**Bachelor Project**

**Czech Technical University in Prague**

**F3** Faculty of Electrical Engineering

# Application for personal development

**Artem Vanyukhin**

**Supervisor: Ing. Pavel Náplava, Ph.D.**
**Field of study: Software Engineering and Technology**
**May 2021**

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Vanyukhin**  Jméno: **Artem**  Osobní číslo: **439831**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

## II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

**Aplikace pro podporu osobního rozvoje**

Název bakalářské práce anglicky:

**Application for Personal Development**

Pokyny pro vypracování:

Vytvořte první verzi aplikace pro podporu osobního rozvoje, která bude založena principu vzájemné motivace skupiny uživatelů. Postupujte následujícím způsobem:
1) Analyzujte doménu osobního rozvoje, především pak motivaci na principu
soutěživosti (například porovnávání dosahování definovaného cíle mezi
přáteli).
2) Proveďte rešerši existujících aplikací v dané doméně.
3) Navrhněte vlastní aplikaci, která bude reflektovat standardy již existujících aplikací a požadavky, vycházející z předchozí analýzy.
4) Pro popis návrhu aplikace použijte standardní nástroje a postupy softwarového inženýrství (technická analýza: high-level design, datový model atd.).
5) Navrženou aplikaci implementujte minimálně v první, funkční verzi, která umožní alespoň evidenci, sledování a porovnání aktivit mezi uživateli.
6) Funkčnost aplikace ověřte prostřednictvím uživatelského testování.

Seznam doporučené literatury:

[1] Orosz G, Tóth-Király I, Büki N, Ivaskevics K, Bothe B and Fülöp M &quot;The Four
Faces of Competition: The Development of the Multidimensional Competitive
Orientation Inventory.&quot; Front.Psychol., 2018.
[2] Ryckman, R. M., Hammer, M., Kaczor, L. M., and Gold, J. A. &quot;Construction of a
personal development competitive attitude scale.&quot;, 1996.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

**Ing. Pavel Náplava, Ph.D.,   katedra softwarového inženýrství   FIT**

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce:  **11.02.2021**  Termín odevzdání bakalářské práce:  **21.05.2021**

Platnost zadání bakalářské práce:  **30.09.2022**

_____  _____  _____
Ing. Pavel Náplava, Ph.D.  podpis vedoucí(ho) ústavu/katedry  prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce  podpis děkana(ky)

## III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

.
_____                    _____
Datum převzetí zadání                                          Podpis studenta

# Acknowledgements

I would like to thank my supervisor Ing. Pavel Náplava, Ph.D. for his guidance and valuable input. I would like to thank my friends for their feedback: Sofya Lartseva, Ryan Awo, Jelizaveta Zimina, Chistiaan Dekkers, Nail Al-Gadyan.

# Declaration

I hereby declare that the present thesis was composed by myself and was not previously included in a thesis or dissertation.

In Prague on May 20, 2021

# Abstract

This paper researches the domain of personal development and the possibility of using competitiveness as a motivation for personal development. Performed research serves as a basis for the analysis of existing solutions and proposition of a new application. Based on this analysis the first version of application was implemented and tested.

**Keywords:** personal development, competitiveness, Kotlin, Spring, React

**Supervisor:** Ing. Pavel Náplava, Ph.D.

# Abstrakt

Tato práce se zabývá rešerši domény osobního rozvoje a možnosti využití soutěživosti k motivaci osobního rozvoje. Provedená rešerše slouží základem k analýze existujících řešení a následujícímu návrhu nové aplikace. Na základě tohoto navrhu byla implementována a otestována první verze aplikace.

**Klíčová slova:** osobní rozvoj, soutěživost, Kotlin, Spring, React

**Překlad názvu:** Aplikace pro podporu osobního rozvoje

# Contents

# Figures

# Tables

# Chapter 1

## Introduction

This chapter will introduce the reader to author's motivation, problem description and the basic idea of the future application.

