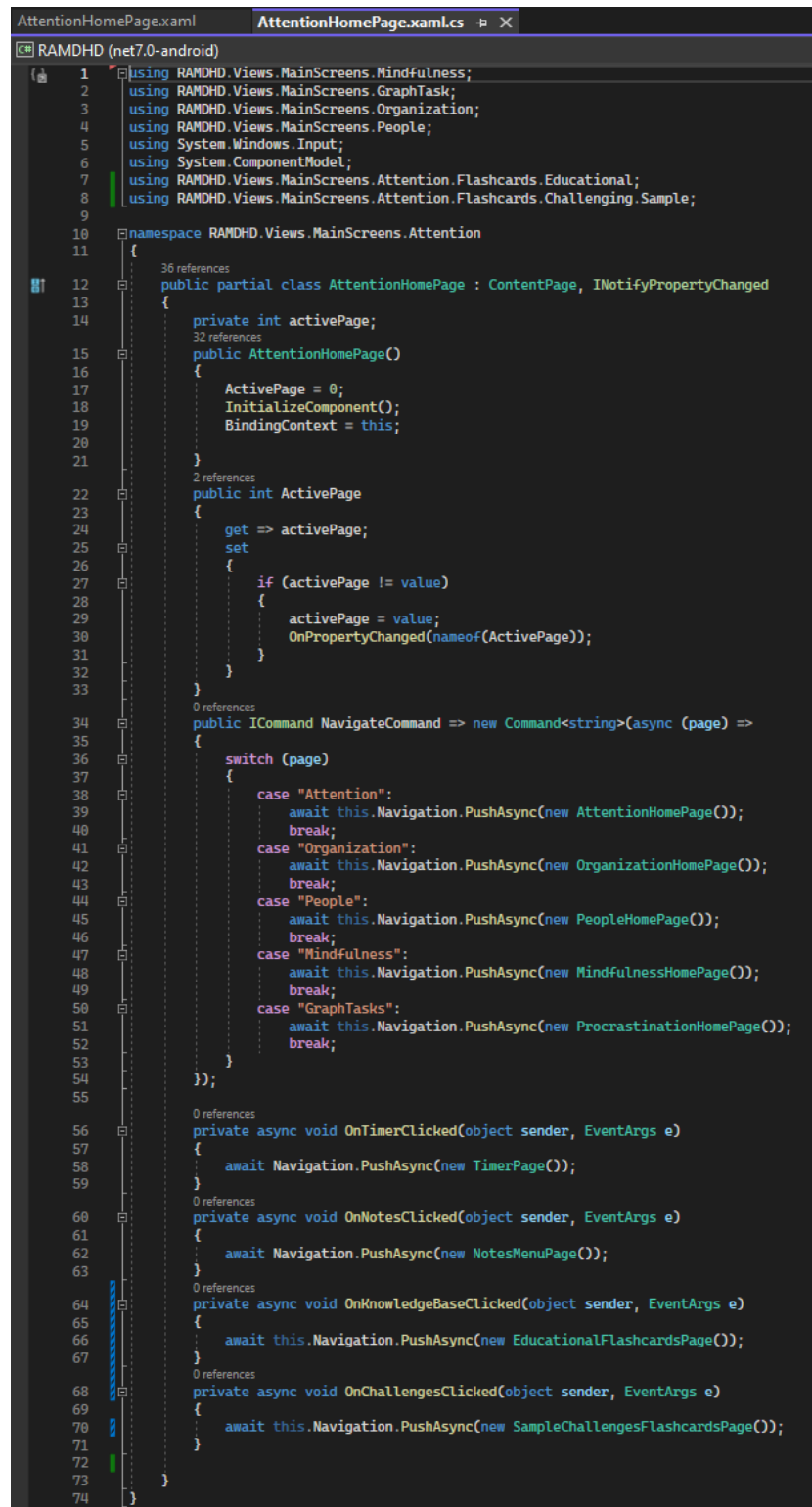
AttentionHomePage.xaml  AttentionHomePage.xaml.cs
RAMDHD (net7.0-android)
4      x:Class="RAMDHD.Views.MainScreens.Attention.AttentionHomePage"
5      Title="Attention"
6      BackgroundColor="{StaticResource White}"
7      <AbsoluteLayout HorizontalOptions="FillAndExpand" VerticalOptions="FillAndExpand">
8
9          <!-- TIMER panel -->
10         <StackLayout WidthRequest="150" HeightRequest="150"
11             AbsoluteLayout.LayoutBounds="0.25, 0.25, -0.5, -1"
12             AbsoluteLayout.LayoutFlags="PositionProportional">
13             <StackLayout.GestureRecognizers>
14                 <TapGestureRecognizer Tapped="OnTimerClicked"/>
15             </StackLayout.GestureRecognizers>
16             <!-- Image inside StackLayout -->
17             <Image Source="undraw_time_management_re_th5w.svg"
18                 HeightRequest="100"
19                 WidthRequest="100"
20                 HorizontalOptions="Center"
21                 VerticalOptions="Center"/>
22
23             <!-- Label inside StackLayout -->
24             <Label Text="TIMER"
25                 TextColor="{StaticResource SharcBlack}"
26                 FontFamily="ReemKufiFunRegular"
27                 FontSize="13"
28                 HorizontalOptions="Center"
29                 VerticalOptions="End"/>
30         </StackLayout>
31
32         <!-- NOTES panel -->
33         <StackLayout WidthRequest="150" HeightRequest="150"
34             AbsoluteLayout.LayoutBounds="0.75, 0.25, -0.5, -1"
35             AbsoluteLayout.LayoutFlags="PositionProportional">
36             <StackLayout.GestureRecognizers>
37                 <TapGestureRecognizer Tapped="OnNotesClicked"/>
38             </StackLayout.GestureRecognizers>
39             <!-- Image inside StackLayout -->
40             <Image Source="undraw_add_notes_re_ln36.svg"
41                 HeightRequest="100"
42                 WidthRequest="100"
43                 HorizontalOptions="Center"
44                 VerticalOptions="Center"/>
45
46             <!-- Label inside StackLayout -->
47             <Label Text="NOTES"
48                 TextColor="{StaticResource SharcBlack}"
49                 FontFamily="ReemKufiFunRegular"
50                 FontSize="13"
51                 HorizontalOptions="Center"
52                 VerticalOptions="End"/>
53         </StackLayout>
54
55         <!-- KNOWLEDGE BASE panel -->
56         <StackLayout WidthRequest="150" HeightRequest="150"
57             AbsoluteLayout.LayoutBounds="0.25, 0.6, -0.5, -1"
58             AbsoluteLayout.LayoutFlags="PositionProportional">
59             <StackLayout.GestureRecognizers>
60                 <TapGestureRecognizer Tapped="OnknowledgeBaseClicked"/>
61             </StackLayout.GestureRecognizers>
62             <!-- Image inside StackLayout -->
63             <Image Source="undraw_education_f8ru.svg"
64                 HeightRequest="100"
65                 WidthRequest="100"
66                 HorizontalOptions="Center"
67                 VerticalOptions="Center"/>
68
69             <!-- Label inside StackLayout -->
70             <Label Text="KNOWLEDGE BASE"
71                 TextColor="{StaticResource SharcBlack}"
72                 FontFamily="ReemKufiFunRegular"
73                 FontSize="13"
74                 HorizontalOptions="Center"
75                 VerticalOptions="End"/>
76         </StackLayout>
77
78         <!-- CHALLENGES panel -->
79         <StackLayout WidthRequest="150" HeightRequest="150"
80             AbsoluteLayout.LayoutBounds="0.75, 0.6, -0.5, -1"
81             AbsoluteLayout.LayoutFlags="PositionProportional">
82             <StackLayout.GestureRecognizers>
83                 <TapGestureRecognizer Tapped="OnChallengesClicked"/>
84             </StackLayout.GestureRecognizers>
85             <!-- Image inside StackLayout -->
86             <Image Source="undraw_fitness_stats_sht6.svg"
87                 HeightRequest="100"
88                 WidthRequest="100"
89                 HorizontalOptions="Center"
90                 VerticalOptions="Center"/>
91
92             <!-- Label inside StackLayout -->
93             <Label Text="CHALLENGES"
94                 TextColor="{StaticResource SharcBlack}"
95                 FontFamily="ReemKufiFunRegular"
96                 FontSize="13"
97                 HorizontalOptions="Center"
98                 VerticalOptions="End"/>
99         </StackLayout>
100
101         <!-- Bottom Navigation -->
102         <Local BottomNavigationView AbsoluteLayout.LayoutBounds="0.5, 1, 1, AutoSize"
103             AbsoluteLayout.LayoutFlags="XProportional, WidthProportional"
104             ActivePage="{Binding ActivePage}"
105             NavigateCommand="{Binding NavigateCommand}"/>
106     </AbsoluteLayout>
107 </ContentPage>

```

Figure 67. Source code of the AttentionHomePage.xaml (tool: Visual Studio IDE [70], source: self)

7.2. Navigating between modules

Before explaining the creation of a custom routine it is important to understand the main navigation. As seen on figure 68, the Attention module (just like other modules) handles the navigation elegantly.



```
1 using RAMDHD.Views.MainScreens.Mindfulness;
2 using RAMDHD.Views.MainScreens.GraphTask;
3 using RAMDHD.Views.MainScreens.Organization;
4 using RAMDHD.Views.MainScreens.People;
5 using System.Windows.Input;
6 using System.ComponentModel;
7 using RAMDHD.Views.MainScreens.Attention.Flashcards.Educational;
8 using RAMDHD.Views.MainScreens.Attention.Flashcards.Challenging.Sample;
9
10 namespace RAMDHD.Views.MainScreens.Attention
11 {
12     public partial class AttentionHomePage : ContentPage, INotifyPropertyChanged
13     {
14         private int activePage;
15         public AttentionHomePage()
16         {
17             ActivePage = 0;
18             InitializeComponent();
19             BindingContext = this;
20         }
21
22         public int ActivePage
23         {
24             get => activePage;
25             set
26             {
27                 if (activePage != value)
28                 {
29                     activePage = value;
30                     OnPropertyChanged(nameof(ActivePage));
31                 }
32             }
33         }
34
35         public ICommand NavigateCommand => new Command<string>(async (page) =>
36         {
37             switch (page)
38             {
39                 case "Attention":
40                     await this.Navigation.PushAsync(new AttentionHomePage());
41                     break;
42                 case "Organization":
43                     await this.Navigation.PushAsync(new OrganizationHomePage());
44                     break;
45                 case "People":
46                     await this.Navigation.PushAsync(new PeopleHomePage());
47                     break;
48                 case "Mindfulness":
49                     await this.Navigation.PushAsync(new MindfulnessHomePage());
50                     break;
51                 case "GraphTasks":
52                     await this.Navigation.PushAsync(new ProcrastinationHomePage());
53                     break;
54             }
55         });
56
57         private async void OnTimerClicked(object sender, EventArgs e)
58         {
59             await Navigation.PushAsync(new TimerPage());
60         }
61
62         private async void OnNotesClicked(object sender, EventArgs e)
63         {
64             await Navigation.PushAsync(new NotesMenuPage());
65         }
66
67         private async void OnKnowledgeBaseClicked(object sender, EventArgs e)
68         {
69             await this.Navigation.PushAsync(new EducationalFlashcardsPage());
70         }
71
72         private async void OnChallengesClicked(object sender, EventArgs e)
73         {
74             await this.Navigation.PushAsync(new SampleChallengesFlashcardsPage());
75         }
76     }
77 }
```

Figure 68. Source code of the AttentionHomePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Module pages get the navigation bar injected as a reusable component as seen on figure 69. Due to the repetition of the navigation logic the author decided to extract the bottom navigation bar to a separate reusable component as seen on figure 70.

```
103      <!-- Bottom Navigation -->
104      <local:BottomNavigationView AbsoluteLayout.LayoutBounds="0.5, 1, 1, AutoSize"
105                               AbsoluteLayout.LayoutFlags="XProportional, WidthProportional, PositionProportional"
106                               ActivePage="{Binding ActivePage}"
107                               NavigateCommand="{Binding NavigateCommand}"/>
```

Figure 69. Reusable bottom navigation bar component in the AttentionHomePage.xaml (tool: Visual Studio IDE [70], source: self)

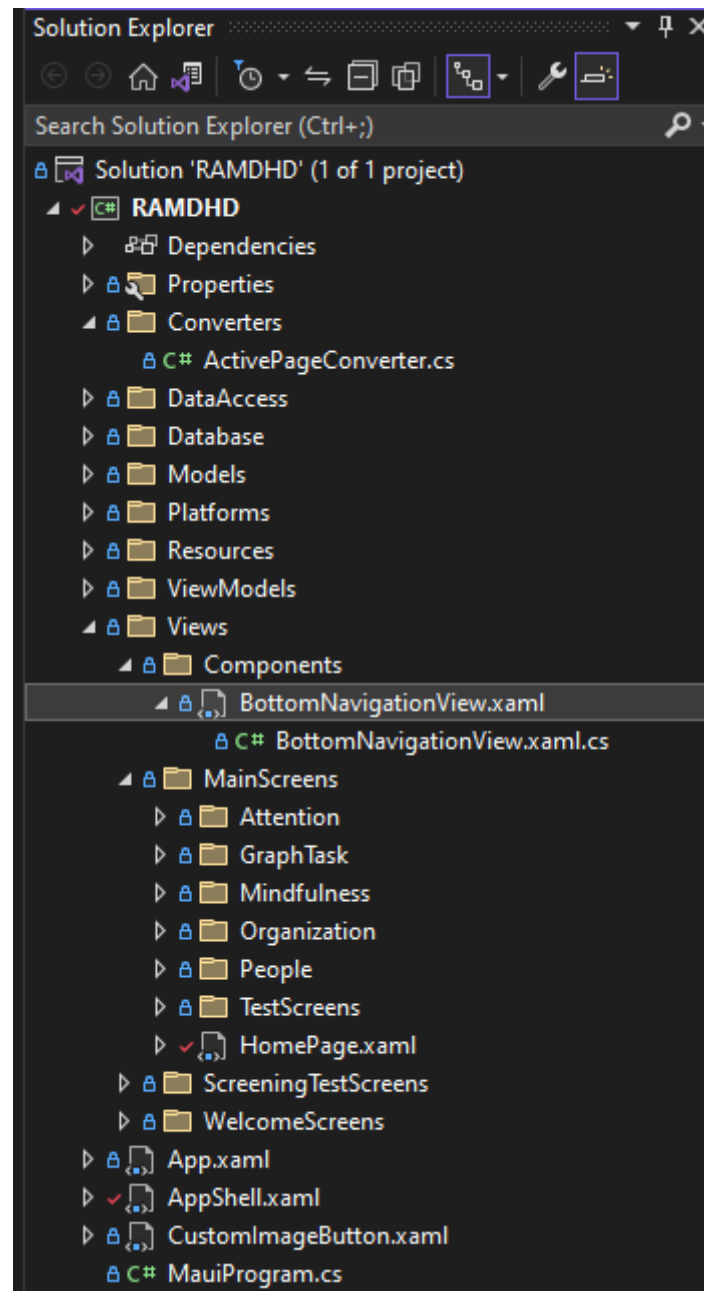
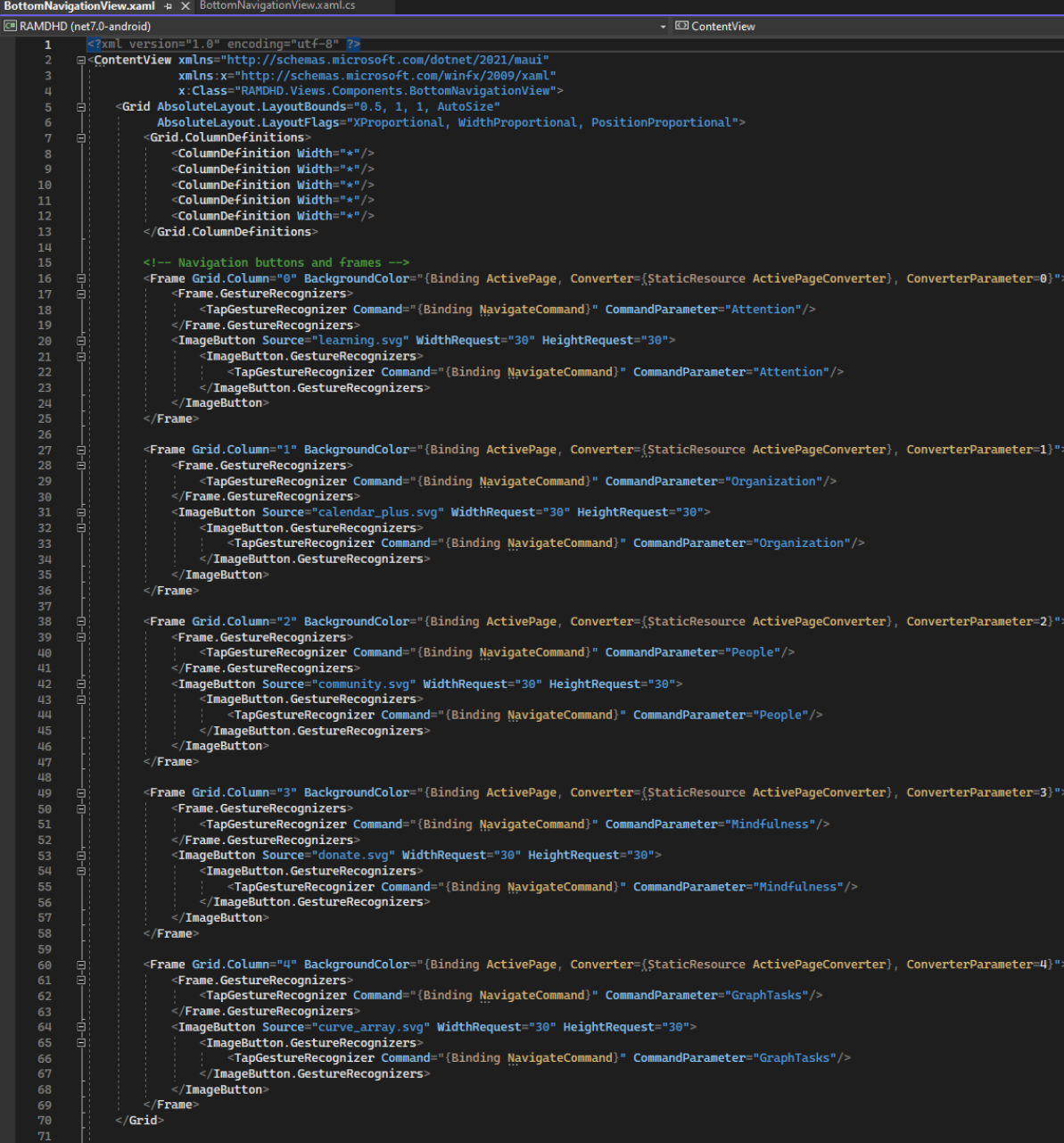


Figure 70. Reusable bottom navigation bar component in the Views/Components directory (tool: Visual Studio IDE [69], source: self)

The component is quite simple in logic which makes it a very good candidate for such extraction and reusability. The BottomNavigationView in the MAUI app uses a Grid layout to create a set of navigation buttons, each represented by a Frame containing an ImageButton as seen on figure 71.



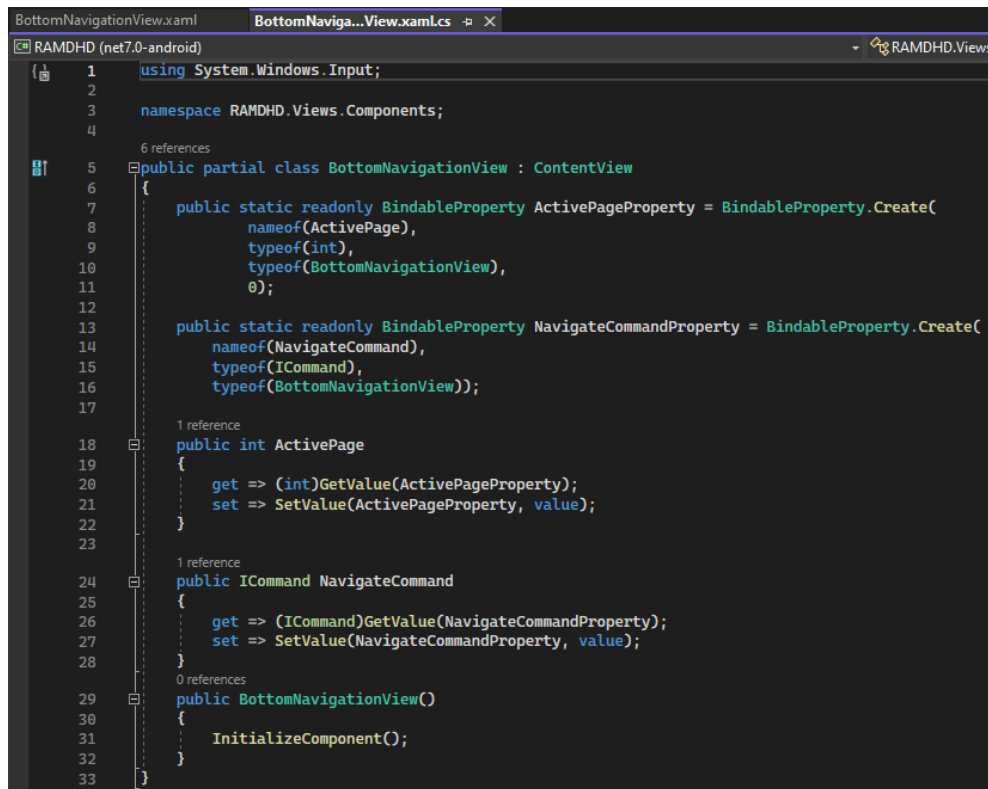
```

1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentView xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
3      xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4      x:Class="RAMDHD.Views.Components.BottomNavigationView">
5      <Grid AbsoluteLayout.LayoutBounds="0.5, 1, 1, AutoSize"
6          AbsoluteLayout.LayoutFlags="XProportional, WidthProportional, PositionProportional">
7          <Grid.ColumnDefinitions>
8              <ColumnDefinition Width="*" />
9              <ColumnDefinition Width="*" />
10             <ColumnDefinition Width="*" />
11             <ColumnDefinition Width="*" />
12             <ColumnDefinition Width="*" />
13         </Grid.ColumnDefinitions>
14
15         <!-- Navigation buttons and frames -->
16         <Frame Grid.Column="0" BackgroundColor="{Binding ActivePage, Converter={StaticResource ActivePageConverter}, ConverterParameter=0}">
17             <Frame.GestureRecognizers>
18                 <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="Attention"/>
19             </Frame.GestureRecognizers>
20             <ImageButton Source="learning.svg" WidthRequest="30" HeightRequest="30">
21                 <ImageButton.GestureRecognizers>
22                     <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="Attention"/>
23                 </ImageButton.GestureRecognizers>
24             </ImageButton>
25         </Frame>
26
27         <Frame Grid.Column="1" BackgroundColor="{Binding ActivePage, Converter={StaticResource ActivePageConverter}, ConverterParameter=1}">
28             <Frame.GestureRecognizers>
29                 <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="Organization"/>
30             </Frame.GestureRecognizers>
31             <ImageButton Source="calendar_plus.svg" WidthRequest="30" HeightRequest="30">
32                 <ImageButton.GestureRecognizers>
33                     <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="Organization"/>
34                 </ImageButton.GestureRecognizers>
35             </ImageButton>
36         </Frame>
37
38         <Frame Grid.Column="2" BackgroundColor="{Binding ActivePage, Converter={StaticResource ActivePageConverter}, ConverterParameter=2}">
39             <Frame.GestureRecognizers>
40                 <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="People"/>
41             </Frame.GestureRecognizers>
42             <ImageButton Source="community.svg" WidthRequest="30" HeightRequest="30">
43                 <ImageButton.GestureRecognizers>
44                     <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="People"/>
45                 </ImageButton.GestureRecognizers>
46             </ImageButton>
47         </Frame>
48
49         <Frame Grid.Column="3" BackgroundColor="{Binding ActivePage, Converter={StaticResource ActivePageConverter}, ConverterParameter=3}">
50             <Frame.GestureRecognizers>
51                 <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="Mindfulness"/>
52             </Frame.GestureRecognizers>
53             <ImageButton Source="donate.svg" WidthRequest="30" HeightRequest="30">
54                 <ImageButton.GestureRecognizers>
55                     <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="Mindfulness"/>
56                 </ImageButton.GestureRecognizers>
57             </ImageButton>
58         </Frame>
59
60         <Frame Grid.Column="4" BackgroundColor="{Binding ActivePage, Converter={StaticResource ActivePageConverter}, ConverterParameter=4}">
61             <Frame.GestureRecognizers>
62                 <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="GraphTasks"/>
63             </Frame.GestureRecognizers>
64             <ImageButton Source="curve_array.svg" WidthRequest="30" HeightRequest="30">
65                 <ImageButton.GestureRecognizers>
66                     <TapGestureRecognizer Command="{Binding NavigateCommand}" CommandParameter="GraphTasks"/>
67                 </ImageButton.GestureRecognizers>
68             </ImageButton>
69         </Frame>
70     </Grid>
71 </ContentView>
72

```

Figure 71. Source code of the BottomNavigationView.xaml (tool: Visual Studio IDE [70], source: self)

These buttons use a TapGestureRecognizer to bind tap actions to a NavigateCommand that changes the current ActivePage as seen on figure 72. As seen on figure 73 the ActivePageConverter compares the ActivePage (the page currently being viewed) with the buttonPageIndex (the page each button represents, passed as a ConverterParameter) to change the background color of each Frame, providing visual feedback. When a button is tapped, the command updates the ActivePage, and the converter updates the button's appearance to indicate the active page, ensuring an interactive and responsive navigation experience.

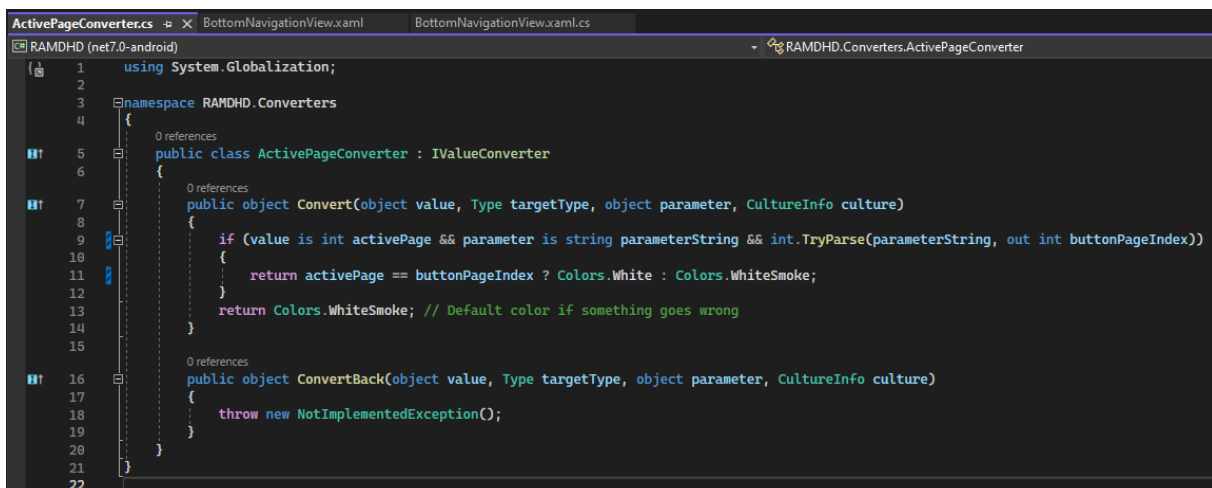