## 1.1 Motivation

From elementary school to the final year of university studies, one was able to compare themselves with peers. In some cases this could be the source of strong motivation that stimulates a continuous process of learning. In order to keep myself from falling into a daily routine of work at the completion of my studies and to be able to continue to actively study, I decided to look into applications for personal development.

Prompt research demonstrated that applications with social engagement elements tend to be well accepted by users but also revealed a shortage of such applications. Those that complied with the social engagement criterion lacked crucial functions that could be beneficial for the process of personal development. Also, only a small amount of them used competitiveness as a motivation for personal development.

This paper will shortly examine the domain of personal development and justify that friendly competition might be used as a tool for personal development; compare existing solutions and provide an analysis for a new application.

## 1.2 Problem description

Personal development is an ongoing process. As well as many other activities, this process is more effective with a defined and measured approach. There are many well-built solutions aimed to structure the process of personal development. These solutions have a variety of ways to approach their users thus user(s) could find an application that works for them. The problem with these applications is the almost nonexistent utilization of social engagement. Users could rely exclusively on their motivation and discipline in order to use such application(s) effectively and on a long-term basis.

Another type of personal development applications are habit trackers. These applications aim at the development of newly desired habits and discarding redundant ones. Habit trackers often include gamification elements and usually there is minimal social engagement elements. There are also habit trackers that use the idea of friendly competition which introduces higher level of social engagement. On the other hand, these applications usually lack a structural approach, visual reports and planning features.

A new application that motivates a user through the interaction with others, capable of long-term activity planning and repetitive habit management should be proposed. This will take the best from both application types mentioned above and could provide better user experience. The problem that this paper is trying to solve is an analysis and proposal of such application.

## ▉ 1.3 Solution idea

In order to use the best parts of previously mentioned application types, two main aspects should be designed thoroughly. The first is the ability of the application to provide an effective way of data entry and management. The second is the high but optional level of social engagement.

The fist objective, is going to be achieved by separation of activities on two types - goals and habits. Goals will provide an interface for input and organization of targets that a user would like to pursue. A goal's progress is going to be measured by completion of tasks. Habits, on the other hand, will provide an interface for repetitive activities that do not require planning.

The second objective is to increase user's motivation through members of their social circle. This is going to be achieved through activity sharing. Optionality of this feature would be available through visibility setting in goals and habits. This separation will provide a substantial amount of social engagement.

The biggest challenge of the described application is the design of an intuitive and minimalistic user interface.

# Chapter 2

# Personal development domain

This section will introduce the reader to the domain of personal development. Justification of the idea behind the application will also be presented here.

## 2.1 Definition of personal development

In order to create a helpful application, it is crucial to build it based on principles and ideas of personal development. Familiarization with these will start from the definition of personal development and with Maslow's hierarchy of needs in particular.
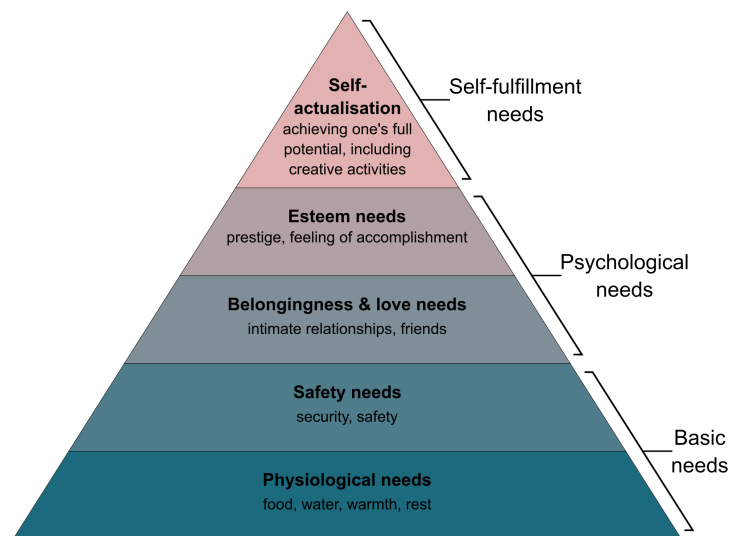