```

1  using System.Windows.Input;
2
3  namespace RAMDHD.Views.Components;
4
5  public partial class BottomNavigationView : ContentView
6  {
7      public static readonly BindableProperty ActivePageProperty = BindableProperty.Create(
8          nameof(ActivePage),
9          typeof(int),
10         typeof(BottomNavigationView),
11         0);
12
13     public static readonly BindableProperty NavigateCommandProperty = BindableProperty.Create(
14         nameof(NavigateCommand),
15         typeof(ICommand),
16         typeof(BottomNavigationView));
17
18     public int ActivePage
19     {
20         get => (int)GetValue(ActivePageProperty);
21         set => SetValue(ActivePageProperty, value);
22     }
23
24     public ICommand NavigateCommand
25     {
26         get => (ICommand)GetValue(NavigateCommandProperty);
27         set => SetValue(NavigateCommandProperty, value);
28     }
29
30     public BottomNavigationView()
31     {
32         InitializeComponent();
33     }

```

Figure 72. Source code of the BottomNavigationView.xaml.cs (tool: Visual Studio IDE [70], source: self)



```

1  using System.Globalization;
2
3  namespace RAMDHD.Converters
4  {
5      public class ActivePageConverter : IValueConverter
6      {
7          public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
8          {
9              if (value is int activePage && parameter is string parameterString && int.TryParse(parameterString, out int buttonPageIndex))
10             {
11                 return activePage == buttonPageIndex ? Colors.White : Colors.WhiteSmoke;
12             }
13             return Colors.WhiteSmoke; // Default color if something goes wrong
14         }
15
16         public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
17         {
18             throw new NotImplementedException();
19         }
20     }
21 }

```

Figure 73. Source code of the ActivePageConverter.cs (tool: Visual Studio IDE [70], source: self)

7.3. Creating custom routine

Since the navigation implementation was explained, this and the following subchapters will go in depth into the implementation behind each of the functionalities tested during the task completion.

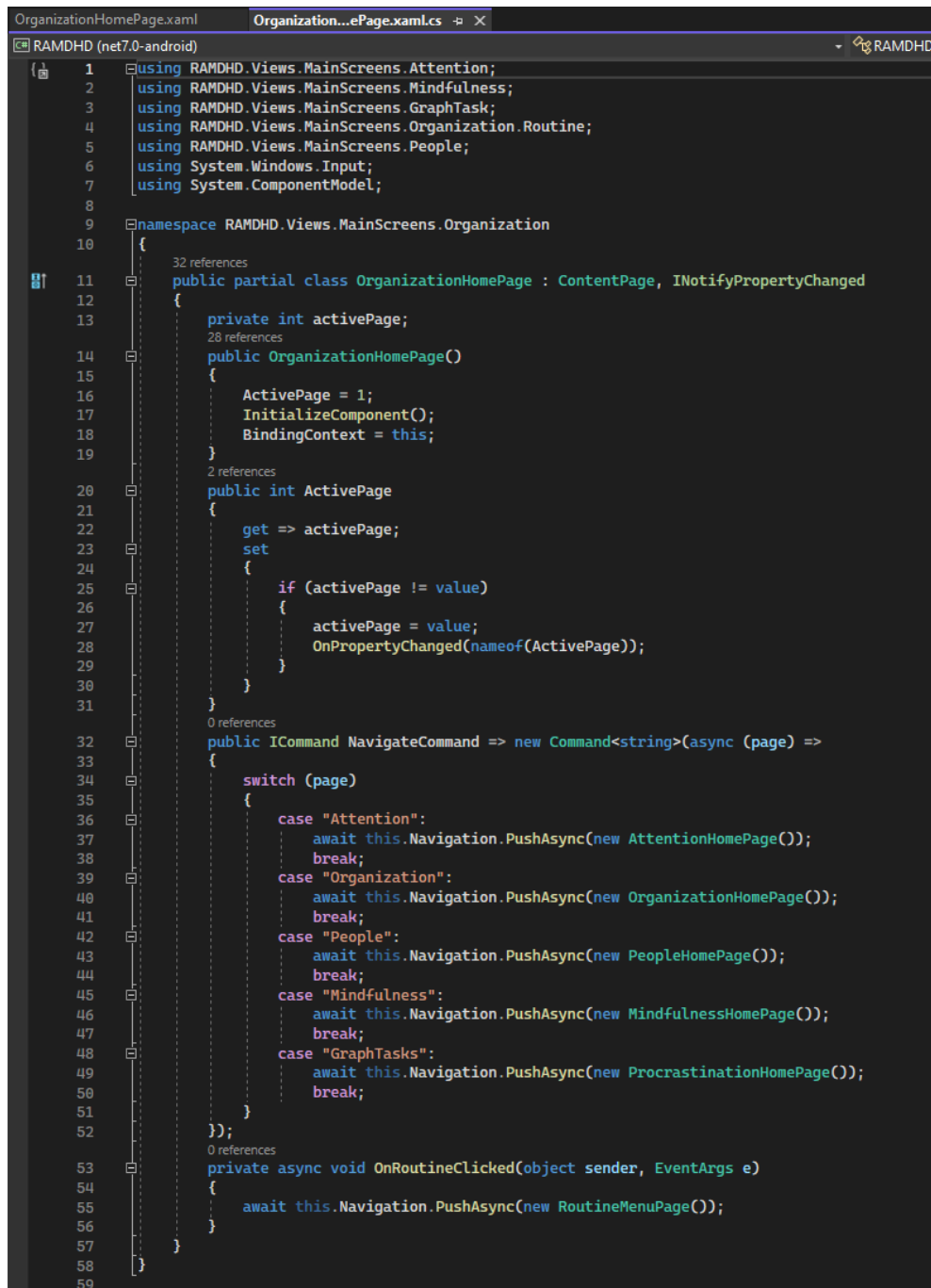
Creating a custom routine first requires navigating to the Routines panel inside the Organization module. The XAML code of the Organization module is presented on figure 74. Organization module contains 3 panels - Calendar, Diary and Routines. In the current version of the application only Routines panel is active.

```
OrganizationHomePage.xaml  OrganizationHomePage.xaml.cs
RAMDHD (net7.0-android)  AbsoluteLayout

7  <AbsoluteLayout HorizontalOptions="FillAndExpand" VerticalOptions="FillAndExpand">
8
9
10  <!-- CALENDAR panel -->
11  <StackLayout
12    WidthRequest="150" HeightRequest="150"
13    AbsoluteLayout.LayoutBounds="0.25, 0.25, -0.5, -1"
14    AbsoluteLayout.LayoutFlags="PositionProportional">
15    <StackLayout.GestureRecognizers>
16      <TapGestureRecognizer/>
17    </StackLayout.GestureRecognizers>
18    <Image Source="undraw_online_calendar_re_wk3t_blacknwhite.svg"
19    HeightRequest="100"
20    WidthRequest="100"
21    HorizontalOptions="Center"
22    VerticalOptions="Center"/>
23    <Label Text="CALENDAR"
24    TextColor="{StaticResource SharcBlack}"
25    FontFamily="ReemKufiFunRegular"
26    FontSize="13"
27    HorizontalTextAlignment="Center"
28    VerticalOptions="End"/>
29  </StackLayout>
30
31  <!-- DIARY panel -->
32  <StackLayout
33    WidthRequest="150" HeightRequest="150"
34    AbsoluteLayout.LayoutBounds="0.75, 0.25, -0.5, -1"
35    AbsoluteLayout.LayoutFlags="PositionProportional">
36    <StackLayout.GestureRecognizers>
37      <TapGestureRecognizer/>
38    </StackLayout.GestureRecognizers>
39    <Image Source="undraw_diary_re_4jpc_blacknwhite.svg"
40    HeightRequest="100"
41    WidthRequest="100"
42    HorizontalOptions="Center"
43    VerticalOptions="Center"/>
44    <Label Text="DIARY"
45    TextColor="{StaticResource SharcBlack}"
46    FontFamily="ReemKufiFunRegular"
47    FontSize="13"
48    HorizontalTextAlignment="Center"
49    VerticalOptions="End"/>
50  </StackLayout>
51
52
53  <!-- ROUTINES panel -->
54  <StackLayout
55    WidthRequest="150" HeightRequest="150"
56    AbsoluteLayout.LayoutBounds="0.25, 0.6, -0.5, -1"
57    AbsoluteLayout.LayoutFlags="PositionProportional">
58    <StackLayout.GestureRecognizers>
59      <TapGestureRecognizer Tapped="OnRoutineClicked"/>
60    </StackLayout.GestureRecognizers>
61    <Image Source="undraw_to_do_re_jaef.svg"
62    HeightRequest="100"
63    WidthRequest="100"
64    HorizontalOptions="Center"
65    VerticalOptions="Center"/>
66    <Label Text="ROUTINES"
67    TextColor="{StaticResource SharcBlack}"
68    FontFamily="ReemKufiFunRegular"
69    FontSize="13"
70    HorizontalTextAlignment="Center"
71    VerticalOptions="End"/>
72  </StackLayout>
73
74  <!-- Bottom Navigation -->
75  <Local:BottomNavigationView AbsoluteLayout.LayoutBounds="0.5, 1, 1, AutoSize"
76    AbsoluteLayout.LayoutFlags="XProportional, WidthProportional, PositionProportional"
77    ActivePage="{Binding ActivePage}"
78    NavigateCommand="{Binding NavigateCommand}"/>
```

Figure 74. Source code of the OrganizationHomePage.xaml (tool: Visual Studio IDE [70], source: self)

Navigation logic of the organization module is presented on figure 75. To proceed to the Routines panel user shall click on the “ROUTINES” button which would trigger OnRoutineClicked function.



```

1  using RAMDHD.Views.MainScreens.Attention;
2  using RAMDHD.Views.MainScreens.Mindfulness;
3  using RAMDHD.Views.MainScreens.GraphTask;
4  using RAMDHD.Views.MainScreens.Organization.Routine;
5  using RAMDHD.Views.MainScreens.People;
6  using System.Windows.Input;
7  using System.ComponentModel;
8
9  namespace RAMDHD.Views.MainScreens.Organization
10 {
11     public partial class OrganizationHomePage : ContentPage, INotifyPropertyChanged
12     {
13         private int activePage;
14         public OrganizationHomePage()
15         {
16             ActivePage = 1;
17             InitializeComponent();
18             BindingContext = this;
19         }
20         public int ActivePage
21         {
22             get => activePage;
23             set
24             {
25                 if (activePage != value)
26                 {
27                     activePage = value;
28                     OnPropertyChanged(nameof(ActivePage));
29                 }
30             }
31         }
32         public ICommand NavigateCommand => new Command<string>(async (page) =>
33         {
34             switch (page)
35             {
36                 case "Attention":
37                     await this.Navigation.PushAsync(new AttentionHomePage());
38                     break;
39                 case "Organization":
40                     await this.Navigation.PushAsync(new OrganizationHomePage());
41                     break;
42                 case "People":
43                     await this.Navigation.PushAsync(new PeopleHomePage());
44                     break;
45                 case "Mindfulness":
46                     await this.Navigation.PushAsync(new MindfulnessHomePage());
47                     break;
48                 case "GraphTasks":
49                     await this.Navigation.PushAsync(new ProcrastinationHomePage());
50                     break;
51             }
52         });
53         private async void OnRoutineClicked(object sender, EventArgs e)
54         {
55             await this.Navigation.PushAsync(new RoutineMenuPage());
56         }
57     }
58 }
59

```

Figure 75. Source code of the OrganizationHomePage.xamlcs (tool: Visual Studio IDE [70], source: self)

As seen on figure 76 on the Routine Menu Page there are 5 slots available for saving custom routines. Clicking the orange “edit_pencil.svg” icon button triggers the EditRoutine function and redirects the user to the Edit Routine page of the given routine (based on its ID).

```

RoutineMenuPage.xaml  RoutineMenuPage.xaml.cs
RAMDHD (net7.0-android)
1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
3    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4    xmlns:local="clr-namespace:RAMDHD.Views.Components"
5    x:Class="RAMDHD.Views.MainScreens.Organization.Routine.RoutineMenuPage"
6    Title="Routine Menu Page"
7    BackgroundColor="White">
8    <AbsoluteLayout x:Name="MainLayout" VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand">
9
10     <!-- Header -->
11     <HorizontalStackLayout
12       AbsoluteLayout.LayoutBounds="0.2, 0.02, -1, -1"
13       AbsoluteLayout.LayoutFlags="PositionProportional">
14       <Label
15         FontSize="16"
16         TextColor="{StaticResource Tertiary}"
17         FontFamily="ReemKufiFunBold"
18         HorizontalOptions="Center"
19         HorizontalTextAlignment="Center"
20       />
21     </HorizontalStackLayout>
22
23     <!-- Placeholder Frames for Adding New Graph Tasks -->
24     <Frame
25       x:Name="placeholder1"
26       WidthRequest="400" HeightRequest="100"
27       BackgroundColor="WhiteSmoke"
28       CornerRadius="10"
29       AbsoluteLayout.LayoutBounds="0.5, 0.16, 0.5, 1"
30       AbsoluteLayout.LayoutFlags="PositionProportional">
31       <Frame.GestureRecognizers>
32         <TapGestureRecognizer Tapped="OnPlaceholderTapped" CommandParameter="1"/>
33       </Frame.GestureRecognizers>
34       <HorizontalStackLayout>
35         <Label
36           x:Name="placeholderLabel1"
37           Text="Add new routine"
38           FontFamily="ReemKufiFunRegular"
39           TextColor="{StaticResource Gray200}"
40           HeightRequest="50"
41           WidthRequest="220"
42           HorizontalOptions="Center"
43           VerticalOptions="CenterAndExpand"
44           Margin="40, 20, 0, 0">
45
46         </Label>
47         <Image Source="{Binding EditIcon, FallbackValue='edit_pencil.svg'}"
48           WidthRequest="30"
49           HeightRequest="30"
50           HorizontalOptions="EndAndExpand"
51           VerticalOptions="Center"
52           Margin="0, 0, 0, 0">
53           <Image.GestureRecognizers>
54             <TapGestureRecognizer Tapped="EditRoutine" CommandParameter="1"/>
55           </Image.GestureRecognizers>
56         </Image>
57         <Image
58           Source="{Binding DeleteIcon, FallbackValue='x.svg'}"
59           WidthRequest="30"
60           HeightRequest="30"
61           HorizontalOptions="EndAndExpand"
62           VerticalOptions="Center"
63           Margin="20, 0, 0, 0">
64           <Image.GestureRecognizers>
65             <TapGestureRecognizer Tapped="DeleteRoutine" CommandParameter="1"/>
66           </Image.GestureRecognizers>
67         </Image>
68       </HorizontalStackLayout>
69     </Frame>
70
71     <Frame
72       x:Name="placeholder2"
73       WidthRequest="400" HeightRequest="100"
74       BackgroundColor="WhiteSmoke"
75       CornerRadius="10"
76       AbsoluteLayout.LayoutBounds="0.5, 0.32, 0.5, 1"
77       AbsoluteLayout.LayoutFlags="PositionProportional">
78       <Frame.GestureRecognizers>
79         <TapGestureRecognizer Tapped="OnPlaceholderTapped" CommandParameter="2"/>
80       </Frame.GestureRecognizers>
81       <HorizontalStackLayout>
82         <Label
83           x:Name="placeholderLabel2"
84           Text="Add new routine"
85           FontFamily="ReemKufiFunRegular"
86           TextColor="{StaticResource Gray200}"
87           HeightRequest="50"
88           WidthRequest="220"
89           HorizontalOptions="Center"
90           VerticalOptions="CenterAndExpand"
91           Margin="40, 20, 0, 0">
92
93         </Label>
94         <Image Source="{Binding EditIcon, FallbackValue='edit_pencil.svg'}"
95           WidthRequest="30"
96           HeightRequest="30"
97           HorizontalOptions="EndAndExpand"
98           VerticalOptions="Center"
99           Margin="0, 0, 0, 0">
100          <Image.GestureRecognizers>
101            <TapGestureRecognizer Tapped="EditRoutine" CommandParameter="2"/>
102          </Image.GestureRecognizers>
103        </Image>
104        <Image
105          Source="{Binding DeleteIcon, FallbackValue='x.svg'}"
106          WidthRequest="30"

```

Figure 76. Source code of the RoutineMenuPage.xaml (tool: Visual Studio IDE [70], source: self)

As seen on figure 77 the EditRoutine method is an asynchronous event handler designed to handle tap events on images within a Xamarin.Forms or MAUI application. When an image is tapped, the method first verifies that the sender is an Image with a TapGestureRecognizer. It then retrieves an ID from the CommandParameter of the gesture recognizer, logs this ID to the console, and navigates to the EditRoutinePage, passing the ID as a parameter. This allows the application to dynamically load the appropriate edit page based on which image was tapped.

```

RoutineMenuPage.xaml RoutineMenuPage.xaml.cs -B X
RAMDHD (net7.0-android)
87 private async void OnPlaceholderTapped(object sender, EventArgs e)
88 {
89     if (sender is Label label && label.GestureRecognizers.FirstOrDefault() is TapGestureRecognizer tapGesture)
90     {
91         string id = tapGesture.CommandParameter.ToString();
92         Console.WriteLine($"Label {label.Text} with ID {id} tapped");
93         if (label.Text == "Add new routine")
94         {
95             Console.WriteLine($"Image with ID {id} tapped");
96             // The label indicates that a new graph task should be added.
97             // Navigate to the EditGraphTaskPage to add a new task.
98             await Navigation.PushAsync(new EditNotePage(id));
99         }
100     }
101     else
102     {
103         // The label indicates an existing graph task.
104         // Navigate to the GraphTaskTemplatePage to view the task.
105         // Pass the ID as a parameter to the page.
106         await Navigation.PushAsync(new RoutineTemplatePage(id));
107         Console.WriteLine($"Title {label.Text} with ID {id} tapped");
108     }
109 }
110
111 // Reference
112 private async void EditRoutine(object sender, EventArgs e)
113 {
114     if (sender is Image image && image.GestureRecognizers.FirstOrDefault() is TapGestureRecognizer tapGesture)
115     {
116         string id = tapGesture.CommandParameter.ToString();
117         Console.WriteLine($"Image with ID {id} tapped");
118         await Navigation.PushAsync(new EditRoutinePage(id));
119     }
120 }
121
122 // Reference
123 private async void DeleteRoutine(object sender, EventArgs e)
124 {
125     if (sender is Image image && image.GestureRecognizers.FirstOrDefault() is TapGestureRecognizer tapGesture)
126     {
127         int.TryParse(tapGesture.CommandParameter.ToString(), out int id);
128
129         bool answer = await Application.Current.MainPage.DisplayAlert(
130             "Confirm Delete",
131             "Are you sure you want to delete this routine?",
132             "Yes",
133             "No"
134         );
135
136         if (answer)
137         {
138             // Perform the delete action
139             await _databaseAccess.DeleteRoutineByIdAsync(id);
140
141             await DisplayAlert("Success", $"Routine deleted successfully", "OK");
142
143             // Insert the new GraphTaskMenu page before the current page and then pop the current one
144             // Creates the illusion of page refreshing
145             Navigation.InsertPageBefore(new RoutineMenuPage(), this);
146             await Navigation.PopAsync();
147         }
148     }
149 }
150
151 // Reference
152 private async void LoadAndDisplayNotes()
153 {
154     var routines = await _databaseAccess.GetAllRoutinesAsync();
155     foreach (var routine in routines)
156     {
157         Console.WriteLine($"LOADED routine with id {routine.Id}, title {routine.Title}");
158     }
159     DisplayNote(routine);
160 }
161
162 // Reference
163 private void DisplayNote(Models.Routine routine)
164 {
165     // If the task title is null or empty, set the default title.
166     string title = string.IsNullOrEmpty(routine.Title) ? "Add new routine" : routine.Title;
167     Console.WriteLine($"routine {routine.Id}, Title {title}");
168
169     var viewModel = new RoutineViewModel
170     {
171         Title = title,
172         EditIcon = "edit_pencil.svg",
173         DeleteIcon = "trash.svg"
174     };
175
176     Frame placeholder = FindPlaceholderById(routine.Id);
177     if (placeholder != null)
178     {
179         // Ensuring the Frame has a HorizontalStackLayout to hold the content
180         var stackLayout = placeholder.Content as HorizontalStackLayout ?? new HorizontalStackLayout();
181         placeholder.Content = stackLayout;
182
183         // Applying the ViewModel to the Frame's BindingContext
184         stackLayout.BindingContext = viewModel;
185
186         // Optionally clear existing children if needed
187         stackLayout.Children.Clear();
188
189         // Add the newly bound image and label to the StackLayout
190         stackLayout.Children.Add(CreateLabel(viewModel, routine.Id));
191         stackLayout.Children.Add(CreateEditIcon(viewModel, routine.Id));
192         stackLayout.Children.Add(CreateDeleteIcon(viewModel, routine.Id));
193     }
194 }

```

Figure 77. Source code of the RoutineMenuPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

EditRoutinePage.xaml is presented on figure 78. It is responsible for handling text insertion, saving or canceling the edition of the routine.

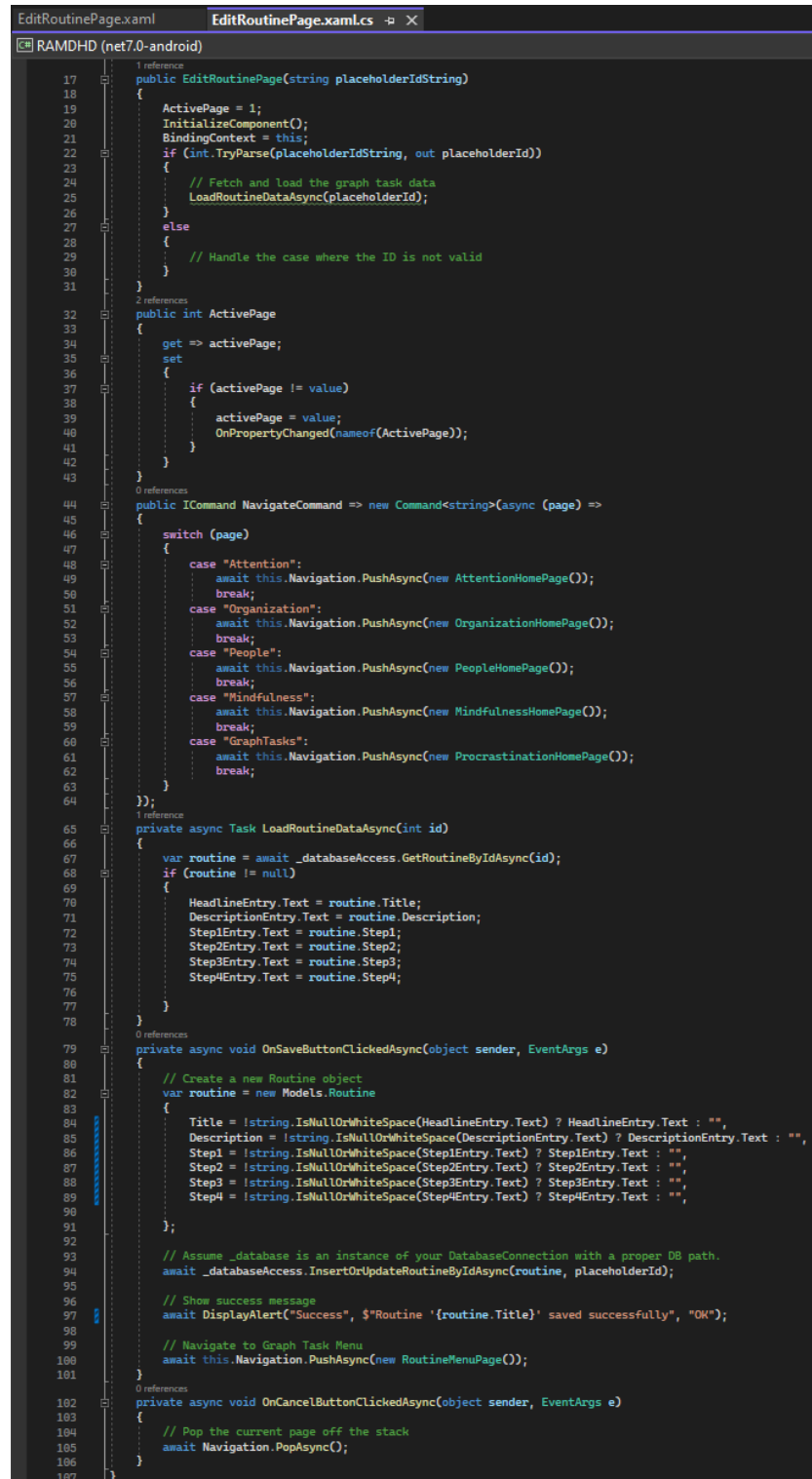
```