**Figure 2.1:** Maslow's Hierarchy of Needs [2]

The pyramid shown in figure 2.1 might be familiar to the reader. It was first introduced by A. Maslow in 1943.[1] This pyramid represents the hierarchy of common human needs. It may be presumed, that a person has satisfied lower levels before beginning to satisfy the higher ones. Going from the bottom to the top, the first two levels are basic needs that every person requires the most. The following two levels are psychological needs, these come after physical ones. At the top of the pyramid there is a need for

*self-actualization.* Maslow described it as restlessness that often develops even with satisfied needs. Self-actualization refers to the desire of reaching one's full potential and could be considered as the goal and motivator for the process of personal development. As the pyramid contains highly relatable needs and was popularized in modern culture, it is safe to assume that described needs will be known to the reader.

*Personal development* is the set of activities aimed at enhancement of employability, improvement of quality of life and raising one's confidence.[3] Motivation for personal development comes from the need for self-actualization and thereby makes it a natural and relatable feeling. Five following steps should be considered when managing one's personal development hence should be attended in the application: personal vision development, personal development planning, recording personal development, reviewing and revisiting personal development plans.

*Development of personal vision* means having an idea of what a person wants to become in the future. Decision-making cannot be helped by the application because of highly personal and case-specific nature of the process. On the other hand, results of those decisions will be managed by the application; those will be the source of the goals. One might consider goals as personal projects. Due to simplicity of the action, a new goal creation should also stay simple and require the lowest possible amount of mandatory fields.

After personal vision development comes *personal development planning.* This step is optional but crucial for long-term goals with a large amount of different milestones. For example, learning new skills might be more efficient with a plan rather than with an ad-hoc approach. This also opens an opportunity for a plan sharing feature. Planning would be obsolete in the case of simple and repetitive activities, which emphasizes the need for optionality of this feature.

*Recording personal development* progress is always a good idea as it makes the process measurable. This is also one of the main points of the future application. Activity record opens opportunity for both: data visualization and social engagement.

*Reviewing and revisiting personal development plans* will allow one to learn more from the past activities. Another benefit is that the record might allow the user to define a comfortable learning pace, improve future experience with the application and personal development in general. For the application, this means that there should be an easily accessible history of past goals and activities.

Steps from above describe the recommended approach to the personal development. Those steps also define a data management flow and processes of the application. Domain-based approach will provide better user experience when it comes to the process planning and data input.

## 2.2 Competitive attitude

Whereas the previous section created a basis for data management and data flow, this section will introduce justification for competitiveness as an instrument for personal development. Social engagement element of the application will be based on this justification.

In psychology, competition was considered a unidimensional scale for a long time - up until the 1990s. Cooperation could be seen as the opposite of competitiveness. A research article "The Four Faces of Competition" summarized the development and research of recent decades and introduced four-dimensional scale for competitiveness.[4] This scale is heavily based on works of R.M.Ryckman.[5, 7, 6] Ryckman studied competitiveness as attitude rather than personal quality. Overall, there were four identified attitudes regarding competition: hypercompetitive, anxiety-driven competition avoidant, self-developmental and lack of interest toward competition.

*Hypercompetitive* attitude is represented by an aggressive approach to competitive situations. One with hypercompetitive attitude has a goal of winning any competition they participate. Winning could be gained even by unfair means. This dominant attitude is destructive for personal relationships and considered psychologically unhealthy. It is also associated with low self-actualization, low self-esteem, high aggression and dominance.[5]

*Anxiety-driven competition avoidant* attitude is the opposite of hypercompetitive attitude. A person with competition avoidant attitude will try to avoid entering competition by all means. Their anxiety is derivative of the general process of competition rather than the fear of loosing. Such attitude correlates with lower self-esteem and lower optimal psychological health overall.[7]

*Self-developmental* attitude or the *personal-development competitive attitude* (PDCA) takes competition as an opportunity for personal development. A person with dominant PDCA enjoys the process of competition but in another way than one with dominant hypercompetitive attitude. For a PDCA dominant person, enjoyment comes from the process of skill improvement during a competition. Winning also takes an important part but not at the price of cheating or derogation of other competitors. This trait is associated with higher self-esteem, high self-actualization, task enjoyment and self-discovery.[6]

Finally, *lack of interest toward competition* differs from the above mentioned. It simply describes disinterest in competitions and lack of any approach or avoidance.