13  <Entry x:Name="HeadLineEntry"
14      Placeholder="Enter routine name"
15      FontSize="16"
16      TextColor="{StaticResource SharcBlack}"
17      FontFamily="ReemKufiFunBold"
18      MaxLength="160" />
19
20  <Label
21      Text="Below you can describe the routine precisely"
22      LineBreakMode="WordWrap"
23      WidthRequest="300"
24      FontSize="13"
25      TextColor="{StaticResource Tertiary}"
26      FontFamily="ReemKufiFunBold"/>
27
28  <Entry x:Name="DescriptionEntry"
29      Placeholder="Enter note description"
30      FontSize="16"
31      TextColor="{StaticResource SharcBlack}"
32      FontFamily="ReemKufiFunBold"
33      MaxLength="160" />
34
35  <Grid>
36      <Grid.RowDefinitions>
37          <RowDefinition Height="Auto" />
38          <RowDefinition Height="Auto" />
39          <RowDefinition Height="Auto" />
40          <RowDefinition Height="Auto" />
41      </Grid.RowDefinitions>
42
43      <Grid.ColumnDefinitions>
44          <ColumnDefinition Width="Auto" />
45          <ColumnDefinition Width="*" />
46      </Grid.ColumnDefinitions>
47
48      <Label Grid.Row="0" Grid.Column="0"
49          Text="1." TextColor="{StaticResource Tertiary}" FontFamily="ReemKufiFunBold" FontSize="16"
50          VerticalTextAlignment="Center" />
51      <Entry x:Name="Step1Entry" Grid.Row="0" Grid.Column="1"
52          Placeholder="Enter step 1 of the routine" FontSize="16" TextColor="{StaticResource SharcBlack}" FontFamily="ReemKufiFunBold"
53          MaxLength="160" />
54
55      <Label Grid.Row="1" Grid.Column="0" Text="2."
56          TextColor="{StaticResource Tertiary}" FontFamily="ReemKufiFunBold" FontSize="16"
57          VerticalTextAlignment="Center" />
58      <Entry x:Name="Step2Entry" Grid.Row="1" Grid.Column="1"
59          Placeholder="Enter step 2 of the routine" FontSize="16" TextColor="{StaticResource SharcBlack}" FontFamily="ReemKufiFunBold"
60          MaxLength="160" />
61
62      <Label Grid.Row="2" Grid.Column="0" Text="3."
63          TextColor="{StaticResource Tertiary}" FontFamily="ReemKufiFunBold" FontSize="16"
64          VerticalTextAlignment="Center" />
65      <Entry x:Name="Step3Entry" Grid.Row="2" Grid.Column="1"
66          Placeholder="Enter step 3 of the routine" FontSize="16" TextColor="{StaticResource SharcBlack}" FontFamily="ReemKufiFunBold"
67          MaxLength="160" />
68
69      <Label Grid.Row="3" Grid.Column="0" Text="4."
70          TextColor="{StaticResource Tertiary}" FontFamily="ReemKufiFunBold" FontSize="16"
71          VerticalTextAlignment="Center" />
72      <Entry x:Name="Step4Entry" Grid.Row="3" Grid.Column="1"
73          Placeholder="Enter step 4 of the routine" FontSize="16" TextColor="{StaticResource SharcBlack}" FontFamily="ReemKufiFunBold"
74          MaxLength="160" />
75  </Grid>
76
77  </VerticalStackLayout>
78
79  <!-- Save Button -->
80  <Button
81      Text="SAVE"
82      WidthRequest="200" HeightRequest="50"
83      Margin="0,100,0,0"
84      BackgroundColor="LightGreen" TextColor="Green" BorderWidth="1"
85      HorizontalOptions="Center" VerticalOptions="Center"
86      AbsoluteLayout.LayoutBounds="0.5, 0.67, 0, 0"
87      AbsoluteLayout.LayoutFlags="PositionProportional"
88      Clicked="OnSaveButtonClickedAsync"
89      IsEnabled="True"
90  />
91
92  <!-- Delete Button -->
93  <Button
94      Text="CANCEL"
95      WidthRequest="200" HeightRequest="50"
96      AbsoluteLayout.LayoutBounds="0.5, 0.83, 0, 0"
97      AbsoluteLayout.LayoutFlags="PositionProportional"
98      BackgroundColor="Red" TextColor="DarkRed" BorderWidth="1"
99      HorizontalOptions="Center" VerticalOptions="Center"
100     Clicked="OnCancelButtonClickedAsync"
101     IsEnabled="True"
102  />
103
104  <!-- Bottom Navigation -->
105  <local:BottomNavigationView AbsoluteLayout.LayoutBounds="0.5, 1, 1, AutoSize"
106      AbsoluteLayout.LayoutFlags="XProportional, WidthProportional, PositionProportional"
107      ActivePage="{Binding ActivePage}"
108      NavigateCommand="{Binding NavigateCommand}"/>

```

Figure 78. Source code of the EditRoutinePage.xaml (tool: Visual Studio IDE [70], source: self)

First of all, if there is an existing routine under the given ID it is fetched from the database and presented to the user so that the user can literally edit an existing routine instead of retyping it again. OnSaveButtonClicked handles saving of the routine as seen on figure 79. First, entries are validated whether they are non empty. If in fact they are empty, they are simply saved as such because the user is not forced to fill all the entries of the routine. The validation is however necessary to ensure reliability of the system.



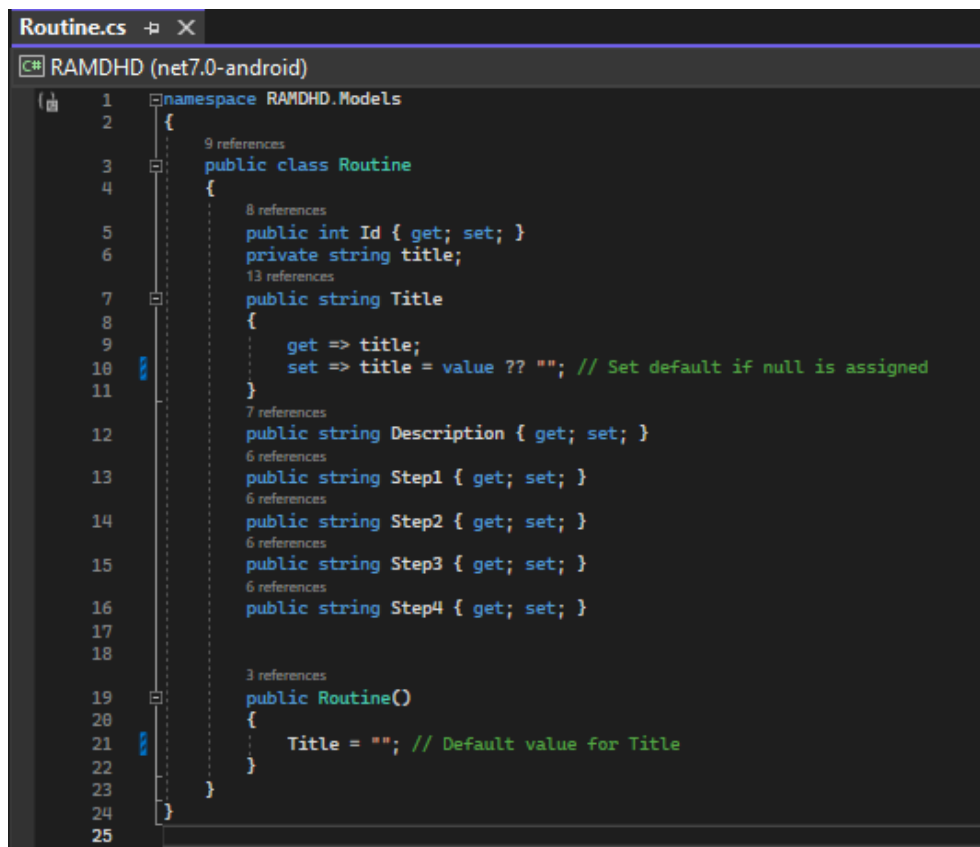
```

EditRoutinePage.xaml EditRoutinePage.xaml.cs
RAMDHD (net7.0-android)
17 1 reference
18 public EditRoutinePage(string placeholderIdString)
19 {
20     ActivePage = 1;
21     InitializeComponent();
22     BindingContext = this;
23     if (int.TryParse(placeholderIdString, out placeholderId))
24     {
25         // Fetch and load the graph task data
26         LoadRoutineDataAsync(placeholderId);
27     }
28     else
29     {
30         // Handle the case where the ID is not valid
31     }
32 }
33 2 references
34 public int ActivePage
35 {
36     get => activePage;
37     set
38     {
39         if (activePage != value)
40         {
41             activePage = value;
42             OnPropertyChanged(nameof(ActivePage));
43         }
44     }
45 }
46 0 references
47 public ICommand NavigateCommand => new Command<string>(async (page) =>
48 {
49     switch (page)
50     {
51         case "Attention":
52             await this.Navigation.PushAsync(new AttentionHomePage());
53             break;
54         case "Organization":
55             await this.Navigation.PushAsync(new OrganizationHomePage());
56             break;
57         case "People":
58             await this.Navigation.PushAsync(new PeopleHomePage());
59             break;
60         case "Mindfulness":
61             await this.Navigation.PushAsync(new MindfulnessHomePage());
62             break;
63         case "GraphTasks":
64             await this.Navigation.PushAsync(new ProcrastinationHomePage());
65             break;
66     }
67 });
68 1 reference
69 private async Task LoadRoutineDataAsync(int id)
70 {
71     var routine = await _databaseAccess.GetRoutineByIdAsync(id);
72     if (routine != null)
73     {
74         HeadlineEntry.Text = routine.Title;
75         DescriptionEntry.Text = routine.Description;
76         Step1Entry.Text = routine.Step1;
77         Step2Entry.Text = routine.Step2;
78         Step3Entry.Text = routine.Step3;
79         Step4Entry.Text = routine.Step4;
80     }
81 }
82 0 references
83 private async void OnSaveButtonClickedAsync(object sender, EventArgs e)
84 {
85     // Create a new Routine object
86     var routine = new Models.Routine
87     {
88         Title = !string.IsNullOrWhiteSpace(HeadlineEntry.Text) ? HeadlineEntry.Text : "",
89         Description = !string.IsNullOrWhiteSpace(DescriptionEntry.Text) ? DescriptionEntry.Text : "",
90         Step1 = !string.IsNullOrWhiteSpace(Step1Entry.Text) ? Step1Entry.Text : "",
91         Step2 = !string.IsNullOrWhiteSpace(Step2Entry.Text) ? Step2Entry.Text : "",
92         Step3 = !string.IsNullOrWhiteSpace(Step3Entry.Text) ? Step3Entry.Text : "",
93         Step4 = !string.IsNullOrWhiteSpace(Step4Entry.Text) ? Step4Entry.Text : ""
94     };
95     // Assume _database is an instance of your DatabaseConnection with a proper DB path.
96     await _databaseAccess.InsertOrUpdateRoutineByIdAsync(routine, placeholderId);
97     // Show success message
98     await DisplayAlert("Success", $"Routine '{routine.Title}' saved successfully", "OK");
99     // Navigate to Graph Task Menu
100    await this.Navigation.PushAsync(new RoutineMenuPage());
101 }
102 0 references
103 private async void OnCancelButtonClickedAsync(object sender, EventArgs e)
104 {
105     // Pop the current page off the stack
106     await Navigation.PopAsync();
107 }

```

Figure 79. Source code of the EditRoutinePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

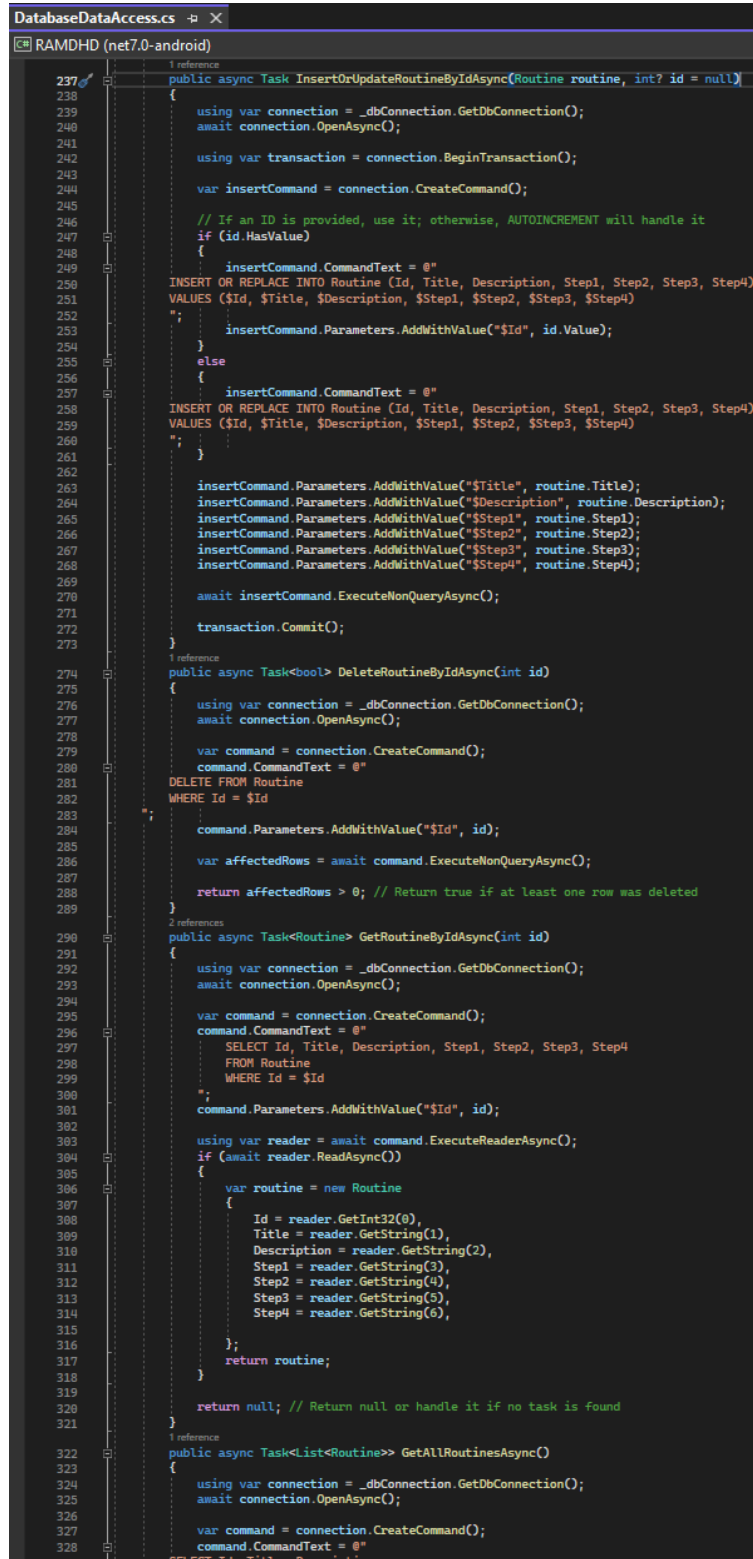
As previously seen on figure 79, a routine model object is created. The routine model contains fundamental information about a routine as seen on figure 80.



```
1 namespace RAMDHD.Models
2 {
3     9 references
4     public class Routine
5     {
6         8 references
7         public int Id { get; set; }
8         private string title;
9         13 references
10        public string Title
11        {
12            get => title;
13            set => title = value ?? ""; // Set default if null is assigned
14        }
15        7 references
16        public string Description { get; set; }
17        6 references
18        public string Step1 { get; set; }
19        6 references
20        public string Step2 { get; set; }
21        6 references
22        public string Step3 { get; set; }
23        6 references
24        public string Step4 { get; set; }
25
26        3 references
27        public Routine()
28        {
29            Title = ""; // Default value for Title
30        }
31    }
32 }
```

Figure 80. Source code of the Routine.cs model (tool: Visual Studio IDE [70], source: self)

The DatabaseDataAccess.cs class which acts as an intermediary between the application screen logic and database as seen on figure 81. InsertOrUpdateRoutineByIdAsync is an asynchronous function that is responsible for calling the database for insertion query execution.



```

DatabaseDataAccess.cs
RAMDHD (net7.0-android)

237 public async Task InsertOrUpdateRoutineByIdAsync(Routine routine, int? id = null)
238 {
239     using var connection = _dbConnection.GetDbConnection();
240     await connection.OpenAsync();
241
242     using var transaction = connection.BeginTransaction();
243
244     var insertCommand = connection.CreateCommand();
245
246     // If an ID is provided, use it; otherwise, AUTOINCREMENT will handle it
247     if (id.HasValue)
248     {
249         insertCommand.CommandText = @"
250 INSERT OR REPLACE INTO Routine (Id, Title, Description, Step1, Step2, Step3, Step4)
251 VALUES ($Id, $Title, $Description, $Step1, $Step2, $Step3, $Step4)
252 ";
253         insertCommand.Parameters.AddWithValue("$Id", id.Value);
254     }
255     else
256     {
257         insertCommand.CommandText = @"
258 INSERT OR REPLACE INTO Routine (Id, Title, Description, Step1, Step2, Step3, Step4)
259 VALUES ($Id, $Title, $Description, $Step1, $Step2, $Step3, $Step4)
260 ";
261     }
262
263     insertCommand.Parameters.AddWithValue("$Title", routine.Title);
264     insertCommand.Parameters.AddWithValue("$Description", routine.Description);
265     insertCommand.Parameters.AddWithValue("$Step1", routine.Step1);
266     insertCommand.Parameters.AddWithValue("$Step2", routine.Step2);
267     insertCommand.Parameters.AddWithValue("$Step3", routine.Step3);
268     insertCommand.Parameters.AddWithValue("$Step4", routine.Step4);
269
270     await insertCommand.ExecuteNonQueryAsync();
271
272     transaction.Commit();
273 }
274
275 public async Task<bool> DeleteRoutineByIdAsync(int id)
276 {
277     using var connection = _dbConnection.GetDbConnection();
278     await connection.OpenAsync();
279
280     var command = connection.CreateCommand();
281     command.CommandText = @"
282 DELETE FROM Routine
283 WHERE Id = $Id
284 ";
285     command.Parameters.AddWithValue("$Id", id);
286
287     var affectedRows = await command.ExecuteNonQueryAsync();
288
289     return affectedRows > 0; // Return true if at least one row was deleted
290 }
291
292 public async Task<Routine> GetRoutineByIdAsync(int id)
293 {
294     using var connection = _dbConnection.GetDbConnection();
295     await connection.OpenAsync();
296
297     var command = connection.CreateCommand();
298     command.CommandText = @"
299 SELECT Id, Title, Description, Step1, Step2, Step3, Step4
300 FROM Routine
301 WHERE Id = $Id
302 ";
303     command.Parameters.AddWithValue("$Id", id);
304
305     using var reader = await command.ExecuteReaderAsync();
306     if (await reader.ReadAsync())
307     {
308         var routine = new Routine
309         {
310             Id = reader.GetInt32(0),
311             Title = reader.GetString(1),
312             Description = reader.GetString(2),
313             Step1 = reader.GetString(3),
314             Step2 = reader.GetString(4),
315             Step3 = reader.GetString(5),
316             Step4 = reader.GetString(6),
317         };
318         return routine;
319     }
320
321     return null; // Return null or handle it if no task is found
322 }
323
324 public async Task<List<Routine>> GetAllRoutinesAsync()
325 {
326     using var connection = _dbConnection.GetDbConnection();
327     await connection.OpenAsync();
328
329     var command = connection.CreateCommand();
330     command.CommandText = @"
331 SELECT Id, Title, Description

```

Figure 81. Source code of the DatabaseDataAccess.cs (tool: Visual Studio IDE [70], source: self)

DatabaseConnection.cs is fairly simple and is solely responsible for initial creation of the database tables as seen on figure 82.

```

1  using Microsoft.Data.Sqlite;
2
3  namespace RAMDHD.Database
4  {
5      public class DatabaseConnection
6      {
7          private readonly string _dbPath;
8
9          public DatabaseConnection(string dbPath)
10         {
11             _dbPath = dbPath;
12         }
13
14         public SQLiteConnection GetDbConnection()
15         {
16             return new SQLiteConnection($"Filename={_dbPath}");
17         }
18
19         public void InitializeDatabase()
20         {
21             using var connection = GetDbConnection();
22             connection.Open();
23
24             CreateGraphTaskTable(connection); // Create GraphTask table
25             CreateNotesTable(connection); // Create Note table
26             CreateRoutineTable(connection); // Create Routine table
27         }
28
29         private void CreateGraphTaskTable(SQLiteConnection connection)
30         {
31             var tableCommandText = @"
32             CREATE TABLE IF NOT EXISTS GraphTask (
33                 Id INTEGER PRIMARY KEY,
34                 Title TEXT NULLABLE,
35                 Activity1 TEXT NULLABLE,
36                 Activity2 TEXT NULLABLE,
37                 Activity3 TEXT NULLABLE,
38                 Activity4 TEXT NULLABLE,
39                 Procrastination INTEGER NULLABLE
40             );
41             var createTableCommand = new SQLiteCommand(tableCommandText, connection);
42             createTableCommand.ExecuteNonQuery();
43         }
44
45         private void CreateNotesTable(SQLiteConnection connection)
46         {
47             var tableCommandText = @"
48             CREATE TABLE IF NOT EXISTS Note (
49                 Id INTEGER PRIMARY KEY,
50                 Headline TEXT NULLABLE,
51                 Description TEXT NULLABLE
52             );
53             var createTableCommand = new SQLiteCommand(tableCommandText, connection);
54             createTableCommand.ExecuteNonQuery();
55         }
56         private void CreateRoutineTable(SQLiteConnection connection)
57         {
58             var tableCommandText = @"
59             CREATE TABLE IF NOT EXISTS Routine (
60                 Id INTEGER PRIMARY KEY,
61                 Title TEXT NULLABLE,
62                 Description TEXT NULLABLE,
63                 Step1 TEXT NULLABLE,
64                 Step2 TEXT NULLABLE,
65                 Step3 TEXT NULLABLE,
66                 Step4 TEXT NULLABLE
67             );
68             var createTableCommand = new SQLiteCommand(tableCommandText, connection);
69             createTableCommand.ExecuteNonQuery();
70         }
71     }
72 }
73

```

Figure 82. Source code of the DatabaseConnection.cs (tool: Visual Studio IDE [70], source: self)

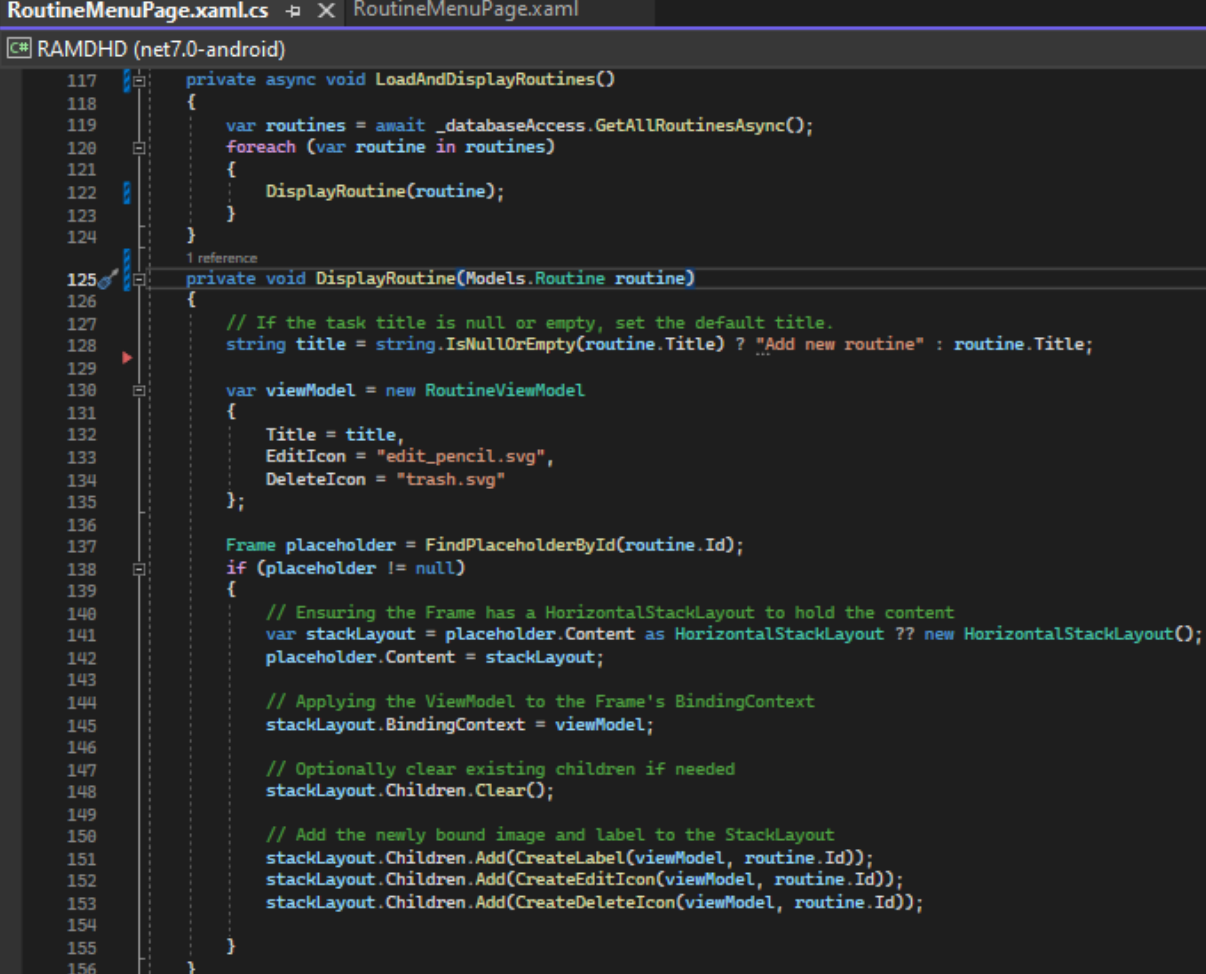
After a successful record insertion system notifies the user about a finished operation as seen on figure 83. Task is finished once the “Success” notification is displayed, although technically the function still awaits for the user to click “OK” and then navigates back to the Routine Menu Page.