These traits could be combined with each other, as not all of them are mutually exclusive. For example, one could have lack of interest as a primary trait and PDCA as a secondary. A person with such traits, would usually ignore competitive opportunities but when engaged into one would use it as a possibility for personal growth and enjoy it.[4]

As it was revealed by the research, competition might be used as a tool of personal development. Individuals with PDCA as a primary or secondary

trait might benefit from an application that will utilize their healthy attitude toward competition. Since competitions in the application will be created by users themselves and for their close social circle, it could be helpful for ones with anxiety-driven competition avoidance. Friendly competitions in the controlled environment, might help to overcome their anxiety. Individuals with hypercompetitive attitude probably will not benefit from the social part of application as they might take these competitions too seriously. Nonetheless, the application still remains useful for them as a tool for personal development planning.

## ■ 2.3 Competitive attitude

The domain research provided an understanding into the personal development process. Based on this, data management and application flow will be proposed. The research of competitiveness justified that a competition might be used as a motivator for personal development.

# Chapter 3

## Existing solutions

This section will introduce the reader to the analysis of existing solutions.

## 3.1 Comparison criteria

Based on the previous chapter, core features for the application are as follows: personal development planning and social engagement. The application should be multi-platformed as well.

*Personal development planning* encapsulates the features which the application provides for a structured approach to personal development. This should handle repetitive activities as well as long-term plans. Progress tracking and possibility to review old plans are also a part of that criteria.

*Social engagement* criterion includes features that the application provides for interaction between users. It might be the sharing of goal(s) progression or mutual goals of different users that are pursued simultaneously.

*Multi-platform* is a criterion required by the idea of a shared user experience. Limited platform availability might be a problem for some groups of users, e.g, Android and iOS Only web, Android and iOS platforms are considered with regard to this project. Windows, Macintosh and Linux are not going to be considered due to general decrease in popularity of desktop platform.[9]

## 3.2 Data collection

Data collection was done using Google search engine as it was the most popular search engine in 2020 with over 90% market share.[8] The first step was to find applications using following keywords: "personal development apps", "habit tracker", "self-improvement apps". The top five results for each keyword were used to gain the initial dataset. The next step was to manually filter gathered data. The first step of filtration was to manually remove duplicates. The second step was to remove activity specific applications: sport, meditation, brain training. The third step was to remove applications with static textual content on topics of psychology, personal development and life improvement techniques. The fourth step was to remove applications aimed exclusively at personal care, skincare, mood journals and sleep journals. The fifth step

| Action | K1 | K2 | K3 | Total results | Difference |
|---|---|---|---|---|---|
| Initial dataset | 52 | 38 | 62 | 152 | +152 |
| Duplicates | 46 | 25 | 23 | 94 | -58 |
| Sport oriented | 41 | 23 | 22 | 86 | -8 |
| Meditation oriented | 38 | 19 | 19 | 76 | -10 |
| Brain training oriented | 25 | 16 | 18 | 59 | -17 |
| Topics oriented | 15 | 16 | 10 | 41 | -18 |
| Personal care oriented | 5 | 11 | 7 | 23 | -18 |
| Business and productivity oriented | 4 | 11 | 5 | 20 | -3 |
| Bad habits oriented | 4 | 10 | 1 | 15 | -5 |
| Single platform | 2 | 5 | 1 | 8 | -7 |
| Similar applications | 2 | 2 | 1 | 5 | -3 |

K1="Personal development apps", K2="Habit tracker", K3="Self-improvement apps"

**Table 3.1:** Dataset filtering

was to remove business and productivity applications aimed primarily at small scale time management, short-term goals and daily productivity. The sixth step was to remove applications aimed strongly at bad habit breaking, such as smoking and drinking. The seventh step was to remove applications available for single platform exclusively, e.g., iOS. The final step was to remove applications that are very similar in terms of interface, user experience and functionality.

The result of filtering is 5 applications that will be analyzed in the next section. Details of the dataset filtering process are described in Table 3.1. *Action* column contains the removal criteria during a filtration step. *K1* to *K3* columns contain the amount of applications by a keyword. *Total results* column contains the sum of K1, K2 and K3 columns. *Difference* column contains the difference of total results against previous row.

## ◼ 3.3 Data analysis

The following 5 applications will be tested and analyzed in this section: Uloo, Habitica, Habitify, TickTick, Habitshare. Android and PlayMarket are going to be used as a testing platform. Downloads, reviews and overall score are captured at the time of writing that is October 2020.

### ◼ 3.3.1 Uloo



**Figure 3.1:** Uloo logo [10]

Uloo is a goal pursuing application. It has features for goals setting, progress tracking, communication between users, common and private goals tracking and paid mentoring by coaches. It is not really suitable for simple repetitive tasks. Another demerit is that it is not usable without a paid premium subscription. Functions are very limited in a free version. Even with a premium subscription, it is not possible to create personalised goals. A core feature, which is goal's progress tracking, demonstrated a breaking bug during the test of functionalities.

| Pros | Cons |
|---|---|
| Can start discovering without an account | Unusable without a paid premium |
| Fair UI/UX | Only predefined goals |
| Time or count as | Squad size limited to 8 |
| progress measurement | Small amount of |
| Squads for goals pursuing in teams | installs and reviews |
| Chat in squads | Bugs and unstable behaviour |

**Table 3.2:** Uloo pros and cons

Uloo has between 1,000–2,000 downloads and 28 reviews with an average score 4.2 out of 5 on PlayMarket. Uloo's implementation of progress sharing and squads might serve as an inspiration. Limitation to only predefined goals is an example of not giving enough freedom to a user and should be avoided in the new application.

### 3.3.2 Habitica



**Figure 3.2:** Habitica logo [11]

Habitica is an application for habits, daily goals and To-Do's management. It has features for interaction between users. Habitica is a great example of gamification. Application is presented in a role-playing game style. Users develop their game characters by executing daily chores and pursuing goals. What Habitica lacks is long-term goal management. Another potential problem is that the amount of gamification in the app would not work for many people, especially those who are not interested in video games.

Habitica has over 1,000,000 downloads and 17,334 reviews with an average score of 4.4 out of 5 on PlayMarket. Habitica is a well-developed application with a large user base. Core idea of Habitica is to convert daily routines into a game, which is a feature that would not work for everyone. Habitica's strong aim at gamification and overall UI makes it irrelevant in terms of the future

| Pros | Cons |
|------|------|
| Great UI/UX | May be too much of gamification |
| Many free features | for some users |
| Gamification | Not suitable for long-term goals |
| Social features | |

**Table 3.3:** Habitica pros and cons

application design. On the other hand, it shows importance of possibility to interact between users which is a great part of Habitica.

### ■ 3.3.3 Habitify



**Figure 3.3:** Habitify logo [12]

Habitify describes itself as a data-driven habit tracker. It provides strong data visualization and activity planning features. Another strong feature is synchronization with multiple devices such as smartwatches and integration with health applications. Application does not provide social elements and UI might have been more interesting. The biggest problem is a very limited functionality of the free version.

| Pros | Cons |
|------|------|
| Data visualization | Social element is missing |
| Devices synchronization | Unusable without a paid premium |
| Health applications integration | Unengaging UI |

**Table 3.4:** Habitify pros and cons

Habitify has over 100,000 downloads and 1,137 reviews with an average score 3.9 out of 5 on PlayMarket. Data visualization in this application serves as a good inspiration. On the other hand, UI is an example of minimalistic approach done wrong with a disengaging and empty feel overall.