```
79 private async void OnSaveButtonClickedAsync(object sender, EventArgs e)
80 {
81     // Create a new Routine object
82     var routine = new Models.Routine
83     {
84         Title = !string.IsNullOrEmpty(HeadlineEntry.Text) ? HeadlineEntry.Text : "",
85         Description = !string.IsNullOrEmpty(DescriptionEntry.Text) ? DescriptionEntry.Text : "",
86         Step1 = !string.IsNullOrEmpty(Step1Entry.Text) ? Step1Entry.Text : "",
87         Step2 = !string.IsNullOrEmpty(Step2Entry.Text) ? Step2Entry.Text : "",
88         Step3 = !string.IsNullOrEmpty(Step3Entry.Text) ? Step3Entry.Text : "",
89         Step4 = !string.IsNullOrEmpty(Step4Entry.Text) ? Step4Entry.Text : "",
90     };
91
92     // Assume _database is an instance of your DatabaseConnection with a proper DB path.
93     await _databaseAccess.InsertOrUpdateRoutineByIdAsync(routine, placeholderId);
94
95     // Show success message
96     await DisplayAlert("Success", $"Routine '{routine.Title}' saved successfully", "OK");
97
98     // Navigate to Graph Task Menu
99     await this.Navigation.PushAsync(new RoutineMenuPage());
100 }
101
```

Figure 83. Source code of the OnSaveButtonClickedAsync function in the EditRoutinePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

7.4. Completing the custom routine


Once a custom routine has been created it can be accessed from the Routine Menu Page. LoadAndDisplayRoutines function is called on screen initialization. It fetches all the routines with the databaseAccess interface method GetAllRoutinesAsync and for each of the routines it calls the DisplayRoutine function as seen on figure 84.



```
117 private async void LoadAndDisplayRoutines()
118 {
119     var routines = await _databaseAccess.GetAllRoutinesAsync();
120     foreach (var routine in routines)
121     {
122         DisplayRoutine(routine);
123     }
124 }
125 private void DisplayRoutine(Models.Routine routine)
126 {
127     // If the task title is null or empty, set the default title.
128     string title = string.IsNullOrEmpty(routine.Title) ? "Add new routine" : routine.Title;
129
130     var viewModel = new RoutineViewModel
131     {
132         Title = title,
133         EditIcon = "edit_pencil.svg",
134         DeleteIcon = "trash.svg"
135     };
136
137     Frame placeholder = FindPlaceholderById(routine.Id);
138     if (placeholder != null)
139     {
140         // Ensuring the Frame has a HorizontalStackLayout to hold the content
141         var stackLayout = placeholder.Content as HorizontalStackLayout ?? new HorizontalStackLayout();
142         placeholder.Content = stackLayout;
143
144         // Applying the ViewModel to the Frame's BindingContext
145         stackLayout.BindingContext = viewModel;
146
147         // Optionally clear existing children if needed
148         stackLayout.Children.Clear();
149
150         // Add the newly bound image and label to the StackLayout
151         stackLayout.Children.Add(CreateLabel(viewModel, routine.Id));
152         stackLayout.Children.Add(CreateEditIcon(viewModel, routine.Id));
153         stackLayout.Children.Add(CreateDeleteIcon(viewModel, routine.Id));
154     }
155 }
156 }
```

Figure 84. Source code of LoadAndDisplayRoutines and DisplayRoutine functions in the RoutineMenuPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

RoutineViewModel contains only the necessary properties that can be displayed in the Routine Menu Page. These are the routine's title, editIcon and deleteIcon as seen on figure 85.



```

1  using System.ComponentModel;
2
3  namespace RAMDHD.ViewModels.Routine
4  {
5      public class RoutineViewModel
6      {
7          public event PropertyChangedEventHandler PropertyChanged;
8
9          private string _title = "Add new routine";
10         private string _editIcon;
11         private string _deleteIcon;
12
13         public string Title
14         {
15             get => _title;
16             set
17             {
18                 if (_title != value)
19                 {
20                     _title = value;
21                     OnPropertyChanged(nameof(Title));
22                 }
23             }
24         }
25
26         public string EditIcon
27         {
28             get => _editIcon;
29             set
30             {
31                 if (_editIcon != value)
32                 {
33                     _editIcon = value;
34                     OnPropertyChanged(nameof(EditIcon));
35                 }
36             }
37         }
38
39         public string DeleteIcon
40         {
41             get => _deleteIcon;
42             set
43             {
44                 if (_deleteIcon != value)
45                 {
46                     _deleteIcon = value;
47                     OnPropertyChanged(nameof(DeleteIcon));
48                 }
49             }
50         }
51
52         protected virtual void OnPropertyChanged(string propertyName)
53         {
54             PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));
55         }
56     }

```

Figure 85. Source code of the RoutineViewModel.cs (tool: Visual Studio IDE [70], source: self)

Once all the routines are fetched they can be accessed by clicking on their label (presented in black font). Displaying the routine technically means displaying a RoutineTemplatePage as seen on figure 86.

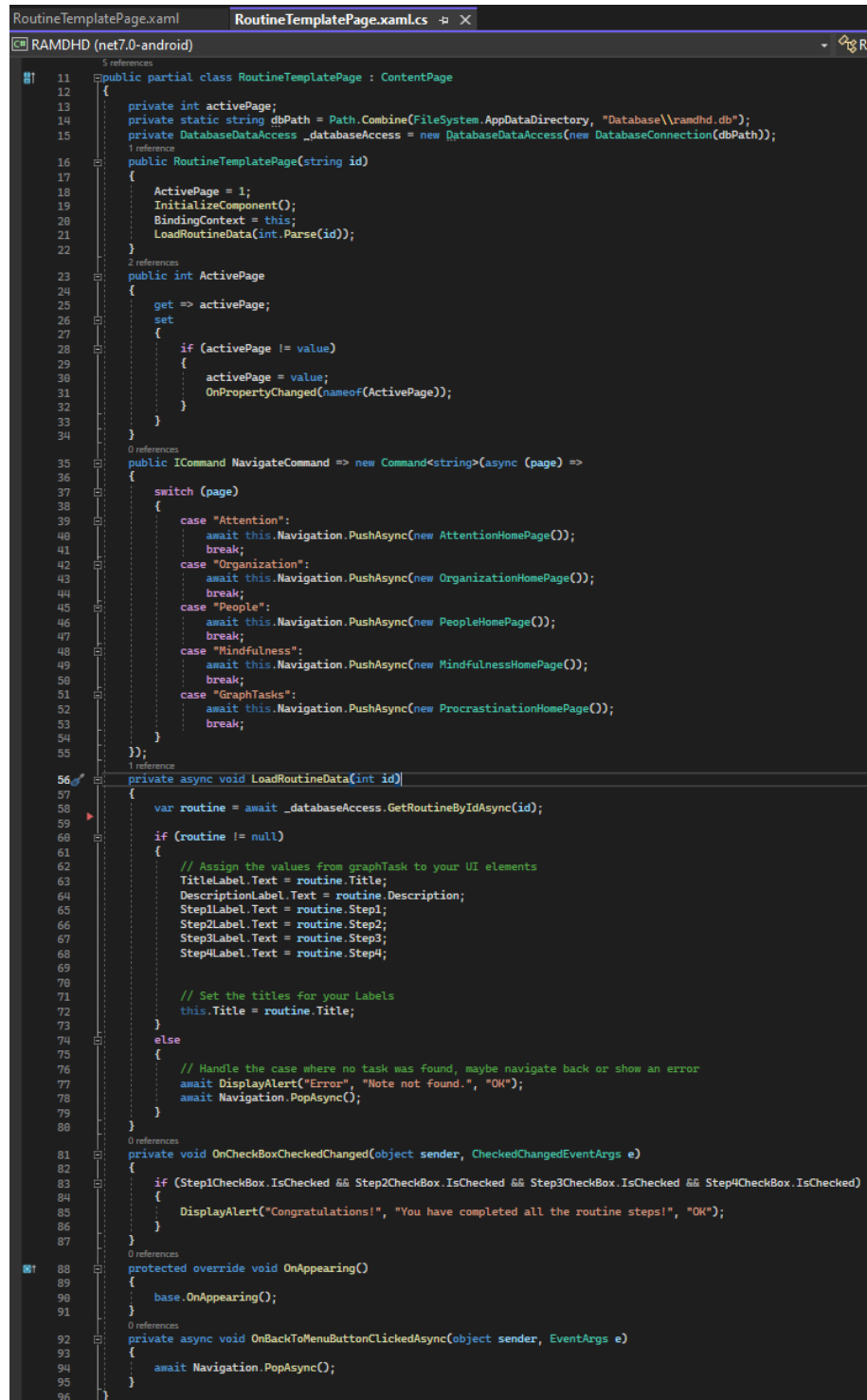
```

1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
3    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4    xmlns:local="clr-namespace:RAMDHD.Views.Components"
5    x:Class="RAMDHD.Views.MainScreens.Organization.Routine.RoutineTemplatePage"
6    Title="TitleLabel"
7    BackgroundColor="{StaticResource White}">
8    <AbsoluteLayout HorizontalOptions="FillAndExpand" VerticalOptions="FillAndExpand">
9      <Label
10        x:Name="TitleLabel"
11        Text="title"
12        FontFamily="ReemKufiFunRegular"
13        FontSize="20"
14        TextColor="{StaticResource SharcBlack}"
15        TextDecorations="Underline"
16        AbsoluteLayout.LayoutBounds="0.4, 0.05, -0.5, -1"
17        AbsoluteLayout.LayoutFlags="PositionProportional"
18        WidthRequest="200"
19        HorizontalOptions="Center"
20        VerticalOptions="Center"
21        LineBreakMode="WordWrap">
22        <Label.GestureRecognizers>
23          <TapGestureRecognizer />
24        </Label.GestureRecognizers>
25      </Label>
26
27      <Label
28        x:Name="DescriptionLabel"
29        Text="description"
30        FontFamily="ReemKufiFunRegular"
31        FontSize="16"
32        TextColor="{StaticResource SharcBlack}"
33        AbsoluteLayout.LayoutBounds="0.4, 0.15, -0.5, -1"
34        AbsoluteLayout.LayoutFlags="PositionProportional"
35        WidthRequest="200"
36        HorizontalOptions="Center"
37        VerticalOptions="Center"
38        LineBreakMode="WordWrap">
39        <Label.GestureRecognizers>
40          <TapGestureRecognizer />
41        </Label.GestureRecognizers>
42      </Label>
43
44      <StackLayout
45        Padding="20"
46        AbsoluteLayout.LayoutBounds="0, 0.4, -1, -1"
47        AbsoluteLayout.LayoutFlags="PositionProportional"
48        Spacing="10">
49        <!-- Routine Step 1 -->
50        <StackLayout Orientation="Horizontal" VerticalOptions="Center">
51          <CheckBox x:Name="Step1CheckBox" Color="{StaticResource LightGreen}" CheckedChanged="OnCheckBoxCheckedChanged"/>
52          <Label x:Name="Step1Label" Text="Routine Step 1"
53            TextColor="{StaticResource SharcBlack}" FontFamily="ReemKufiFunRegular" FontSize="18"
54            VerticalOptions="Center" />
55        </StackLayout>
56        <!-- Routine Step 2 -->
57        <StackLayout Orientation="Horizontal" VerticalOptions="Center">
58          <CheckBox x:Name="Step2CheckBox" Color="{StaticResource LightGreen}" CheckedChanged="OnCheckBoxCheckedChanged"/>
59          <Label x:Name="Step2Label" Text="Routine Step 2"
60            TextColor="{StaticResource SharcBlack}" FontFamily="ReemKufiFunRegular" FontSize="18"
61            VerticalOptions="Center" />
62        </StackLayout>
63        <!-- Routine Step 3 -->
64        <StackLayout Orientation="Horizontal" VerticalOptions="Center">
65          <CheckBox x:Name="Step3CheckBox" Color="{StaticResource LightGreen}" CheckedChanged="OnCheckBoxCheckedChanged"/>
66          <Label x:Name="Step3Label" Text="Routine Step 3"
67            TextColor="{StaticResource SharcBlack}" FontFamily="ReemKufiFunRegular" FontSize="18"
68            VerticalOptions="Center" />
69        </StackLayout>
70        <!-- Routine Step 4 -->
71        <StackLayout Orientation="Horizontal" VerticalOptions="Center">
72          <CheckBox x:Name="Step4CheckBox" Color="{StaticResource LightGreen}" CheckedChanged="OnCheckBoxCheckedChanged"/>
73          <Label x:Name="Step4Label" Text="Routine Step 4"
74            TextColor="{StaticResource SharcBlack}" FontFamily="ReemKufiFunRegular" FontSize="18"
75            VerticalOptions="Center" />
76        </StackLayout>
77      </StackLayout>
78
79      <!-- Back to Menu Button -->
80      <Button
81        Text="BACK TO MENU"
82        WidthRequest="200" HeightRequest="50"
83        AbsoluteLayout.LayoutBounds="0.5, 0.8, 0, 0"
84        AbsoluteLayout.LayoutFlags="PositionProportional"
85        BackgroundColor="{StaticResource Secondary}" TextColor="{StaticResource SharcBlack}" BorderWidth="1"
86        HorizontalOptions="Center" VerticalOptions="Center"
87        Clicked="OnBackToMenuButtonClickedAsync"
88        IsEnabled="True"
89      />
90
91      <!-- Bottom Navigation -->
92      <local:BottomNavigationView AbsoluteLayout.LayoutBounds="0.5, 1, 1, AutoSize"
93        AbsoluteLayout.LayoutFlags="XProportional, WidthProportional, PositionProportional"
94        ActivePage="{Binding ActivePage}"
95        NavigateCommand="{Binding NavigateCommand}" />
96    </AbsoluteLayout>

```

Figure 86. Source code of the RoutineTemplatePage.xaml (tool: Visual Studio IDE [70], source: self)

This template page fetches the given routine data and displays it dynamically on initialization as seen on figure 87. LoadRoutineData calls the databaseAccess interface method GetRoutineByIdAsync and populates the screen with the given routine data (based on its ID). Completing the routine requires marking all the green checkboxes as defined by the OnCheckBoxChanged function. If all 4 checkboxes are marked the system notifies the user about a successful routine completion. This concludes the second task.



```

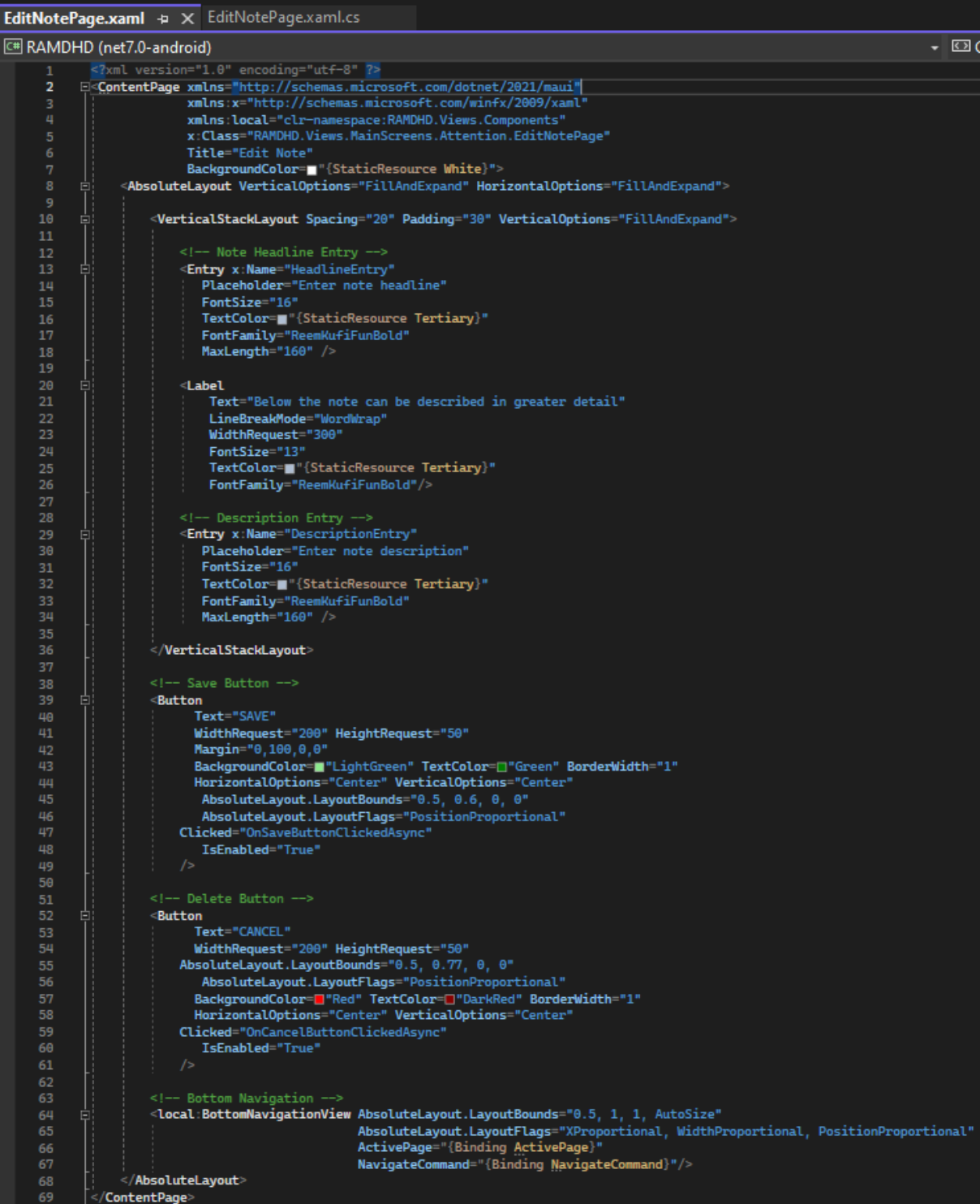
11 public partial class RoutineTemplatePage : ContentPage
12 {
13     private int activePage;
14     private static string dbPath = Path.Combine(FileSystem.AppDataDirectory, "Database\\ramdhd.db");
15     private DatabaseDataAccess _databaseAccess = new DatabaseDataAccess(new DatabaseConnection(dbPath));
16     public RoutineTemplatePage(string id)
17     {
18         ActivePage = 1;
19         InitializeComponent();
20         BindingContext = this;
21         LoadRoutineData(int.Parse(id));
22     }
23     public int ActivePage
24     {
25         get => activePage;
26         set
27         {
28             if (activePage != value)
29             {
30                 activePage = value;
31                 OnPropertyChanged(nameof(ActivePage));
32             }
33         }
34     }
35     public ICommand NavigateCommand => new Command<string>(async (page) =>
36     {
37         switch (page)
38         {
39             case "Attention":
40                 await this.Navigation.PushAsync(new AttentionHomePage());
41                 break;
42             case "Organization":
43                 await this.Navigation.PushAsync(new OrganizationHomePage());
44                 break;
45             case "People":
46                 await this.Navigation.PushAsync(new PeopleHomePage());
47                 break;
48             case "Mindfulness":
49                 await this.Navigation.PushAsync(new MindfulnessHomePage());
50                 break;
51             case "GraphTasks":
52                 await this.Navigation.PushAsync(new ProcrastinationHomePage());
53                 break;
54         }
55     });
56     private async void LoadRoutineData(int id)
57     {
58         var routine = await _databaseAccess.GetRoutineByIdAsync(id);
59         if (routine != null)
60         {
61             // Assign the values from graphTask to your UI elements
62             TitleLabel.Text = routine.Title;
63             DescriptionLabel.Text = routine.Description;
64             Step1Label.Text = routine.Step1;
65             Step2Label.Text = routine.Step2;
66             Step3Label.Text = routine.Step3;
67             Step4Label.Text = routine.Step4;
68             // Set the titles for your Labels
69             this.Title = routine.Title;
70         }
71         else
72         {
73             // Handle the case where no task was found, maybe navigate back or show an error
74             await DisplayAlert("Error", "Note not found.", "OK");
75             await Navigation.PopAsync();
76         }
77     }
78     private void OnCheckBoxCheckedChanged(object sender, CheckedChangedEventArgs e)
79     {
80         if (Step1CheckBox.IsChecked && Step2CheckBox.IsChecked && Step3CheckBox.IsChecked && Step4CheckBox.IsChecked)
81         {
82             DisplayAlert("Congratulations!", "You have completed all the routine steps!", "OK");
83         }
84     }
85     protected override void OnAppearing()
86     {
87         base.OnAppearing();
88     }
89     private async void OnBackToMenuButtonClickedAsync(object sender, EventArgs e)
90     {
91         await Navigation.PopAsync();
92     }
93 }

```

Figure 87. Source code of the RoutineTemplatePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

7.5. Creating custom note

Creating a custom note is technically very similar to creating a routine, hence it is sufficient to showcase the important differences between these two. As presented on figure 88, the EditNotePage.xaml contains the headline entry and note's description.



```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
3             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4             xmlns:local="clr-namespace:RAMDHD.Views.Components"
5             x:Class="RAMDHD.Views.MainScreens.Attention.EditNotePage"
6             Title="Edit Note"
7             BackgroundColor="{StaticResource White}">
8     <AbsoluteLayout VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand">
9
10         <VerticalStackLayout Spacing="20" Padding="30" VerticalOptions="FillAndExpand">
11
12             <!-- Note Headline Entry -->
13             <Entry x:Name="HeadlineEntry"
14                 Placeholder="Enter note headline"
15                 FontSize="16"
16                 TextColor="{StaticResource Tertiary}"
17                 FontFamily="ReemKufiFunBold"
18                 MaxLength="160" />
19
20             <Label
21                 Text="Below the note can be described in greater detail"
22                 LineBreakMode="WordWrap"
23                 WidthRequest="300"
24                 FontSize="13"
25                 TextColor="{StaticResource Tertiary}"
26                 FontFamily="ReemKufiFunBold"/>
27
28             <!-- Description Entry -->
29             <Entry x:Name="DescriptionEntry"
30                 Placeholder="Enter note description"
31                 FontSize="16"
32                 TextColor="{StaticResource Tertiary}"
33                 FontFamily="ReemKufiFunBold"
34                 MaxLength="160" />
35         </VerticalStackLayout>
36
37         <!-- Save Button -->
38         <Button
39             Text="SAVE"
40             WidthRequest="200" HeightRequest="50"
41             Margin="0, 100, 0, 0"
42             BackgroundColor="LightGreen" TextColor="Green" BorderWidth="1"
43             HorizontalOptions="Center" VerticalOptions="Center"
44             AbsoluteLayout.LayoutBounds="0.5, 0.6, 0, 0"
45             AbsoluteLayout.LayoutFlags="PositionProportional"
46             Clicked="OnSaveButtonClickedAsync"
47             IsEnabled="True"
48             />
49
50         <!-- Delete Button -->
51         <Button
52             Text="CANCEL"
53             WidthRequest="200" HeightRequest="50"
54             AbsoluteLayout.LayoutBounds="0.5, 0.77, 0, 0"
55             AbsoluteLayout.LayoutFlags="PositionProportional"
56             BackgroundColor="Red" TextColor="DarkRed" BorderWidth="1"
57             HorizontalOptions="Center" VerticalOptions="Center"
58             Clicked="OnCancelButtonClickedAsync"
59             IsEnabled="True"
60             />
61
62         <!-- Bottom Navigation -->
63         <local:BottomNavigationView AbsoluteLayout.LayoutBounds="0.5, 1, 1, AutoSize"
64             AbsoluteLayout.LayoutFlags="XProportional, WidthProportional, PositionProportional"
65             ActivePage="{Binding ActivePage}"
66             NavigateCommand="{Binding NavigateCommand}" />
67     </AbsoluteLayout>
68 </ContentPage>
```

Figure 88. Source code of the EditNotePage.xaml (tool: Visual Studio IDE [70], source: self)

OnSaveButtonClickedAsync function is responsible for creating a note model object as seen on figure 89.

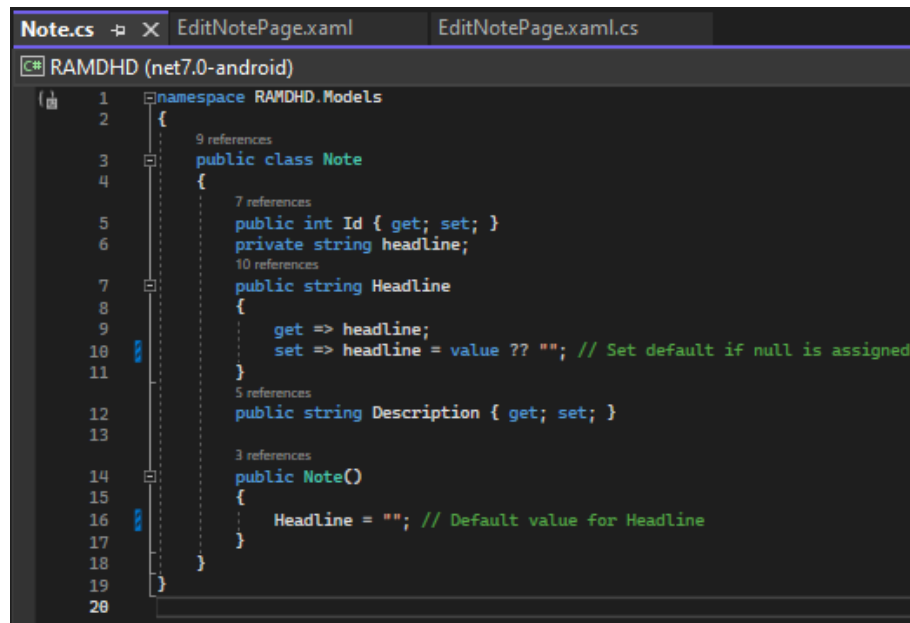
```

EditNotePage.xaml EditNotePage.xaml.cs
RAMDHD (net7.0-android)
4 using RAMDHD.Views.MainScreens.GraphTask;
5 using RAMDHD.Views.MainScreens.Organization;
6 using RAMDHD.Views.MainScreens.People;
7 using System.Windows.Input;
8 using System.ComponentModel;
9
10 namespace RAMDHD.Views.MainScreens.Attention;
11
12 public partial class EditNotePage : ContentPage, INotifyPropertyChanged
13 {
14     private int placeholderId;
15     private int activePage;
16     private static string dbPath = Path.Combine(FileSystem.AppDataDirectory, "Database\\ramdhd.db");
17     private DatabaseDataAccess _databaseAccess = new DatabaseDataAccess(new DatabaseConnection(dbPath));
18
19     public EditNotePage(string placeholderIdString)
20     {
21         InitializeComponent();
22         if (int.TryParse(placeholderIdString, out placeholderId))
23         {
24             ActivePage = 0;
25             BindingContext = this;
26         }
27         else
28         {
29             // Handle the case where the ID is not valid
30         }
31     }
32
33     public int ActivePage
34     {
35         get => activePage;
36         set
37         {
38             if (ActivePage != value)
39             {
40                 activePage = value;
41                 OnPropertyChanged(nameof(ActivePage));
42             }
43         }
44     }
45
46     public ICommand NavigateCommand => new Command<string>(async (page) =>
47     {
48         switch (page)
49         {
50             case "Attention":
51                 await this.Navigation.PushAsync(new AttentionHomePage());
52                 break;
53             case "Organization":
54                 await this.Navigation.PushAsync(new OrganizationHomePage());
55                 break;
56             case "People":
57                 await this.Navigation.PushAsync(new PeopleHomePage());
58                 break;
59             case "Mindfulness":
60                 await this.Navigation.PushAsync(new MindfulnessHomePage());
61                 break;
62             case "GraphTasks":
63                 await this.Navigation.PushAsync(new ProcrastinationHomePage());
64                 break;
65         }
66     });
67
68     private async void OnSaveButtonClickedAsync(object sender, EventArgs e)
69     {
70         // Create a new Note object
71         var note = new Models.Note
72         {
73             Headline = !string.IsNullOrWhiteSpace(HeadlineEntry.Text) ? HeadlineEntry.Text : "",
74             Description = !string.IsNullOrWhiteSpace(DescriptionEntry.Text) ? DescriptionEntry.Text : "",
75         };
76
77         // Assume _database is an instance of your DatabaseConnection with a proper DB path.
78         await _databaseAccess.InsertOrUpdateNoteByIdAsync(note, placeholderId);
79
80         // Show success message
81         await DisplayAlert("Success", $"Note '{note.Headline}' saved successfully", "OK");
82
83         await this.Navigation.PushAsync(new NotesMenuPage());
84     }
85
86     private async void OnCancelButtonClickedAsync(object sender, EventArgs e)
87     {
88         Console.WriteLine("OnCancelButtonClicked");
89         await Navigation.PopAsync();
90     }
91 }

```

Figure 89. Source code of the EditNotePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Note model is presented on figure 90. NoteViewModel, containing the necessary information to be presented on the Notes Menu page is shown on figure 91.

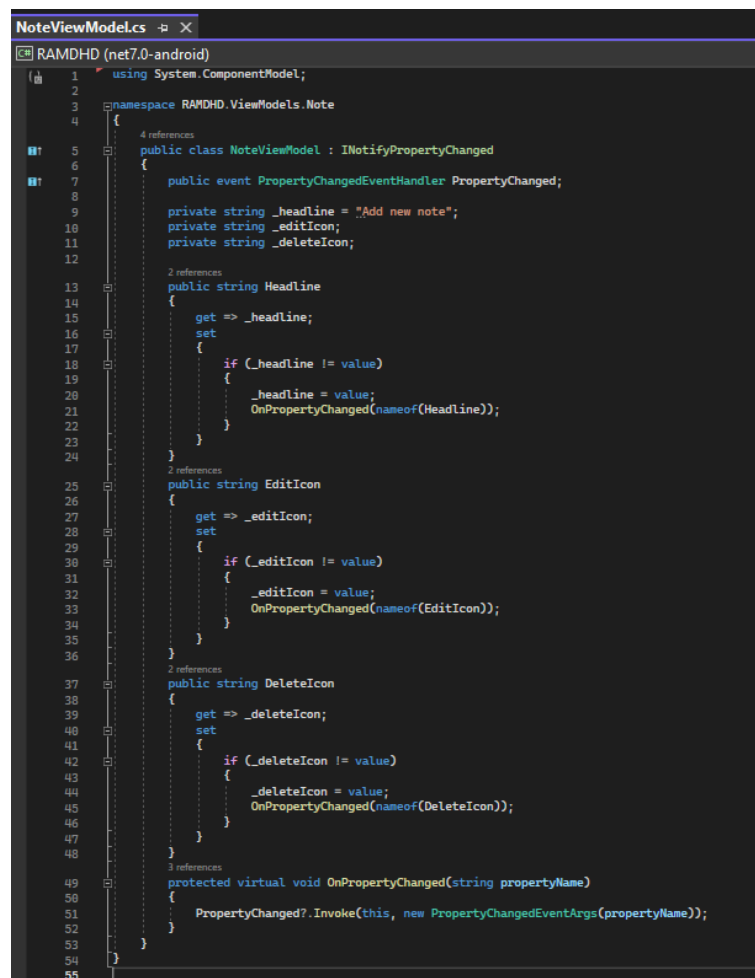


```

1 namespace RAMDHD.Models
2 {
3     public class Note
4     {
5         public int Id { get; set; }
6         private string headline;
7         public string Headline
8         {
9             get => headline;
10            set => headline = value ?? ""; // Set default if null is assigned
11        }
12        public string Description { get; set; }
13
14        public Note()
15        {
16            Headline = ""; // Default value for Headline
17        }
18    }
19 }
20

```

Figure 90. Source code of the Note.cs model (tool: Visual Studio IDE [70], source: self)



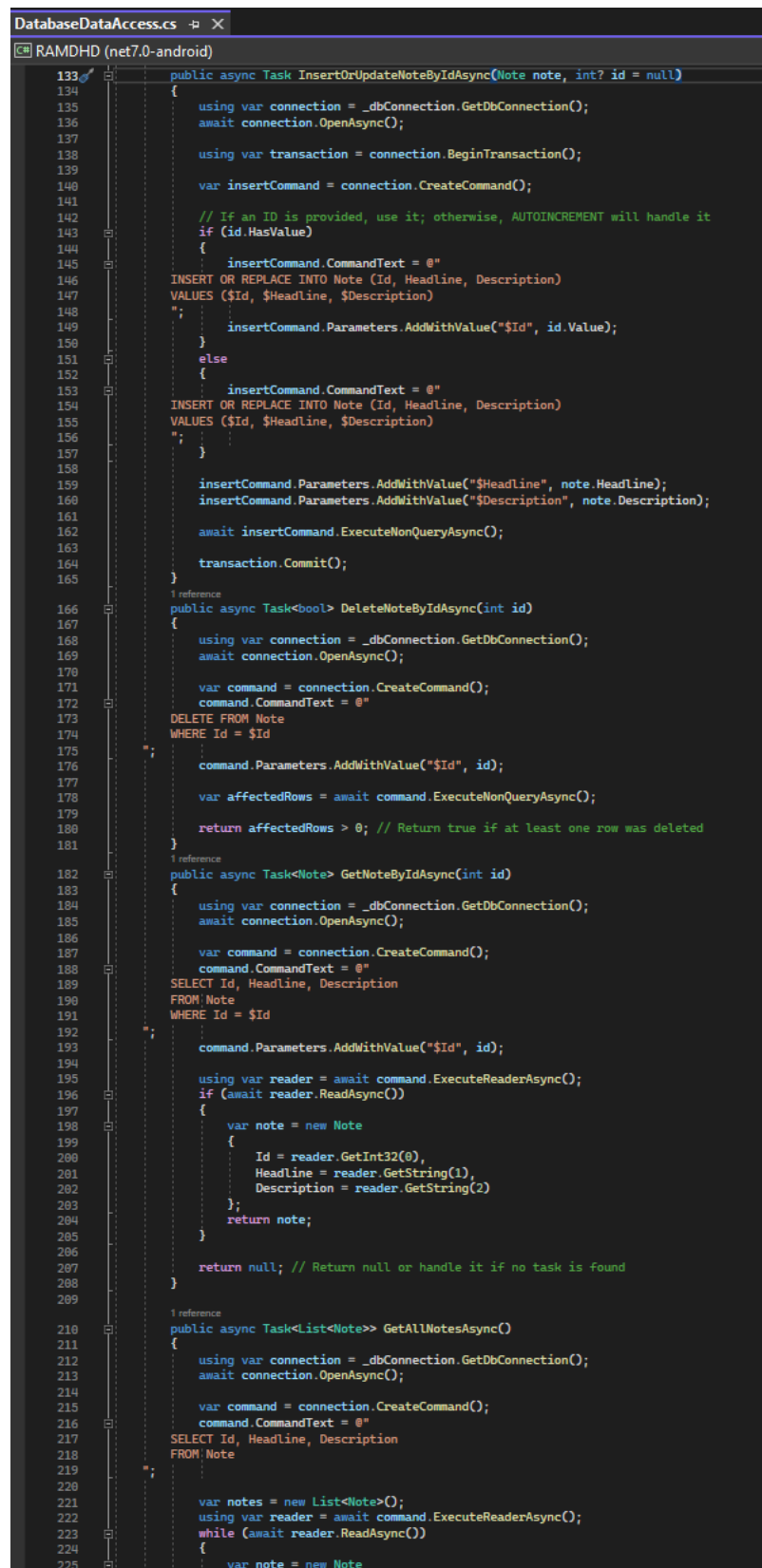
```

1 using System.ComponentModel;
2
3 namespace RAMDHD.ViewModels.Note
4 {
5     public class NoteViewModel : INotifyPropertyChanged
6     {
7         public event PropertyChangedEventHandler PropertyChanged;
8
9         private string _headline = "Add new note";
10        private string _editIcon;
11        private string _deleteIcon;
12
13        public string Headline
14        {
15            get => _headline;
16            set
17            {
18                if (_headline != value)
19                {
20                    _headline = value;
21                    OnPropertyChanged(nameof(Headline));
22                }
23            }
24        }
25        public string EditIcon
26        {
27            get => _editIcon;
28            set
29            {
30                if (_editIcon != value)
31                {
32                    _editIcon = value;
33                    OnPropertyChanged(nameof(EditIcon));
34                }
35            }
36        }
37        public string DeleteIcon
38        {
39            get => _deleteIcon;
40            set
41            {
42                if (_deleteIcon != value)
43                {
44                    _deleteIcon = value;
45                    OnPropertyChanged(nameof(DeleteIcon));
46                }
47            }
48        }
49        protected virtual void OnPropertyChanged(string propertyName)
50        {
51            PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));
52        }
53    }
54 }
55

```

Figure 91. Source code of the NoteViewModel.cs (tool: Visual Studio IDE [70], source: self)

Finally, the DatabaseDataAccess function responsible for handling the creation of a new note is presented on figure 92.



```

DatabaseDataAccess.cs
RAMDHD (net7.0-android)

133 public async Task InsertOrUpdateNoteByIdAsync(Note note, int? id = null)
134 {
135     using var connection = _dbConnection.GetDbConnection();
136     await connection.OpenAsync();
137
138     using var transaction = connection.BeginTransaction();
139
140     var insertCommand = connection.CreateCommand();
141
142     // If an ID is provided, use it; otherwise, AUTOINCREMENT will handle it
143     if (id.HasValue)
144     {
145         insertCommand.CommandText = @"
146 INSERT OR REPLACE INTO Note (Id, Headline, Description)
147 VALUES ($Id, $Headline, $Description)
148 ";
149         insertCommand.Parameters.AddWithValue("$Id", id.Value);
150     }
151     else
152     {
153         insertCommand.CommandText = @"
154 INSERT OR REPLACE INTO Note (Id, Headline, Description)
155 VALUES ($Id, $Headline, $Description)
156 ";
157     }
158
159     insertCommand.Parameters.AddWithValue("$Headline", note.Headline);
160     insertCommand.Parameters.AddWithValue("$Description", note.Description);
161
162     await insertCommand.ExecuteNonQueryAsync();
163
164     transaction.Commit();
165 }
166
167 1 reference
168 public async Task<bool> DeleteNoteByIdAsync(int id)
169 {
170     using var connection = _dbConnection.GetDbConnection();
171     await connection.OpenAsync();
172
173     var command = connection.CreateCommand();
174     command.CommandText = @"
175 DELETE FROM Note
176 WHERE Id = $Id
177 ";
178     command.Parameters.AddWithValue("$Id", id);
179
180     var affectedRows = await command.ExecuteNonQueryAsync();
181
182     return affectedRows > 0; // Return true if at least one row was deleted
183 }
184
185 1 reference
186 public async Task<Note> GetNoteByIdAsync(int id)
187 {
188     using var connection = _dbConnection.GetDbConnection();
189     await connection.OpenAsync();
190
191     var command = connection.CreateCommand();
192     command.CommandText = @"
193 SELECT Id, Headline, Description
194 FROM Note
195 WHERE Id = $Id
196 ";
197     command.Parameters.AddWithValue("$Id", id);
198
199     using var reader = await command.ExecuteReaderAsync();
200     if (await reader.ReadAsync())
201     {
202         var note = new Note
203         {
204             Id = reader.GetInt32(0),
205             Headline = reader.GetString(1),
206             Description = reader.GetString(2)
207         };
208         return note;
209     }
210
211     return null; // Return null or handle it if no task is found
212 }
213
214 1 reference
215 public async Task<List<Note>> GetAllNotesAsync()
216 {
217     using var connection = _dbConnection.GetDbConnection();
218     await connection.OpenAsync();
219
220     var command = connection.CreateCommand();
221     command.CommandText = @"
222 SELECT Id, Headline, Description
223 FROM Note
224 ";
225
226     var notes = new List<Note>();
227     using var reader = await command.ExecuteReaderAsync();
228     while (await reader.ReadAsync())
229     {
230         var note = new Note

```

Figure 92. Source code of the DatabaseDataAccess.cs (tool: Visual Studio IDE [70], source: self)

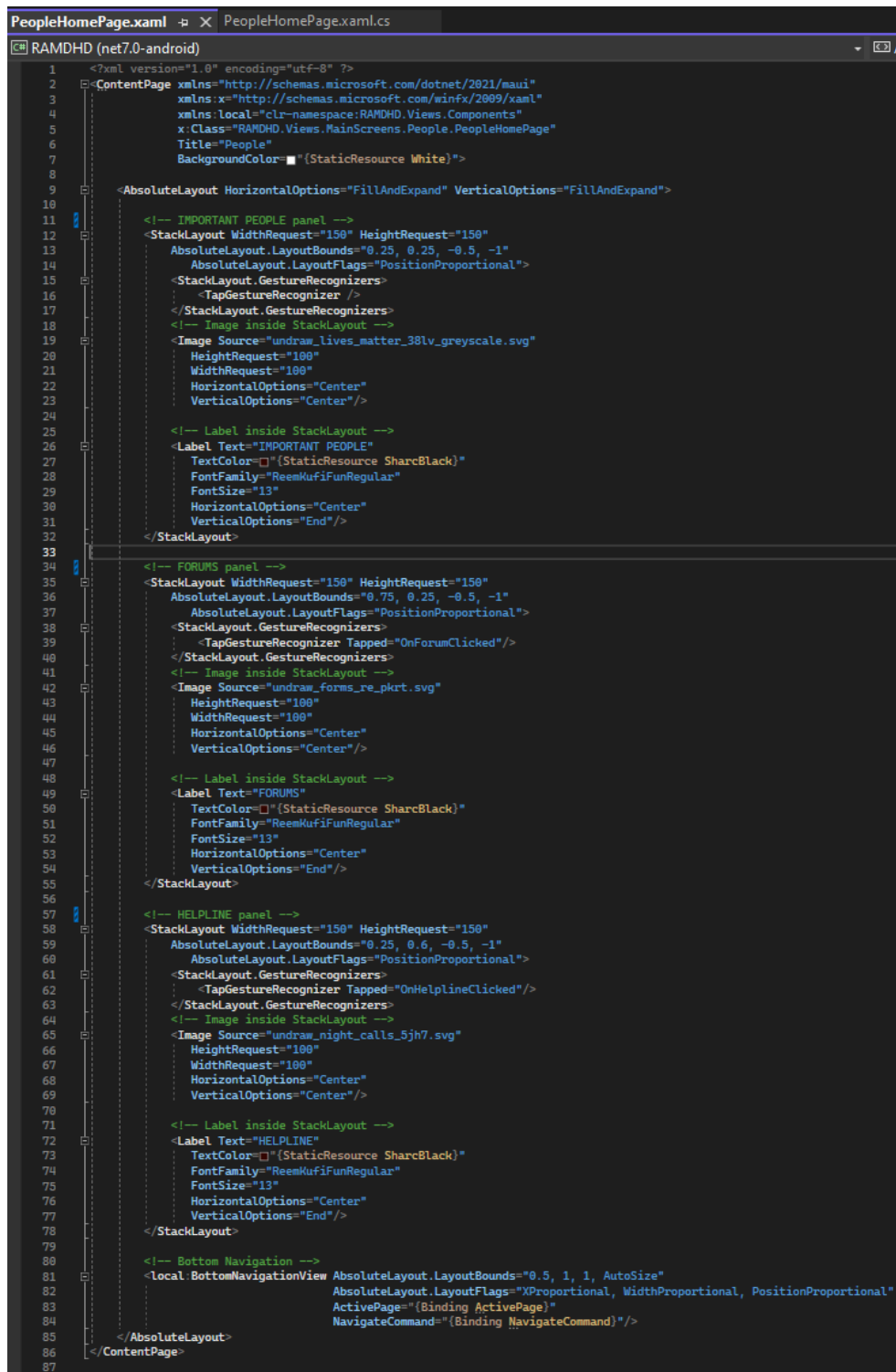
The task is completed once the `OnSaveButtonClickedAsync` function displays a notification about a successful not creation as seen on figure 93.

```
66 private async void OnSaveButtonClickedAsync(object sender, EventArgs e)
67 {
68     // Create a new Note object
69     var note = new Models.Note
70     {
71         Headline = !string.IsNullOrEmpty(HeadlineEntry.Text) ? HeadlineEntry.Text : "",
72         Description = !string.IsNullOrEmpty(DescriptionEntry.Text) ? DescriptionEntry.Text : "",
73     };
74
75     // Assume _database is an instance of your DatabaseConnection with a proper DB path.
76     await _databaseAccess.InsertOrUpdateNoteByIdAsync(note, placeholderId);
77
78     // Show success message
79     await DisplayAlert("Success", $"Note '{note.Headline}' saved successfully", "OK");
80
81     await this.Navigation.PushAsync(new NotesMenuPage());
82 }
```

Figure 93. Source code of the `OnSaveButtonClickedAsync` function in the `EditNotePage.xaml.cs` (tool: Visual Studio IDE [70], source: self)

7.6. Visiting one of the forums

To visit one of the forums the user first has to navigate to the forums page included within the People module as seen on figure 94. To navigate to the Forums Page the user clicks the “FORUMS” panel triggering the OnForumClicked function.



```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
3             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4             xmlns:local="clr-namespace:RAMDHD.Views.Components"
5             x:Class="RAMDHD.Views.MainScreens.People.PeopleHomePage"
6             Title="People"
7             BackgroundColor="{StaticResource White}">
8
9     <AbsoluteLayout HorizontalOptions="FillAndExpand" VerticalOptions="FillAndExpand">
10
11         <!-- IMPORTANT PEOPLE panel -->
12         <StackLayout WidthRequest="150" HeightRequest="150"
13                     AbsoluteLayout.LayoutBounds="0.25, 0.25, -0.5, -1"
14                     AbsoluteLayout.LayoutFlags="PositionProportional">
15             <StackLayout.GestureRecognizers>
16                 <TapGestureRecognizer />
17             </StackLayout.GestureRecognizers>
18             <!-- Image inside StackLayout -->
19             <Image Source="undraw_lives_matter_38lv_greyscale.svg"
20                    HeightRequest="100"
21                    WidthRequest="100"
22                    HorizontalOptions="Center"
23                    VerticalOptions="Center"/>
24
25             <!-- Label inside StackLayout -->
26             <Label Text="IMPORTANT PEOPLE"
27                    TextColor="{StaticResource ShareBlack}"
28                    FontFamily="ReemKufiFunRegular"
29                    FontSize="13"
30                    HorizontalOptions="Center"
31                    VerticalOptions="End"/>
32         </StackLayout>
33
34         <!-- FORUMS panel -->
35         <StackLayout WidthRequest="150" HeightRequest="150"
36                     AbsoluteLayout.LayoutBounds="0.75, 0.25, -0.5, -1"
37                     AbsoluteLayout.LayoutFlags="PositionProportional">
38             <StackLayout.GestureRecognizers>
39                 <TapGestureRecognizer Tapped="OnForumClicked"/>
40             </StackLayout.GestureRecognizers>
41             <!-- Image inside StackLayout -->
42             <Image Source="undraw_forms_re_pkrt.svg"
43                    HeightRequest="100"
44                    WidthRequest="100"
45                    HorizontalOptions="Center"
46                    VerticalOptions="Center"/>
47
48             <!-- Label inside StackLayout -->
49             <Label Text="FORUMS"
50                    TextColor="{StaticResource ShareBlack}"
51                    FontFamily="ReemKufiFunRegular"
52                    FontSize="13"
53                    HorizontalOptions="Center"
54                    VerticalOptions="End"/>
55         </StackLayout>
56
57         <!-- HELPLINE panel -->
58         <StackLayout WidthRequest="150" HeightRequest="150"
59                     AbsoluteLayout.LayoutBounds="0.25, 0.6, -0.5, -1"
60                     AbsoluteLayout.LayoutFlags="PositionProportional">
61             <StackLayout.GestureRecognizers>
62                 <TapGestureRecognizer Tapped="OnHelplineClicked"/>
63             </StackLayout.GestureRecognizers>
64             <!-- Image inside StackLayout -->
65             <Image Source="undraw_night_calls_5jh7.svg"
66                    HeightRequest="100"
67                    WidthRequest="100"
68                    HorizontalOptions="Center"
69                    VerticalOptions="Center"/>
70
71             <!-- Label inside StackLayout -->
72             <Label Text="HELPLINE"
73                    TextColor="{StaticResource ShareBlack}"
74                    FontFamily="ReemKufiFunRegular"
75                    FontSize="13"
76                    HorizontalOptions="Center"
77                    VerticalOptions="End"/>
78         </StackLayout>
79
80         <!-- Bottom Navigation -->
81         <local:BottomNavigationView AbsoluteLayout.LayoutBounds="0.5, 1, 1, AutoSize"
82                                   AbsoluteLayout.LayoutFlags="XProportional, WidthProportional, PositionProportional"
83                                   ActivePage="{Binding ActivePage}"
84                                   NavigateCommand="{Binding NavigateCommand}"/>
85     </AbsoluteLayout>
86 </ContentPage>
87
```

Figure 94. Source code of the PeopleHomePage.xaml (tool: Visual Studio IDE [70], source: self)

Navigation within the People module is presented on figure 95.