### ■ 3.3.4 TickTick

TickTick is an activity planner application. It provides lists, reminders and calendars as well as tools for activity planning. These tools might be used for long-term plans or repetitive activities. It is also possible to share lists between users. TickTick was not primarily designed as a personal development

**Figure 3.4:** TickTick logo [13]

application. However, still might be used as one. Loaded variety of fields and functions creates an exhausting user experience.

| Pros | Cons |
|:---:|:---:|
| Usable without an account | Large amount of typing |
| Shared lists | Too many options |
| Good UI | for data management |
|  | Questionable UX |
|  | for personal development |

**Table 3.5:** TickTick pros and cons

TickTick has over 1,000,000 downloads and 67,692 reviews with an average score 4.6 out of 5 on PlayMarket. Despite being a well-received application for activity planning, it is not the best option for personal development. Giving too much freedom and functions to a user might decrease usability of an application and should be avoided.

### 3.3.5 Habitshare



**Figure 3.5:** Habitshare logo [14]

Habitshare describes itself a social habit tracker. It utilizes user's social circle for extra accountability. User creates habits they want to track and do so using calendar. The application is only suitable for simple and repetitive activity tracking.

| Pros | Cons |
|:---:|:---:|
| Highly social | Not usable without an account |
| Usable without paying | Not suitable for long-term goals |

**Table 3.6:** Habitshare pros and cons

Habitshare has over 100,000 downloads and 658 reviews with an average score 4.5 out of 5 on PlayMarket. Social part of application seems to be well accepted by users as well as simple and minimalistic UI.

## ■ **3.4  Conclusion**

As the analysis showed, there is a variety of applications suited for tracking long-term goals or repetitive activities. Only TickTick proved to be suitable for both. However, it has shortfalls as well.

Social engagement was efficiently utilized in Habitify and Habitica but those applications lack the ability to track anything except for habits.

In conclusion, there is an evident gap that could be filled with a new application that will take social engagement even further and provide features for tracking long-term goals and repetitive activities.

# Chapter 4

# Solution analysis

This chapter will outline a solution analysis for a new application. Based on the previous chapters, there will be introduced entities of the domain and their lifecycle. Later there will be the description of functional requirements.

## 4.1 Overall description

Based on the research and analysis introduced in the previous chapters, following specification will describe an application that fits the criteria. Emphasised words are entities that are present in the domain model and are going to be revealed in the following section.

The application manages two types of personal development activities: goals and habits. *Goal* is an activity with a defined start to finish. It consists of *tasks*, which makes it possible to track the progress of a goal. Goal is built from *goal plan*, which allows separation of goal creation from fulfillment. That opens the possibility to share instructions on how to achieve the goal. When applicable, the goal plan might be created by a person experienced in the field related to the goal. *Habit* is a repetitive activity. It does not require decomposition to smaller tasks. Habit is fulfilled according to a schedule. Described separation to habits and goals allows users to select a more suitable data management form for their use case, which would usually require two different applications.

As mentioned prior, the application will engage users through their close social circle. To do so, there are goal and habit related *activities*. An activity contains text with relevant information on habit streaks or goal's progress. Visibility of activities between users will be driven by habit and goal settings; they might be private, shared between all friends or just with selected users. In order to provide a simple communication tool, it is possible to leave *commentaries* under activities.

*Daily progress* on goals and habits is gathered in order to provide user's with statistics. It is also used for weekly and monthly statistics calculation.