```

1  using RAMDHD.Views.MainScreens.Attention;
2  using RAMDHD.Views.MainScreens.Mindfulness;
3  using RAMDHD.Views.MainScreens.GraphTask;
4  using RAMDHD.Views.MainScreens.Organization;
5  using RAMDHD.Views.MainScreens.People.Forum;
6  using RAMDHD.Views.MainScreens.People.Helpline;
7  using System.Windows.Input;
8
9  namespace RAMDHD.Views.MainScreens.People
10 {
11     public partial class PeopleHomePage : ContentPage
12     {
13         private int activePage;
14         public PeopleHomePage()
15         {
16             ActivePage = 2;
17             InitializeComponent();
18             BindingContext = this;
19         }
20         public int ActivePage
21         {
22             get => activePage;
23             set
24             {
25                 if (activePage != value)
26                 {
27                     activePage = value;
28                     OnPropertyChanged(nameof(ActivePage));
29                 }
30             }
31         }
32         public ICommand NavigateCommand => new Command<string>(async (page) =>
33         {
34             switch (page)
35             {
36                 case "Attention":
37                     await this.Navigation.PushAsync(new AttentionHomePage());
38                     break;
39                 case "Organization":
40                     await this.Navigation.PushAsync(new OrganizationHomePage());
41                     break;
42                 case "People":
43                     await this.Navigation.PushAsync(new PeopleHomePage());
44                     break;
45                 case "Mindfulness":
46                     await this.Navigation.PushAsync(new MindfulnessHomePage());
47                     break;
48                 case "GraphTasks":
49                     await this.Navigation.PushAsync(new ProcrastinationHomePage());
50                     break;
51             }
52         });
53         private async void OnForumClicked(object sender, EventArgs e)
54         {
55             await this.Navigation.PushAsync(new ForumPage());
56         }
57         private async void OnHelplineClicked(object sender, EventArgs e)
58         {
59             await this.Navigation.PushAsync(new HelplinePage());
60         }
61     }
62 }
63

```

Figure 95. Source code of the PeopleHomePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Once on the Forums Page the user has a list of forums to visit as seen on figure 96. By clicking on the "internet.svg" icon the user is navigated to the forum's website.

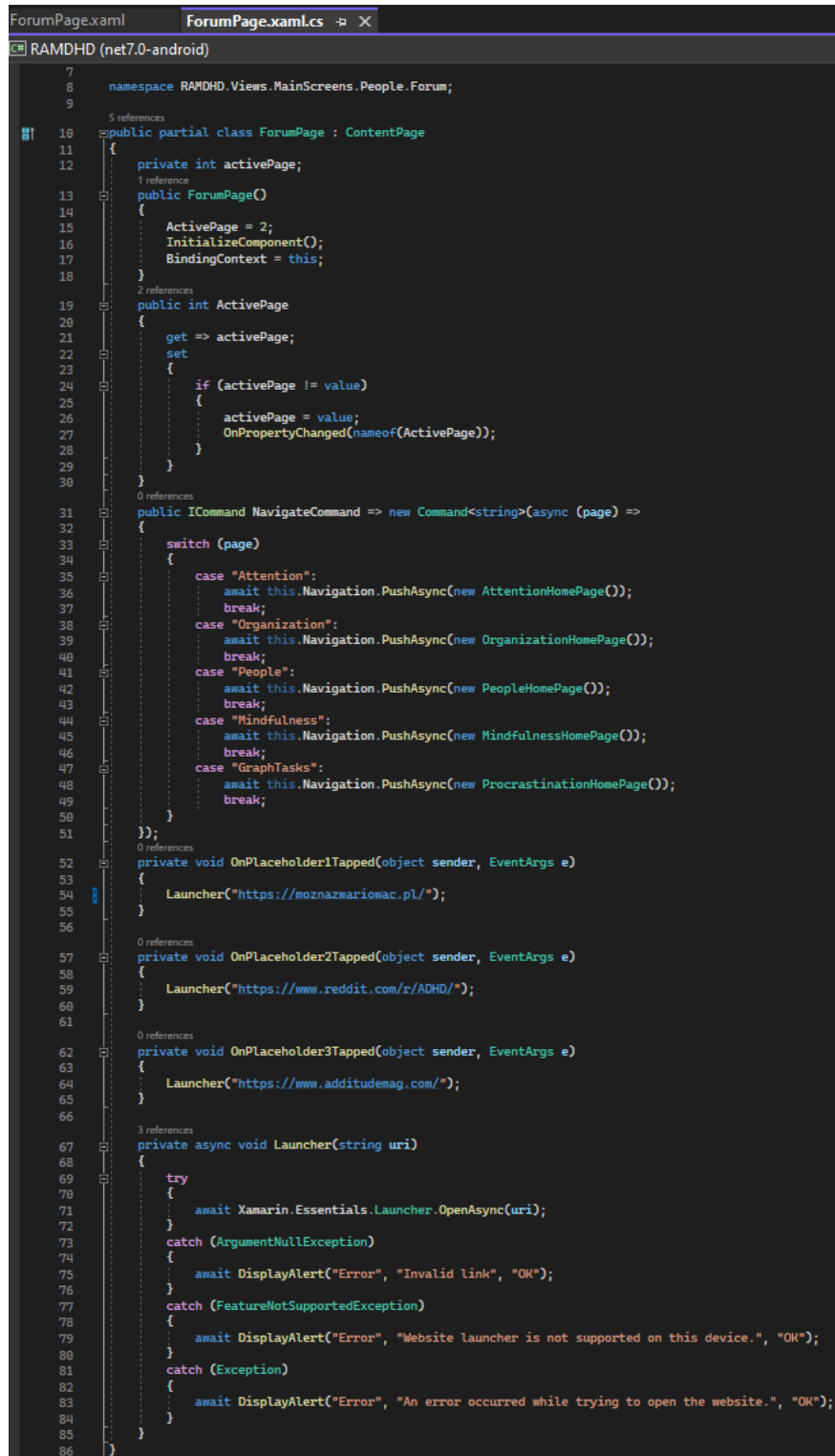
```

ForumPage.xaml  X ForumPage.xaml.cs
RAMDHD (net7.0-android)
1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
3      xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4      xmlns:local="clr-namespace:RAMDHD.Views.Components"
5      x:Class="RAMDHD.Views.MainScreens.People.Forum.ForumPage"
6      Title="Forums Page"
7      BackgroundColor="{StaticResource White}">
8      <AbsoluteLayout x:Name="MainLayout" VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand">
9
10         <!-- Internet Connection Banner -->
11         <StackLayout
12             AbsoluteLayout.LayoutBounds="0.5, 0, 1, AutoSize"
13             AbsoluteLayout.LayoutFlags="WidthProportional, PositionProportional"
14             BackgroundColor="Yellow"
15             Padding="10">
16             <Label Text="You must have an internet connection to access the forums"
17                 FontSize="14"
18                 FontFamily="ReemKufiFunRegular"
19                 TextColor="Red"
20                 HorizontalOptions="Center"
21                 HorizontalTextAlignment="Center" />
22         </StackLayout>
23
24         <!-- Header -->
25         <HorizontalStackLayout
26             AbsoluteLayout.LayoutBounds="0.2, 0.11, -1, -1"
27             AbsoluteLayout.LayoutFlags="PositionProportional">
28             <Label Text="Forums which you can trust"
29                 FontSize="16"
30                 TextColor="{StaticResource Tertiary}"
31                 FontFamily="ReemKufiFunBold"
32                 HorizontalOptions="Center"
33                 HorizontalTextAlignment="Center" />
34         </HorizontalStackLayout>
35
36         <!-- Placeholder Frames for Adding New Graph Tasks -->
37         <Frame
38             x:Name="placeholder1"
39             WidthRequest="400" HeightRequest="100"
40             BackgroundColor="WhiteSmoke"
41             CornerRadius="10"
42             AbsoluteLayout.LayoutBounds="0.5, 0.24, 0.5, 1"
43             AbsoluteLayout.LayoutFlags="PositionProportional">
44             <Frame.GestureRecognizers>
45                 <TapGestureRecognizer Tapped="OnPlaceholder1Tapped"/>
46             </Frame.GestureRecognizers>
47             <HorizontalStackLayout>
48                 <Label
49                     x:Name="placeholderLabel1"
50                     Text="{Binding Headline, FallbackValue='www.moznazwariowac.pl'}"
51                     FontFamily="ReemKufiFunRegular"
52                     TextColor="{StaticResource SharcBlack}"
53                     HeightRequest="50"
54                     WidthRequest="250"
55                     HorizontalOptions="Center"
56                     VerticalOptions="CenterAndExpand"
57                     Margin="40, 20, 0, 0">
58                 </Label>
59                 <Image Source="{Binding ImageSource, FallbackValue='internet.svg'}"
60                     WidthRequest="30"
61                     HeightRequest="30"
62                     HorizontalOptions="EndAndExpand"
63                     VerticalOptions="Center"
64                     Margin="0, 0, 10, 0">
65                     <Image.GestureRecognizers>
66                         <TapGestureRecognizer Tapped="OnPlaceholder1Tapped"/>
67                     </Image.GestureRecognizers>
68                 </Image>
69             </HorizontalStackLayout>
70         </Frame>
71
72         <Frame
73             x:Name="placeholder2"
74             WidthRequest="400" HeightRequest="100"
75             BackgroundColor="WhiteSmoke"
76             CornerRadius="10"
77             AbsoluteLayout.LayoutBounds="0.5, 0.4, 0.5, 1"
78             AbsoluteLayout.LayoutFlags="PositionProportional">
79             <Frame.GestureRecognizers>
80                 <TapGestureRecognizer Tapped="OnPlaceholder2Tapped"/>
81             </Frame.GestureRecognizers>
82             <HorizontalStackLayout>
83                 <Label
84                     x:Name="placeholderLabel2"
85                     Text="{Binding Headline, FallbackValue='ADHD Community on Reddit'}"
86                     FontFamily="ReemKufiFunRegular"
87                     TextColor="{StaticResource SharcBlack}"
88                     HeightRequest="50"
89                     WidthRequest="250"
90                     HorizontalOptions="Center"
91                     VerticalOptions="CenterAndExpand"
92                     Margin="40, 20, 0, 0">
93                 </Label>
94             </HorizontalStackLayout>
95         </Frame>
96     </AbsoluteLayout>

```

Figure 96. Source code of the ForumPage.xaml (tool: Visual Studio IDE [70], source: self)

Navigation to the external resource (outside the application) can be easily handled using the built-in `Xamarin.Essentials.Launcher` library. When the user clicks on any of the listed forums, the respecting placeholder function is triggered and calls on the `Launcher` function which handles opening the default browser as well as any of the possible exceptions as seen on figure 97.



```

7
8 namespace RAMDHD.Views.MainScreens.People.Forum;
9
10 public partial class ForumPage : ContentPage
11 {
12     private int activePage;
13     public ForumPage()
14     {
15         ActivePage = 2;
16         InitializeComponent();
17         BindingContext = this;
18     }
19     public int ActivePage
20     {
21         get => activePage;
22         set
23         {
24             if (activePage != value)
25             {
26                 activePage = value;
27                 OnPropertyChanged(nameof(ActivePage));
28             }
29         }
30     }
31     public ICommand NavigateCommand => new Command<string>(async (page) =>
32     {
33         switch (page)
34         {
35             case "Attention":
36                 await this.Navigation.PushAsync(new AttentionHomePage());
37                 break;
38             case "Organization":
39                 await this.Navigation.PushAsync(new OrganizationHomePage());
40                 break;
41             case "People":
42                 await this.Navigation.PushAsync(new PeopleHomePage());
43                 break;
44             case "Mindfulness":
45                 await this.Navigation.PushAsync(new MindfulnessHomePage());
46                 break;
47             case "GraphTasks":
48                 await this.Navigation.PushAsync(new ProcrastinationHomePage());
49                 break;
50         }
51     });
52     private void OnPlaceholder1Tapped(object sender, EventArgs e)
53     {
54         Launcher("https://moznazwariowac.pl/");
55     }
56     private void OnPlaceholder2Tapped(object sender, EventArgs e)
57     {
58         Launcher("https://www.reddit.com/r/ADHD/");
59     }
60     private void OnPlaceholder3Tapped(object sender, EventArgs e)
61     {
62         Launcher("https://www.additudemag.com/");
63     }
64     private async void Launcher(string uri)
65     {
66         try
67         {
68             await Xamarin.Essentials.Launcher.OpenAsync(uri);
69         }
70         catch (ArgumentNullException)
71         {
72             await DisplayAlert("Error", "Invalid link", "OK");
73         }
74         catch (FeatureNotSupportedException)
75         {
76             await DisplayAlert("Error", "Website launcher is not supported on this device.", "OK");
77         }
78         catch (Exception)
79         {
80             await DisplayAlert("Error", "An error occurred while trying to open the website.", "OK");
81         }
82     }
83 }
84
85
86

```

Figure 97. Source code of the `ForumPage.xaml.cs` (tool: Visual Studio IDE [70], source: self)

7.7. Deleting created earlier custom routine

To delete the created earlier custom routine user shall navigate back to the Routine Menu Page as seen on figure 98. To trigger a delete function user shall click the delete icon which by default is not visible because it applies only to existing routines. Once a routine is created it should be visible as a red “trash.svg” delete icon.

```
RoutineMenuPage.xaml  RoutineMenuPage.xaml.cs
RAMDHD (net7.0-android)
1 <?xml version="1.0" encoding="utf-8" ?>
2 <ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
3             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4             xmlns:local="clr-namespace:RAMDHD.Views.Components"
5             x:Class="RAMDHD.Views.MainScreens.Organization.Routine.RoutineMenuPage"
6             Title="Routine Menu Page"
7             BackgroundColor="White">
8
9     <AbsoluteLayout x:Name="MainLayout" VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand">
10
11         <!-- Header -->
12         <HorizontalStackLayout
13             AbsoluteLayout.LayoutBounds="0.2, 0.02, -1, -1"
14             AbsoluteLayout.LayoutFlags="PositionProportional">
15             <Label Text="Created routines"
16                 FontSize="16"
17                 TextColor="{StaticResource Tertiary}"
18                 FontFamily="ReemKufiFunBold"
19                 HorizontalOptions="Center"
20                 HorizontalTextAlignment="Center"
21             />
22         </HorizontalStackLayout>
23
24         <!-- Placeholder Frames for Adding New Graph Tasks -->
25         <Frame
26             x:Name="placeholder1"
27             WidthRequest="400" HeightRequest="100"
28             BackgroundColor="WhiteSmoke"
29             CornerRadius="10"
30             AbsoluteLayout.LayoutBounds="0.5, 0.16, 0.5, 1"
31             AbsoluteLayout.LayoutFlags="PositionProportional">
32             <Frame.GestureRecognizers>
33             <TapGestureRecognizer Tapped="OnPlaceholderTapped" CommandParameter="1"/>
34             </Frame.GestureRecognizers>
35             <HorizontalStackLayout>
36             <Label
37                 x:Name="placeholderLabel1"
38                 Text="Add new routine"
39                 FontFamily="ReemKufiFunRegular"
40                 TextColor="{StaticResource Gray200}"
41                 HeightRequest="50"
42                 WidthRequest="220"
43                 HorizontalOptions="Center"
44                 VerticalOptions="CenterAndExpand"
45                 Margin="40, 20, 0, 0">
46             </Label>
47             <Image Source="{Binding EditIcon, FallbackValue='edit_pencil.svg'}"
48                 WidthRequest="30"
49                 HeightRequest="30"
50                 HorizontalOptions="EndAndExpand"
51                 VerticalOptions="Center"
52                 Margin="0, 0, 0, 0">
53             <Image.GestureRecognizers>
54             <TapGestureRecognizer Tapped="EditRoutine" CommandParameter="1"/>
55             </Image.GestureRecognizers>
56             </Image>
57             <Image
58                 Source="{Binding DeleteIcon, FallbackValue='x.svg'}"
59                 WidthRequest="30"
60                 HeightRequest="30"
61                 HorizontalOptions="EndAndExpand"
62                 VerticalOptions="Center"
63                 Margin="20, 0, 0, 0">
64             <Image.GestureRecognizers>
65             <TapGestureRecognizer Tapped="DeleteRoutine" CommandParameter="1"/>
66             </Image.GestureRecognizers>
67             </Image>
68             </HorizontalStackLayout>
69         </Frame>
70
71         <Frame
72             x:Name="placeholder2"
73             WidthRequest="400" HeightRequest="100"
74             BackgroundColor="WhiteSmoke"
75             CornerRadius="10"
76             AbsoluteLayout.LayoutBounds="0.5, 0.32, 0.5, 1"
77             AbsoluteLayout.LayoutFlags="PositionProportional">
78             <Frame.GestureRecognizers>
79             <TapGestureRecognizer Tapped="OnPlaceholderTapped" CommandParameter="2"/>
80             </Frame.GestureRecognizers>
81             <HorizontalStackLayout>
82             <Label
83                 x:Name="placeholderLabel2"
84                 Text="Add new routine"
85                 FontFamily="ReemKufiFunRegular"
86                 TextColor="{StaticResource Gray200}"
87                 HeightRequest="50"
88                 WidthRequest="220"
89                 HorizontalOptions="Center"
90                 VerticalOptions="CenterAndExpand"
91                 Margin="40, 20, 0, 0">
92             </Label>
93             <Image Source="{Binding EditIcon, FallbackValue='edit_pencil.svg'}"
94                 WidthRequest="30"
95                 HeightRequest="30"
96                 WidthRequest="30"
97                 HeightRequest="30">
```

Figure 98. Source code of the RoutineMenuPage.xaml (tool: Visual Studio IDE [70], source: self)

Deletion is handled by the DeleteRoutine function. After clicking the red “trash.svg” delete icon user gets prompted for action confirmation as seen on figure 99. To confirm the deletion user shall click the “Yes” option. By the time the system displays successful deletion confirmation the routine is already irreversibly removed from the database. However, to visually indicate the changes, the system performs re-navigation action. First, a new RoutineMenuPage (reflecting the newest changes) is inserted before the current one. Next, the current page (the old RoutineMenuPage) is popped off the navigation stack. This way the user has an illusion of a page refresh.

```

88 private async void DeleteRoutine(object sender, EventArgs e)
89 {
90     if (sender is Image image && image.GestureRecognizers.FirstOrDefault() is TapGestureRecognizer tapGesture)
91     {
92         int.TryParse(tapGesture.CommandParameter.ToString(), out int id);
93
94         bool answer = await Application.Current.MainPage.DisplayAlert(
95             "Confirm Delete",
96             "Are you sure you want to delete this routine?",
97             "Yes",
98             "No"
99         );
100
101         if (answer)
102         {
103             // Perform the delete action
104             await _databaseAccess.DeleteRoutineByIdAsync(id);
105
106             await DisplayAlert("Success", $"Routine deleted successfully", "OK");
107
108             // Insert the new GraphTaskMenu page before the current page and then pop the current one
109             // Creates the illusion of page refreshing
110             Navigation.InsertPageBefore(new RoutineMenuPage(), this);
111             await Navigation.PopAsync();
112         }
113     }
114 }

```

Figure 99. Source code of the DeleteRoutine function in the RoutineMenuPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

This concludes the implementation behind all the functionalities that were tested by users during the conducted user tests. The results of those tests will be presented in the following chapter.

8. Conducting user tests

User tests were conducted once, on the final version of the application as presented in chapters 5. *Design of the proposed solution* and 7. *Implementation of the project*. Any recommendations made by users after the tests will be presented in this chapter and later discussed in chapter 9. *Consideration of the future improvements*.

8.1. User group

User group consisted of 5 users varying in age, their occupation and whether they consider themselves as an ADHD person (no official diagnoses were provided to verify that claim) as seen in table 17. According to Jakob Nielsen testing with only 5 users is sufficient for usability studies because the most significant insights are gained from the first few users. Each additional user provides diminishing returns, as they are likely to encounter the same issues as previous users. Testing with 5 users identifies around 85% of usability problems, making it more efficient and cost-effective. [81]

No.	Age	Occupation	ADHD consideration
User 1	21	receptionist	Yes
User 2	24	student	No
User 3	23	research scientist	Yes
User 4	25	student	No
User 5	46	lawyer	No

Table 17. Listing of users from the tested user group

8.2. Prepared tasks

Each user was given 5 tasks that read as follows:

1. Creating custom routine
2. Completing the custom routine (displaying and checking all the routine steps)
3. Creating custom note (reminding about deleting the routine)
4. Visiting one of the forums
5. Deleting created earlier custom routine

Each user was given a short introduction that reads as follows: “You will be given 5 tasks that should be performed using the application. If you are certain that a task has been completed, please say ‘SUCCESS’. If you are not sure if the task is completed or you are lost during the process, please say ‘STOP’. No additional instructions should be given during the process. You are welcome to think out loud while doing the tasks.”.

8.3. Results of task completion

Results of tasks completion are presented in table 18.

Task No.	Task description	User 1 result	User 2 result	User 3 result	User 4 result	User 5 result
1	Creating custom routine	SUCCESS	SUCCESS	SUCCESS	SUCCESS	STOP
2	Completing the custom routine	SUCCESS	SUCCESS	STOP	SUCCESS	SUCCESS
3	Creating custom note	SUCCESS	SUCCESS	SUCCESS	SUCCESS	SUCCESS
4	Visiting one of the forums	SUCCESS	SUCCESS	SUCCESS	SUCCESS	SUCCESS
5	Deleting created earlier custom routine	SUCCESS	SUCCESS	SUCCESS	SUCCESS	SUCCESS

Table 18. Results of tasks completion

8.4. Questionnaire

After finishing the tasks, regardless of the results, users were asked to answer the questionnaire - regarding the tasks and a final general impressions questionnaire.

8.4.1. Tasks questionnaire

Every task was given the same questionnaire. It was prepared based on the Likert scale [82]. The questionnaire reads as follows:

Assess the truthfulness of the following statements by marking X in the appropriate box

a. I didn't have any problems completing the task.

- i. strongly agree
- ii. somewhat agree
- iii. no opinion
- iv. somewhat disagree
- v. strongly disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?: x/10

Can you briefly justify your answer?

b. User interface was intuitive

- i. strongly agree
- ii. somewhat agree
- iii. no opinion
- iv. somewhat disagree
- v. strongly disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?: x/10

Can you briefly justify your answer?

c. The system was responsive – it reacted to my actions

- i. strongly agree
- ii. somewhat agree
- iii. no opinion
- iv. somewhat disagree
- v. strongly disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?: x/10

Can you briefly justify your answer?

8.4.2. General impressions questionnaire

General impressions questionnaire was to be answered at the very end. It was once again based on the Likert scale [82]. It reads as follows:

Assess the truthfulness of the following statements by marking X in the appropriate box

a. I enjoy using the app.

- i. somewhat agree
- ii. no opinion
- iii. somewhat disagree
- iv. strongly disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?: x/10

Can you briefly justify your answer?

b. Would you use this app in everyday life?

- i. strongly agree
- ii. somewhat agree
- iii. no opinion
- iv. somewhat disagree
- v. strongly disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?: x/10

Can you briefly justify your answer?

c. Have the functions presented met your expectations? ie. do they work like you expect them to?

- i. strongly agree
- ii. somewhat agree
- iii. no opinion
- iv. somewhat disagree
- v. strongly disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?: x/10

Can you briefly justify your answer?

- d. Would you be willing to pay for the app or its individual functions? e.g. additional fields to save notes, or a voluntary amount to support application developers? If so, how much? What would you expect in return?**
- i. strongly agree
 - ii. somewhat agree
 - iii. no opinion
 - iv. somewhat disagree
 - v. strongly disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?: x/10

Can you briefly justify your answer?

8.5. Results of the questionnaire

8.5.1. Tasks questionnaire

8.5.1.1. Task 1 - Creating custom routine

Figure 100 shows the bar graph presenting the results of task 1.

Results of task 1

Creating custom routine

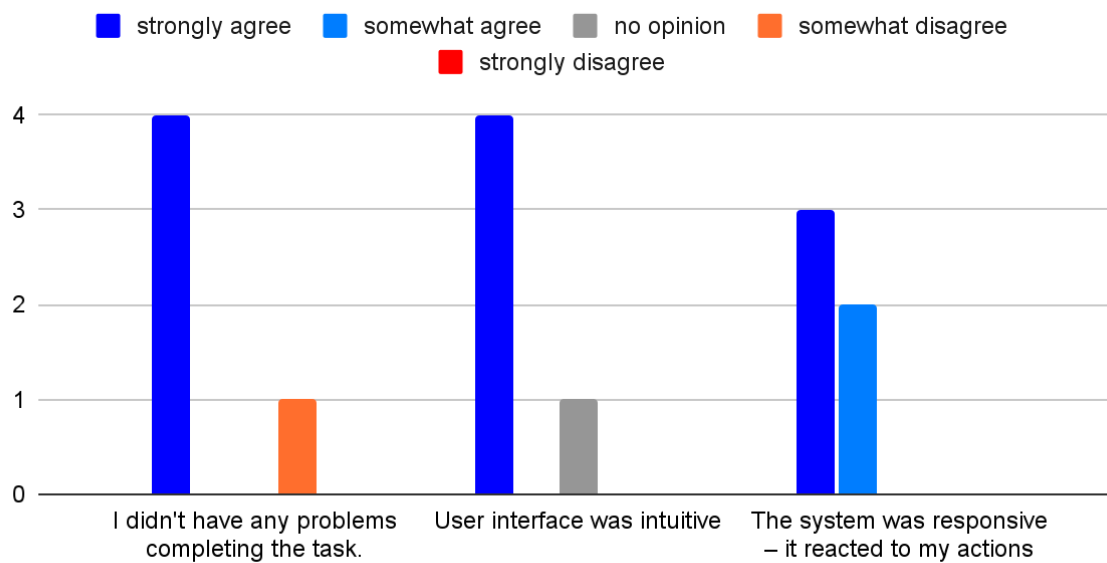


Figure 100. Bar graph presenting the results of task 1 - creating a custom routine

a. I didn't have any problems completing the task.

- user 1: strongly agree
- user 2: strongly agree
- user 3: strongly agree
- user 4: strongly agree
- user 5: somewhat disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- user 1: 9/10
- user 2: 10/10
- user 3: 10/10
- user 4: 10/10
- user 5: 5/10

Can you briefly justify your answer?

- user 1: -

- ii. user 2: *“the task was easy to complete”*
- iii. user 3: -
- iv. user 4: -
- v. user 5: *“I was confused by the initial instructions. I was just given an application and expected to do something with it. That is why the initial tasks were confusing. Once I got the feel of the application and knew what was expected of me, I easily completed all the tasks”*

b. User interface was intuitive

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: strongly agree
- iv. user 4: strongly agree
- v. user 5: no opinion

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 8/10
- iii. user 3: 10/10
- iv. user 4: 7/10
- v. user 5: 5/10

Can you briefly justify your answer?

- i. user 1: *“transparent breakdown for the entry fields”*
- ii. user 2: *“The application interface may need to be ‘clicked through’ to familiarize perself on the first use”*
- iii. user 3: *“CALENDAR WITH PLUS ICON”*
- iv. user 4: *“I thought I should click the ‘add new routine’ label instead of the pencil. Otherwise everything was fine”*
- v. user 5: -

c. The system was responsive – it reacted to my actions

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat agree
- iv. user 4: somewhat agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 8/10

- iv. user 4: 8/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: *“there were no problems, the app didn't jam”*
- ii. user 2: -
- iii. user 3: *“i guess it was fine”*
- iv. user 4: *“the system was responsive, but the transitions between panels take time. Short enough not to be discouraged, but still”*
- v. user 5: -

8.5.1.2. Task 2 - Completing the custom routine

Figure 101 shows the bar graph presenting the results of task 2.

Results of task 2

Completing the custom routine

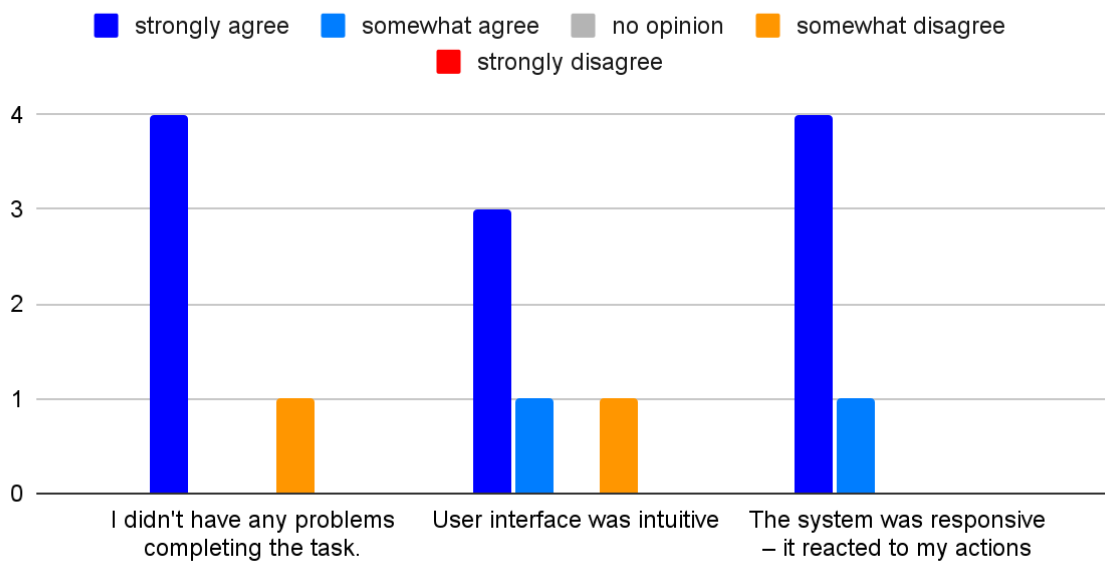


Figure 101. Bar graph presenting the results of task 2 - completing the custom routine

a. I didn't have any problems completing the task.

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat disagree
- iv. user 4: strongly agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10

- ii. user 2: 10/10
- iii. user 3: 2/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: -
- ii. user 2: *“the task was very easy”*
- iii. user 3: *“I didn’t know the routine label could be clicked. Maybe it should have some distinct background color? Or maybe there should be a transition from the routine edition screen to the routine itself? Or some different tab?”*
- iv. user 4: *“easy”*
- v. user 5: *“The only difficulty was misunderstanding the actions the author expects. The execution was easy.”*

b. User interface was intuitive

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat disagree
- iv. user 4: strongly agree
- v. user 5: somewhat agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 4/10
- iv. user 4: 10/10
- v. user 5: 8/10

Can you briefly justify your answer?

- i. user 1: *“everything was clear”*
- ii. user 2: -
- iii. user 3: *“Not so, because of the above; in most of this type of app that I know, the tasks appear graphically only after creating them, so you are sure that they will be found in a given place.”*
- iv. user 4: -
- v. user 5: -

c. The system was responsive – it reacted to my actions

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat agree
- iv. user 4: strongly agree

- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 8/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: *“the effect was visible when a single action was selected (‘tick’)”*
- ii. user 2: *“the system worked without flaws”*
- iii. user 3: -
- iv. user 4: -
- v. user 5: -

8.5.1.3. Task 3 - Creating custom note

Figure 102 shows the bar graph presenting the results of task 3.

Results of task 3

Creating custom note

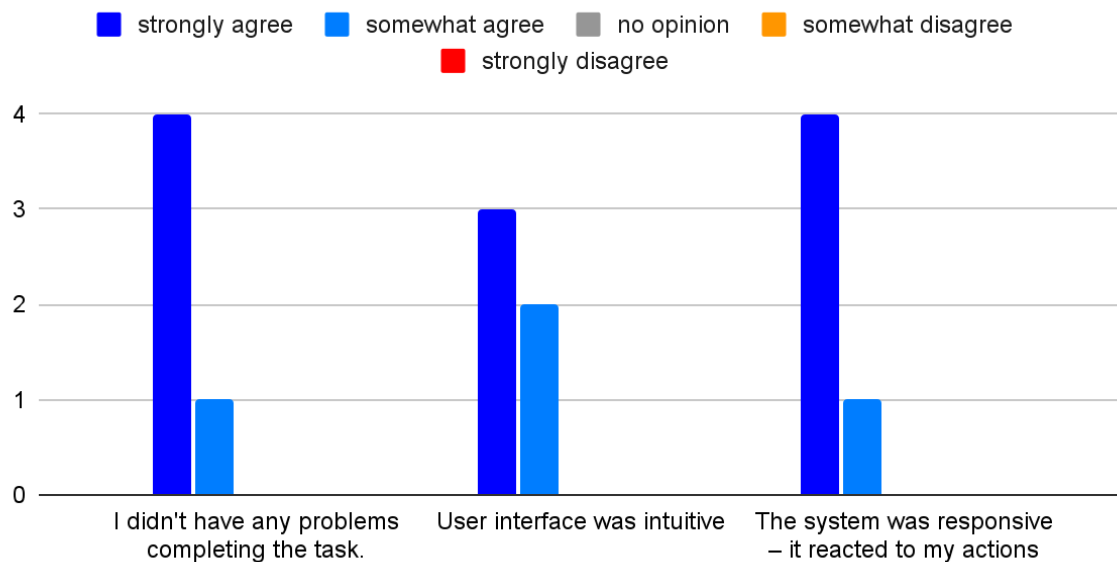


Figure 102. Bar graph presenting the results of task 3 - creating a custom note

a. I didn't have any problems completing the task.

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat agree

- iv. user 4: strongly agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 10/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: -
- ii. user 2: -
- iii. user 3: *“I guessed a little bit where I could find the tab, but I guessed correctly:)”*
- iv. user 4: -
- v. user 5: -

b. User interface was intuitive

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat agree
- iv. user 4: strongly agree
- v. user 5: somewhat agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 7/10
- iv. user 4: 10/10
- v. user 5: 8/10

Can you briefly justify your answer?

- i. user 1: *“everything was clear”*
- ii. user 2: -
- iii. user 3: -
- iv. user 4: -
- v. user 5: -

c. The system was responsive – it reacted to my actions

- i. user 1: strongly agree
- ii. user 2: strongly agree

- iii. user 3: somewhat agree
- iv. user 4: strongly agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 8/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: *“information about successful note creation popped out”*
- ii. user 2: -
- iii. user 3: -
- iv. user 4: -
- v. user 5: -

8.5.1.4. Task 4 - Visiting one of the forums

Figure 103 shows the bar graph presenting the results of task 4.

Results of task 4

Visiting one of the forums

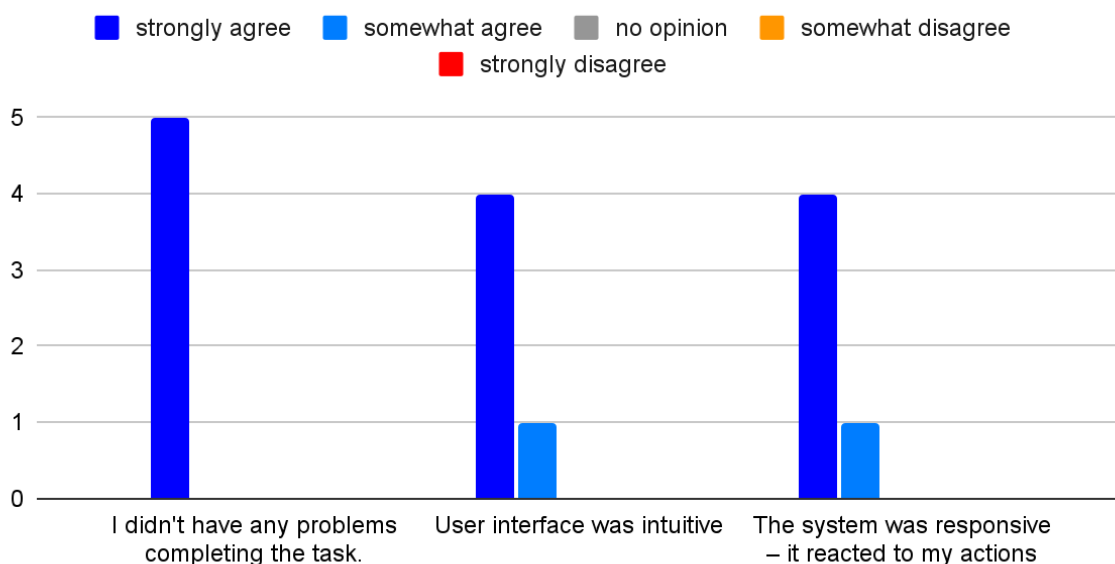


Figure 103. Bar graph presenting the results of task 4 - visiting one of the forums

a. I didn't have any problems completing the task.

- i. user 1: strongly agree

- ii. user 2: strongly agree
- iii. user 3: strongly agree
- iv. user 4: strongly agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 10/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: -
- ii. user 2: -
- iii. user 3: “*very intuitive icon*”
- iv. user 4: -
- v. user 5: “*The only difficulty was misunderstanding the actions the author expects. The execution was easy.*”

b. User interface was intuitive

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: strongly agree
- iv. user 4: strongly agree
- v. user 5: somewhat agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 10/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: -
- ii. user 2: -
- iii. user 3: -
- iv. user 4: -
- v. user 5: -

c. The system was responsive – it reacted to my actions

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat agree
- iv. user 4: strongly agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 8/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: -
- ii. user 2: *“the system worked without flaws”*
- iii. user 3: -
- iv. user 4: -
- v. user 5: -

8.5.1.5. Task 5 - Deleting created earlier custom routine

Figure 104 shows the bar graph presenting the results of task 5.

Results of task 5

Deleting created earlier custom routine

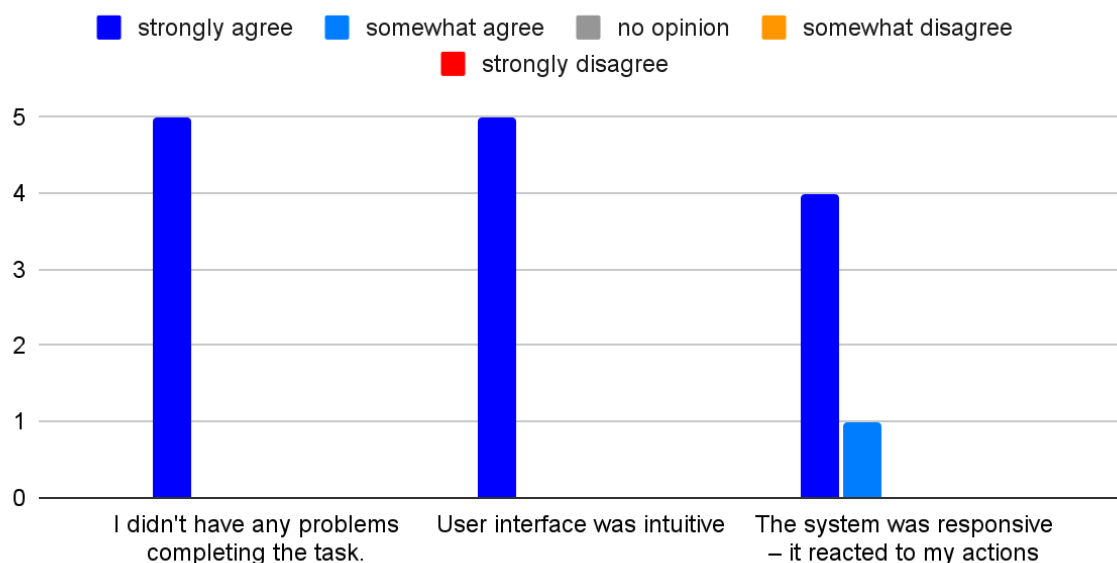


Figure 104. Bar graph presenting the results of task 5 - deleting created earlier custom routine

a. I didn't have any problems completing the task.

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: strongly agree
- iv. user 4: strongly agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 10/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: -
- ii. user 2: *"the task was very easy"*
- iii. user 3: *"I was familiar with the trash icon from task 2, so intuitively again it made sense to come back to the 'calendar with plus' icon."*
- iv. user 4: -
- v. user 5: *"The longer I used the application the easier it was to navigate"*

b. User interface was intuitive

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: strongly agree
- iv. user 4: strongly agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 10/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: -
- ii. user 2: -
- iii. user 3: -
- iv. user 4: -
- v. user 5: -

c. The system was responsive – it reacted to my actions

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat agree
- iv. user 4: strongly agree
- v. user 5: strongly agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 10/10
- iii. user 3: 8/10
- iv. user 4: 10/10
- v. user 5: 10/10

Can you briefly justify your answer?

- i. user 1: *“the notification popped out”*
- ii. user 2: *“the system worked without flaws”*
- iii. user 3: -
- iv. user 4: -
- v. user 5: -

8.5.2. General impressions questionnaire

Figure 105 shows the bar graph presenting general impressions of the application.

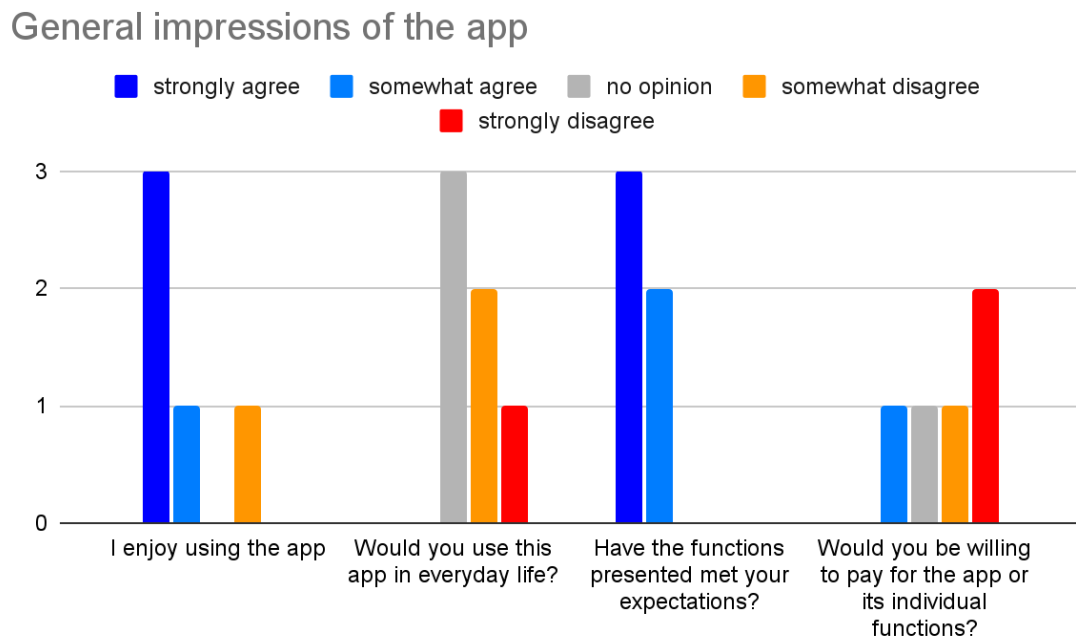


Figure 105. Bar graph presenting generals impressions of the application

a. I enjoy using the app.

- i. user 1: strongly agree
- ii. user 2: strongly agree
- iii. user 3: somewhat agree
- iv. user 4: strongly agree
- v. user 5: somewhat disagree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 10/10
- ii. user 2: 8/10
- iii. user 3: 7/10
- iv. user 4: 10/10
- v. user 5: 8/10

Can you briefly justify your answer?

- i. user 1: *“There were no problems, the application did not jam, the interface was clear, the application worked with me - messages about successful completion of tasks popped out”*
- ii. user 2: *“The application works smoothly and looks very good which translates into satisfactory use”*
- iii. user 3: *“Charming starting screen and a compact user interface”*

- iv. user 4: -
- v. user 5: *"My initial confusion was the result of unclear instructions on what to do"*

b. Would you use this app in everyday life?

- i. user1: somewhat disagree
- ii. user 2: no opinion
- iii. user 3: strongly disagree
- iv. user 4: somewhat disagree
- v. user 5: no opinion

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 3/10
- ii. user 2: 5/10
- iii. user 3: 2/10
- iv. user 4: 4/10
- v. user 5: 5/10

Can you briefly justify your answer?

- i. user 1: *"Currently, I'm trying to limit the use of my phone, so I'd rather do these types of tasks on a regular piece of paper. The option that I liked is the possibility of navigating to social forums, because then you avoid the temptation of scrolling through facebook main page, which sometimes in my case becomes automatic and without thinking when I enter this platform"*
- ii. user 2: *"Currently I have no need to use this application, but if it had changed, I would have given it a chance."*
- iii. user 3: *"I am certain that I would forgot about using the application, but that is by no means an issue with the application itself"*
- iv. user 4: *"I don't think I need this kind of an app."*
- v. user 5: *"I don't use these kinds of apps everyday. I prefer pen and paper"*

c. Have the functions presented met your expectations? ie. do they work like you expect them to?

- i. user 1: somewhat agree
- ii. user 2: strongly agree
- iii. user 3: strongly agree
- iv. user 4: strongly agree
- v. user 5: somewhat agree

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user1: 8/10
- ii. user 2: 9/10

- iii. user 3: 9/10
- iv. user 4: 10/10
- v. user 5: 8/10

Can you briefly justify your answer?

- i. user 1: *"It might be useful for notes to pop up on the front page every now and then in case you forget about one."*
- ii. user 2: *"The application meets my expectations and should satisfy even the most demanding users."*
- iii. user 3: -
- iv. user 4: -
- v. user 5: -

d. Would you be willing to pay for the app or its individual functions? e.g. additional fields to save notes, or a voluntary amount to support application developers? If so, how much? What would you expect in return?

- i. user1: somewhat disagree
- ii. user 2: somewhat agree
- iii. user 3: strongly disagree
- iv. user 4: strongly disagree
- v. user 5: no opinion

Are you able to determine your response, this time on a scale from 1 (strongly disagree) to 10 (strongly agree)?:

- i. user 1: 3/10
- ii. user 2: 7/10
- iii. user 3: 1/10
- iv. user 4: 1/10
- v. user 5: 5/10

Can you briefly justify your answer?

- i. user 1: *"As mentioned earlier, I'm trying to limit my phone usage and use paper instead. However, in case the application is practical for me, I could only deposit a small amount of support for the developers of the application."*
- ii. user 2: *"A voluntary amount supporting the creators is an interesting option."*
- iii. user 3: *"I recommend advertising; paywalls in most good applications of this type practically do not occur and would only reduce demand"*
- iv. user 4: *"No. I don't know what kind of functionality it should offer for me to pay for it"*
- v. user 5: *"I was using the app for only a short period of time to explore it well enough. Also, Mentioned that I prefer pen and paper."*

8.6. Conclusions from the carried out tests

The majority feedback on the application was positive. Users were satisfied with the functionality and the design, finding most tasks easy to complete. Users appreciated the intuitive interface and responsiveness of the system.

- Task 1: Creating a Custom Routine: Users gave an average rating of 8.8. Most users found it easy, with four strongly agreeing and one finding the initial instructions confusing. The interface was rated 8 for intuition, and the system's responsiveness scored 9.2.
- Task 2: Completing the Custom Routine: This task received an average rating of 8.4. While users generally found it easy, there were suggestions for improving the routine label interaction. The interface was rated 8.4, and responsiveness scored 9.2.
- Task 3: Creating a Custom Note: All users rated this task highly, with an average score of 10. Despite one user guessing the tab location, the task was completed easily. The interface intuition averaged 9, and responsiveness scored 9.6.
- Task 4: Visiting One of the Forums: Users gave perfect scores, averaging 10 for both completion ease and interface intuition. The system's responsiveness was rated at 9.6.
- Task 5: Deleting a Previously Created Custom Routine: Consistently high ratings with an average score of 10 for both task completion and interface intuition. Responsiveness also scored 9.6.
- General Impressions: Users enjoyed using the app, with scores ranging from 7 to 10, averaging 8.6. However, they were less inclined to use it daily (average score of 3.8) or pay for additional features (average score of 3.4). They felt the functions met their expectations, with an average score of 8.8.

As suggested by Jakob Nielsen, it is preferable to perform iterative tests to improve the design and feel of the application continuously [80]. However, before the next user testing, the application should be improved as suggested by tested users themselves and by the author. Users pointed out the need for clearer initial instructions, better labeling, and smoother panel transitions. As it is only an early stage of a serious development, there are some things even the author considers to require improvement. The next chapter discusses these possible changes.

9. Consideration of future improvements

This chapter discusses some of the potential improvements to the application that can be introduced with regard to the project's possible wider usage.

9.1. Platform-specific application

The application was implemented in .NET MAUI [71] because the author intended the project to be multi-platform. However, throughout the implementation process it was recognized that MAUI lacks some features and flexibility that platform-specific languages offer. With that in mind, it may be possible to port an existing application to the iOS system but make an Android-specific version written in Kotlin instead. The market-potential of the application will be considered when deciding on whether to implement a platform-specific version.

However, even with great intentions of accessibility Apple, as a company, is notorious for privacy abusing tactics and is the literal anti-actor for free software [83]. For this very reason it will be greatly considered whether it is both morally and ethically allowed to make the application accessible for Apple's ecosystem.

9.2. Implementing suggested changes

As suggested by the psychologist consulted, users tested and the author perself there are changes to the application that are considered to be a priority. These changes shall be implemented in the first place.

9.3. Maintenance of the project in the spirit of free software

If feasible, the author aims to implement the project - mobile application and any of the extensions (ie. website, in the spirit of free software, that is to be GPL compatible. The author believes that, as a developer, it is morally and ethically right to make the software copyleft. By the very least strive to make it copyleft and make the code open.

9.4. Paid access to features

As stated by the idea of Free Software, selling does stand in opposition to redistribution [84]. The application price or paid features does not exclude it being open-sourced and free for redistribution.

Neither should financially supporting the project introduce inequalities in the usage. Paid features are fair if they reflect the user's habits and frequency of usage. User's are not free if they can't access additional features that are behind a paywall. Forcing the user to pay for extra features is not good either. The app should provide sufficient access to features for all the users and only introduce paid features if they extend the app's fundamental capabilities of supporting the user.

Some tested users suggested that a feasible way of making money is to run ads within the application. The author believes this is a fine approach as long as:

1. ads are clearly marked as ads and not sneaked into the app as a regular content (as it seems to be the case for Ex-Twitter [85])
2. ads are carefully curated to advertise ethical and moral products, services and events (this may be curated with the help of the user perself)
3. there is a transparent process of:
 - a. collecting agreed upon user's data to show personalized ads (or not collecting at all and showing general-audience ads)
 - b. explaining in great detail why the given ads are shown
 - c. easily opting out of the ads

Contrary to popular belief that the most important consideration about ads is to make them seamlessly blend into the product, the author strongly believes that ads should be clearly marked and the reason behind their display should be explained.

9.5. Account creation

Mobile applications introduce some advantages over a web application. One of them is often an inherent local-first approach. The user has access to the application on his own device without the need to connect to the internet first. As seen by the example of this project, this approach may very well be sufficient.

However, account creation may be a good option for users that want to access their data across devices. It also introduces a great possibility of creating a web version of the application that can be accessed online.

10. Summary

This paper introduced an idea for a mobile application supporting adults with ADHD that includes strategies for dealing with procrastination, memory and attention management. Moreover, it showed how an app can be both accessible to the users and free, and open for the community and developers.

The presented application is seen as a stepping stone for a more tailored experience to be crafted in the near future. With such a laid foundation it is feasible to develop a feature-rich app that will invite the users without overwhelming them with options. In the foreseeable future the author wishes to develop the app more, introduce all the necessary enhancements and invite more people to access the project and share their feedback on it. It is in great interest of the author to further explore the possibilities of improvement.

The author strongly believes that altruistic endeavors are the most precious ways to spend time and resources. Both free software and altruism put ethics and morality in the core of a person's motives, which is the ultimate directory for making useful actions. Having that in mind, the author will further explore ways in which these values can be spread, including software as presented in this paper.

Bibliography

1. GNU Operating System/Why Open Source Misses the Point of Free Software (<https://www.gnu.org/philosophy/open-source-misses-the-point.html>)
2. fMRI Definition & Usage Examples - Dictionary.com (<https://www.dictionary.com/browse/fmri>)
3. Diagnostic and Statistical Manual of Mental Disorders, Fifth Edition, Text Revision (DSM-5-TR®) - American Psychiatric Association Publishing (<https://www.appi.org/Products/DSM-Library/Diagnostic-and-Statistical-Manual-of-Mental-Di-%281%29?sku=2576>)
4. American Psychological Association | APA Definition & Divisions (<https://study.com/learn/lesson/american-psychological-association-overview-what-is-the-apa.html>)
5. Gehricke, J. G., Kruggel, F., Thampipop, T., Alejo, S. D., Tatos, E., Fallon, J., & Muftuler, L. T. (2017). The brain anatomy of attention-deficit/hyperactivity disorder in young adults - a magnetic resonance imaging study. *PloS one*, 12(4), e0175433. <https://doi.org/10.1371/journal.pone.0175433> (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5391018/>)
6. Inside The ADHD Brain: Structure, Function, And Chemistry (<https://add.org/adhd-brain/>)
7. Regional differences in cerebral perfusion associated with the α -2A-adrenergic receptor genotypes in attention deficit hyperactivity disorder. Boong-Nyun Kim, Jae-Won Kim, Hyejin Kang, Soo-Churl Cho, Min-Sup Shin, Hee-Jeong Yoo, Soon-Beom Hong and Dong Soo Lee J. Psychiatry Neurosci September 01, 2010 35 (5) 330-336; DOI: <https://doi.org/10.1503/jpn.090168> (<https://www.jpn.ca/content/35/5/330>)
8. The ADHD vs. Non-ADHD Brain (<https://www.verywellmind.com/the-adhd-brain-4129396>)
9. Types of Neurological Sleep Disorder and Potential Drug for Their Treatment: A Brief Review (<https://www.eurchembull.com/uploads/paper/d4afe67f991bfb4f4498227bc46e8575.pdf>)
10. Volkow, N. D., Wang, G. J., Kollins, S. H., Wigal, T. L., Newcorn, J. H., Telang, F., Fowler, J. S., Zhu, W., Logan, J., Ma, Y., Pradhan, K., Wong, C., & Swanson, J. M. (2009). Evaluating dopamine reward pathway in ADHD: clinical implications. *JAMA*, 302(10), 1084–1091. <https://doi.org/10.1001/jama.2009.1308> (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2958516/>)
11. DIAGNOSTIC AND STATISTICAL MANUAL OF MENTAL DISORDERS FIFTH EDITION TEXT REVISION DSM-5-TR (<https://www.mredscircleoftrust.com/storage/app/media/DSM%205%20TR.pdf>)
12. ICD-10-CM Codes Lookup (<https://www.aapc.com/codes/icd-10-codes-range/>)
13. Slobodin O and Davidovitch M (2019) Gender Differences in Objective and Subjective Measures of ADHD Among Clinic-Referred Children. *Front. Hum. Neurosci.* 13:441. doi: 10.3389/fnhum.2019.00441 <https://www.frontiersin.org/articles/10.3389/fnhum.2019.00441/full>
14. The History of ADHD: A Timeline (<https://www.healthline.com/health/adhd/history>)
15. The Mini ADHD Coach/The Unofficial List of ADHD Symptoms (<https://www.theminiadhdcoach.com/unofficial-adhd-symptoms#strongthe-unofficial-list-of-adhd-symptomsstrong%E2%80%8D>)
16. Epstein, J. N., & Loren, R. E. (2013). Changes in the Definition of ADHD in DSM-5: Subtle but Important. *Neuropsychiatry*, 3(5), 455–458. <https://doi.org/10.2217/npv.13.59> (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3955126/>)
17. McLennan J. D. (2016). Understanding attention deficit hyperactivity disorder as a continuum. *Canadian family physician Medecin de famille canadien*, 62(12), 979–982. (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5154646/>)
18. Why You Put Things Off Until the Last Minute (<https://www.mcleanhospital.org/essential/procrastination>)
19. The Link Between ADHD and Procrastination (<https://www.theminiadhdcoach.com/living-with-adhd/adhd-procrastination>)
20. Roselló, B., Berenguer, C., Baixauli, I., Mira, Á., Martinez-Raga, J., & Miranda, A. (2020). Empirical examination of executive functioning, ADHD associated behaviors, and functional impairments in adults with persistent ADHD, remittent ADHD, and without ADHD. *BMC psychiatry*, 20(1), 134.

- <https://doi.org/10.1186/s12888-020-02542-y>
(<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7092442/>)
21. Sibley, M. H., Mitchell, J. T., & Becker, S. P. (2016). Method of adult diagnosis influences estimated persistence of childhood ADHD: a systematic review of longitudinal studies. *The lancet. Psychiatry*, 3(12), 1157–1165. [https://doi.org/10.1016/S2215-0366\(16\)30190-0](https://doi.org/10.1016/S2215-0366(16)30190-0)
(<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2626918/>)
 22. Better Genderless Pronouns in English (<https://stallman.org/articles/genderless-pronouns.html>)
 23. ADHD Procrastination: Causes, Triggers, Solutions (<https://numo.so/journal/adhd-procrastination>)
 24. Niermann, H. C., & Scheres, A. (2014). The relation between procrastination and symptoms of attention-deficit hyperactivity disorder (ADHD) in undergraduate students. *International journal of methods in psychiatric research*, 23(4), 411–421. <https://doi.org/10.1002/mpr.1440>
(<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6878228/>)
 25. 6 Ways to Combat Procrastination for Adults With ADHD
(<https://www.psychologytoday.com/gb/blog/the-best-strategies-for-managing-adult-adhd/202106/6-ways-to-combat-procrastination-for-adults>)
 26. ADHD and Procrastination: How They're Connected and What to Do About It
(<https://solvingprocrastination.com/adhd/>)
 27. Henry L. Roediger, Franklin M. Zaromb, Wenbo Lin, 1.02.3.2 Short-Term Storage, Learning and Memory: A Comprehensive Reference (Second Edition), Academic Press, 2017, Pages 7-19, ISBN 9780128052914, <https://doi.org/10.1016/B978-0-12-809324-5.21003-1>
(<https://www.sciencedirect.com/topics/medicine-and-dentistry/short-term-memory>)
 28. Cowan N. (2014). Working Memory Underpins Cognitive Development, Learning, and Education. *Educational psychology review*, 26(2), 197–223. <https://doi.org/10.1007/s10648-013-9246-y>
(<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4207727/>)
 29. What Is Long-Term Memory? (<https://www.verywellmind.com/what-is-long-term-memory-2795347>)
 30. Bart Aben, Sven Stapert, Arjan Blokland (2012). About the Distinction between Working Memory and Short-Term Memory. *Frontiers in Psychology* vol.3. <https://doi.org/10.3389/fpsyg.2012.00301>
(<https://www.frontiersin.org/journals/psychology/articles/10.3389/fpsyg.2012.00301/full>)
 31. How Psychologists Define Attention (<https://www.verywellmind.com/what-is-attention-2795009>)
 32. William James. *The Principles of Psychology* (1890). CHAPTER XI. ATTENTION
(<https://psychclassics.yorku.ca/James/Principles/prin11.htm>)
 33. Mills, B. D., Miranda-Dominguez, O., Mills, K. L., Earl, E., Cordova, M., Painter, J., Karalunas, S. L., Nigg, J. T., & Fair, D. A. (2018). ADHD and attentional control: Impaired segregation of task positive and task negative brain networks. *Network neuroscience* (Cambridge, Mass.), 2(2), 200–217.
https://doi.org/10.1162/netn_a_00034 (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6130439/>)
 34. Notion app homepage (<https://www.notion.so/>)
 35. Any.do app homepage (<https://www.any.do/>)
 36. Numo app homepage (<https://numo.so/>)
 37. Remember The Milk app homepage (<https://www.rememberthemilk.com/>)
 38. Johansson F, Rozental A, Edlund K, et al. Associations Between Procrastination and Subsequent Health Outcomes Among University Students in Sweden. *JAMA Netw Open*. 2023;6(1):e2249346. doi:10.1001/jamanetworkopen.2022.49346
(<https://jamanetwork.com/journals/jamanetworkopen/fullarticle/2800006>)
 39. Svartdal, F., Granmo, S., & Færevag, F. S. (2018). On the Behavioral Side of Procrastination: Exploring Behavioral Delay in Real-Life Settings. *Frontiers in psychology*, 9, 746.
<https://doi.org/10.3389/fpsyg.2018.00746> (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5964561/>)
 40. Mental Health America/ADHD Test - A self-report screening scale of ADHD - Adult ADHD Self-Report Scale (ASRS) v1.1 (<https://screening.mhanational.org/screening-tools/adhd/>)
 41. Dorota Dot Okulicz (<https://dorotadotokulicz.pl/>)
 42. Mental Health America/About our mental health tests - ADHD Test: Adult ADHD Self-Report Scale (ASRS) v1.1 (<https://screening.mhanational.org/about-our-mental-health-tests/#adhd>)

43. Nielsen Norman Group/10 Usability Heuristics for User Interface Design/1: Visibility of System Status (<https://www.nngroup.com/articles/ten-usability-heuristics/#toc-1-visibility-of-system-status-1>)
44. GNU Operating System/GNU General Public License, version 3 (<https://www.gnu.org/licenses/gpl-3.0.html>)
45. Nielsen Norman Group/The 3-Click Rule for Navigation Is False (<https://www.nngroup.com/articles/3-click-rule/>)
46. Nielsen Norman Group/Response Times: The 3 Important Limits (<https://www.nngroup.com/articles/response-times-3-important-limits/>)
47. GNU Operating System/Why Open Source Misses the Point of Free Software? (<https://www.gnu.org/philosophy/open-source-misses-the-point.html>)
48. UMLet 15.1 - Free UML Tool for Fast UML Diagrams (<https://www.umlet.com/>)
49. Malinowski Antoni, "How to efficiently connect people online? Implementing a social media platform that fosters software maintainability and collaboration between developers." (Engineering thesis, Polish-Japanese Academy of Information Technology, 2024) (https://docs.google.com/document/d/1JScNMTtzE-o2z3nql9wiOhPe0rlIFG2sg1Z7jdcJ7Y/edit?usp=s_haring)
50. Nielsen Norman Group/10 Usability Heuristics for User Interface Design (<https://www.nngroup.com/articles/ten-usability-heuristics/>)
51. Marvel (<https://marvelapp.com/>)
52. GNU Operating System/What is Free Software? (<https://www.gnu.org/philosophy/free-sw.html>)
53. GNU Operating System/What is Free Software?/The Free Software Definition/The four essential freedoms (<https://www.gnu.org/philosophy/free-sw.html#four-freedoms>)
54. GNU Operating System homepage (<https://www.gnu.org/>)
55. Wikipedia/UMLet (<https://en.wikipedia.org/wiki/UMLet>)
56. Github repository of UMLet (<https://github.com/umlet/umlet>)
57. DEV Community/An Introduction to Git: The Basics Every Beginning Developer Should Know (<https://dev.to/ionos/an-introduction-to-git-the-basics-every-beginning-developer-should-know-062>)
58. Git homepage (<https://git-scm.com/>)
59. Git/1.2 Getting Started - A Short History of Git (<https://git-scm.com/book/en/v2/Getting-Started-A-Short-History-of-Git>)
60. Wikipedia/BitKeeper (https://en.wikipedia.org/wiki/BitKeeper#BitKeeper_and_the_Linux_Kernel)
61. Linux Journal/A Git Origin Story (<https://www.linuxjournal.com/content/git-origin-story>)
62. SQLite Home Page/About (<https://www.sqlite.org/about.html>)
63. SQLite Home Page (<https://www.sqlite.org/>)
64. SQLite Home Page/License (<https://www.sqlite.org/copyright.html>)
65. TechTarget/GNU General Public License (GNU GPL or GPL) (<https://www.techtarget.com/searchdatacenter/definition/GNU-General-Public-License-GNU-GPL-or-simplify-GPL>)
66. GNU Operating System/Licenses/GNU General Public License, version 2 (<https://www.gnu.org/licenses/old-licenses/gpl-2.0.html>)
67. SQLite Home Page/Size Of The SQLite Library (<https://www.sqlite.org/footprint.html>)
68. Geekflare/Top 13 Open Source Database Software for Your Next Project (<https://geekflare.com/open-source-database/>)
69. DigitalOcean/SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems (<https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems#sqlite>)
70. Visual Studio homepage (<https://visualstudio.microsoft.com/#vs-section>)
71. .NET Multi-platform App UI documentation/What is .NET MAUI? (<https://learn.microsoft.com/en-gb/dotnet/maui/what-is-maui?view=net-maui-8.0>)
72. GitHub/.NET Platform/maui (<https://github.com/dotnet/maui>)
73. Stack Overflow/MIT vs GPL license [closed] (<https://stackoverflow.com/a/11752204/12322728>)

74. .NET Multi-platform App UI documentation/What is .NET MAUI?/How .NET MAUI works (<https://learn.microsoft.com/en-gb/dotnet/maui/what-is-maui?view=net-maui-7.0#how-net-maui-works>)
75. Medium/Announcing unDraw - Katerina Limpitsuni (<https://blog.prototypr.io/announcing-undraw-edb3460e258e>)
76. unDraw/illustrations browsing page (<https://undraw.co/illustrations>)
77. unDraw/license (<https://undraw.co/license>)
78. Iconoir (<https://iconoir.com/>)
79. GitHub/Iconoir (<https://github.com/iconoir-icons/iconoir>)
80. SolutionsHub/What is Proprietary Software? (<https://solutionshub.epam.com/blog/post/proprietary-software-definition-examples>)
81. Nielsen Norman Group/Why You Only Need to Test with 5 Users (<https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/>)
82. Wikipedia/Likert scale (https://en.wikipedia.org/wiki/Likert_scale)
83. Richard Stallman's personal site/Reasons not to use Apple (<https://www.stallman.org/apple.html>)
84. GNU Operating System/Philosophy of the GNU Project/Selling Free Software (<https://www.gnu.org/philosophy/selling.html>)
85. Richard Stallman's personal site/Reasons not to use Ex-Twitter (<https://www.stallman.org/twitter.html>)

Tables listing

Table 1. Dictionary of technical terms

Table 2. Functional requirements table for the ADHD screening test module (source: self)

Table 3. Functional requirements table for the timer panel included within the attention module (source: self)

Table 4. Functional requirements table for the notes panel included within the attention module (source: self)

Table 5. Functional requirements table for the education flashcards included within the attention module (source: self)

Table 6. Functional requirements table for the challenging flashcards included within the attention module (source: self)

Table 7. Functional requirements table for the routine panel included within the organization module (source: self)

Table 8. Functional requirements table for the forums panel included within the people module (source: self)

Table 9. Functional requirements table for the helpline panel included within the people module (source: self)

Table 10. Functional requirements table for the affirmations panel included within the mindfulness module (source: self)

Table 11. Functional requirements table for the graph tasks panel within the graph tasks module (source: self)

Table 12. Non-functional requirements table for accessibility within the application (source: self)

Table 13. Non-functional requirements table for security within the application (source: self)

Table 14. Non-functional requirements table for ethics concerning the application (source: self)

Table 15. Non-functional requirements table for performance within the application (source: self)

Table 16. Non-functional requirements table for maintainability within the application (source: self)

Table 17. Listing of users from the tested user group

Table 18. Results of tasks completion

Figures listing

- Figure 1. “(a) Normal brain and (b) ADHD brain with smaller volume” [7]
- Figure 2. “Scanned Image of Brain Showing Affected Portion due to ADHD” [9]
- Figure 3. Notion app homepage [33]
- Figure 4. Any.do app homepage [34]
- Figure 5. Numo app homepage [35]
- Figure 6. Remember The Milk app homepage [36]
- Figure 7. Diagram of system actors present the application (tool: UMLet [48], source: self)
- Figure 8. Use case diagram for the screening test module (tool: UMLet [48], source: self)
- Figure 9. Use case diagram for the screening test module, marking what has been implemented by the author (tool: UMLet [48], source: self)
- Figure 10. Use case diagram for the panels within the attention module (tool: UMLet [48], source: self)
- Figure 11. Use case diagram for the panels within the attention module, marking what has been implemented by the author (tool: UMLet [48], source: self)
- Figure 12. Use case diagram for the panels within the organization module (tool: UMLet [48], source: self)
- Figure 13. Use case diagram for the panels within the organization module, marking what has been implemented by the author (tool: UMLet [48], source: self)
- Figure 14. Use case diagram for the panels within the people module (tool: UMLet [48], source: self)
- Figure 15. Use case diagram for the panels within the people module, marking what has been implemented by the author (tool: UMLet [48], source: self)
- Figure 16. Use case diagram for the panels within the mindfulness module (tool: UMLet [48], source: self)
- Figure 17. Use case diagram for the panels within the mindfulness module, marking what has been implemented by the author (tool: UMLet [48], source: self)
- Figure 18. Use case diagram for the panels within the graph tasks module (tool: UMLet [48], source: self)
- Figure 19. Use case diagram for the panels within the graph tasks module, marking what has been implemented by the author (tool: UMLet [48], source: self)
- Figure 20. Use case diagram flow of the navigation (tool: UMLet [48], source: self)
- Figure 21. Use case diagram flow of the navigation, marking what has been implemented by the author (tool: UMLet [48], source: self)
- Figure 22. Paper prototype of the title screens (source: self)
- Figure 23. Paper prototype of the main screens (source: self)
- Figure 24. Digital prototype of title screen (on the left) and introduction screen (on the right) (tool: Marvel [51], source: self)
- Figure 25. Digital prototype of the home screen (tool: Marvel [51], source: self)
- Figure 26. Digital prototype of the Attention (Uwaga) homepage screen (tool: Marvel [51], source: self)
- Figure 27. Digital prototype of the Timer page (tool: Marvel [51], source: self)
- Figure 28. Wireframe implementation of the Timer page (source: self)
- Figure 29. Current final design of the Title screen (tool: Pixel 33 emulator, source: self)
- Figure 30. Current final design of the Introduction screen (tool: Pixel 33 emulator, source: self)
- Figure 31. Current final design of the Attention page (tool: Pixel 33 emulator, source: self)
- Figure 32. Current final design of the Organization module page (tool: Pixel 33 emulator, source: self)
- Figure 33. Current final design of the Routine Menu Page (tool: Pixel 33 emulator, source: self)
- Figure 34. Current final design of the Edit Routine view (tool: Pixel 33 emulator, source: self)
- Figure 35. System notification about a successful routine creation (tool: Pixel 33 emulator, source: self)
- Figure 36. Routine Menu Page view with newly created custom routine (tool: Pixel 33 emulator, source: self)
- Figure 37. Routine template page view populated with the data of the chosen routine (tool: Pixel 33 emulator, source: self)
- Figure 38. Custom routine view with some of the steps checked by the user (tool: Pixel 33 emulator, source: self)
- Figure 39. System notification about a successful routine completion (tool: Pixel 33 emulator, source: self)
- Figure 40. Current final design of the Attention module page (tool: Pixel 33 emulator, source: self)

Figure 41. Current final design of the Notes Menu page (tool: Pixel 33 emulator, source: self)

Figure 42. Current final design of the Edit Note view (tool: Pixel 33 emulator, source: self)

Figure 43. System notification about a successful note creation (tool: Pixel 33 emulator, source: self)

Figure 44. Current final design of the People module page (tool: Pixel 33 emulator, source: self)

Figure 45. Forums Page view (tool: Pixel 33 emulator, source: self)

Figure 46. System prompts the user for a deletion confirmation (tool: Pixel 33 emulator, source: self)

Figure 47. System notification about a successful routine deletion (tool: Pixel 33 emulator, source: self)

Figure 48. Homepage of GNU Operating System website (source: [54])

Figure 49. General use case diagram for the “CutCatADHD” application made with UMLet (tool: UMLet [48], source: self)

Figure 50. Homepage of GNU Operating System website (source: git homepage [58])

Figure 51. Homepage of SQLite website (source: SQLite homepage [63])

Figure 52. Section 7 of the GPLv2 license, the so-called "liberty or death" clause (source: GNU GPL v2 license [66])

Figure 53. Visual Studio starter screen presenting the option to create a .NET MAUI App (tool: Visual Studio IDE [70], source: self)

Figure 54. Github repository of .NET MAUI (source: GitHub/dotnet/maui [72])

Figure 55. Concise explanation of the chief difference between MIT and GPLv2 licenses (source: StackOverflow [73])

Figure 56. .NET MAUI architecture diagram (source: [74])

Figure 57. Illustrations browsing page of unDraw (source: unDraw [76])

Figure 58. Full license text of unDraw (source: [77])

Figure 59. Iconoir homepage (source: Iconoir [77])

Figure 60. Github repository of Iconoir (source: [79])

Figure 61. Homepage of Visual Studio website (source: Visual Studio webpage [70])

Figure 62. NuGet Package Manager window inside MS Visual Studio IDE (tool: Visual Studio IDE [70], source: self)

Figure 63. Source code of the TitlePage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 64. Source code of the TitlePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 65. Source code of the IntroductionPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 66. Source code of the IntroductionPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 67. Source code of the AttentionHomePage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 68. Source code of the AttentionHomePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 69. Reusable bottom navigation bar component in the AttentionHomePage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 70. Reusable bottom navigation bar component in the Views/Components directory (tool: Visual Studio IDE [69], source: self)

Figure 71. Source code of the BottomNavigationView.xaml (tool: Visual Studio IDE [70], source: self)

Figure 72. Source code of the BottomNavigationView.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 73. Source code of the ActivePageConverter.cs (tool: Visual Studio IDE [70], source: self)

Figure 74. Source code of the OrganizationHomePage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 75. Source code of the OrganizationHomePage.xamlcs (tool: Visual Studio IDE [70], source: self)

Figure 76. Source code of the RoutineMenuPage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 77. Source code of the RoutineMenuPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 78. Source code of the EditRoutinePage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 79. Source code of the EditRoutinePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 80. Source code of the Routine.cs model (tool: Visual Studio IDE [70], source: self)

Figure 81. Source code of the DatabaseDataAccess.cs (tool: Visual Studio IDE [70], source: self)

Figure 82. Source code of the DabaseConnection.cs (tool: Visual Studio IDE [70], source: self)

Figure 83. Source code of the OnSaveButtonClickedAsync function in the EditRoutinePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 84. Source code of LoadAndDisplayRoutines and DisplayRoutine functions in the RoutineMenuPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 85. Source code of the RoutineViewModel.cs (tool: Visual Studio IDE [70], source: self)

Figure 86. Source code of the RoutineTemplatePage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 87. Source code of the RoutineTemplatePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 88. Source code of the EditNotePage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 89. Source code of the EditNotePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 90. Source code of the Note.cs model (tool: Visual Studio IDE [70], source: self)

Figure 91. Source code of the NoteViewModel.cs (tool: Visual Studio IDE [70], source: self)

Figure 92. Source code of the DatabaseDataAccess.cs (tool: Visual Studio IDE [70], source: self)

Figure 93. Source code of the OnSaveButtonClickedAsync function in the EditNotePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 94. Source code of the PeopleHomePage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 95. Source code of the PeopleHomePage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 96. Source code of the ForumPage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 97. Source code of the ForumPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 98. Source code of the RoutineMenuPage.xaml (tool: Visual Studio IDE [70], source: self)

Figure 99. Source code of the DeleteRoutine function in the RoutineMenuPage.xaml.cs (tool: Visual Studio IDE [70], source: self)

Figure 100. Bar graph presenting the results of task 1 - creating a custom routine

Figure 101. Bar graph presenting the results of task 2 - completing the custom routine

Figure 102. Bar graph presenting the results of task 3 - creating a custom note

Figure 103. Bar graph presenting the results of task 4 - visiting one of the forums

Figure 104. Bar graph presenting the results of task 5 - deleting created earlier custom routine

Figure 105. Bar graph presenting generals impressions of the application