## 4.2   Domain model

This section will describe entities of the application domain model displayed in figure below. Some technical information such as data types are omitted for brevity. Each entity has information about creation date, modification date etc.
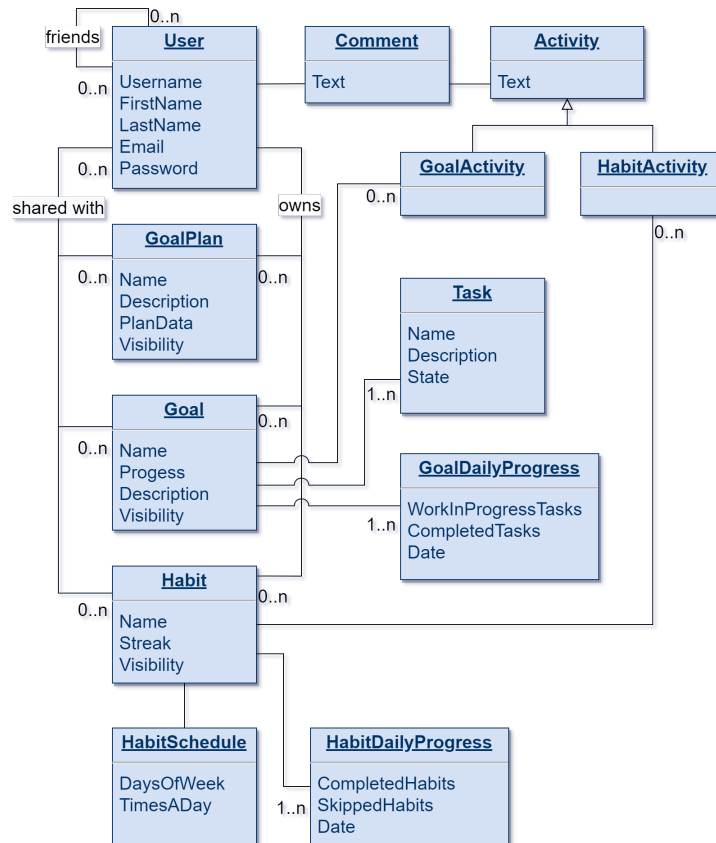


**Figure 4.1:** Domain model [author]

### 4.2.1   User

User entity consists of minimum required attributes in order to simplify registration process for end-user. Recursive relation represents friendship between users, which is required for activity sharing. Goal plans, goals and habits are always owned by one user and might be shared with other users. Sharing is used for displaying private plans, goals and habits with specific users.

### 4.2.2   Goal plan

Goal plan serves as a blueprint for a goal. This separation of a plan and a goal allows users to create and to share not only their progress but also plans.

14

Possible use case would be as follows: a user is proficient in a specific field or skill that others' would like to learn or gain. They create a goal plan with tasks that contains instructions and share it with any amount of users. Other users then are able to start a goal pursuing process using this shared plan.

### ▪ 4.2.3 Goal

Goal is a tool for managing long-term targets. It is also a specific case of fulfilling a goal plan that is created by user or shared with them.

### ▪ 4.2.4 Task

Task is a building block of a goal. Task completion used for calculation of a goal's progress.

### ▪ 4.2.5 Goal daily progress

This entity holds data about daily progress on a specific goal during given day. It is then used for statistics calculation.

### ▪ 4.2.6 Habit

Habit is a tool for managing simple and repetitive targets.

### ▪ 4.2.7 Habit schedule

Each habit has a schedule. Initially, it is filled with default values but possible for users to change them.

### ▪ 4.2.8 Habit daily progress

This entity holds data about daily progress on a specific habit. It is then used for statistics calculation.

### ▪ 4.2.9 Activity

This entity holds data about user activities on goals and habits. When a goal or habit is shared with a user, they will see related activities.

### ▪ 4.2.10 Comment

It is possible to comment on activities in application and reply to commentaries. All users with access to activities, can see others' commentaries.

# ■ **4.3  Functional requirements**

This section will introduce requirements regarding the application functionalities.

Requirements have their priority estimation. High priority feature is a "must have" and necessarily must be present in the application. Medium priority feature is a "nice to have" and should be implemented for the best user experience. Low priority feature is a "might have" which takes user experience level even further.

Complexity estimation is relative. Medium might be considered as average time to implement a feature. Low complexity level requirements takes a noticeably shorter period of time to implement. High complexity level requirements as the opposite will take longer period of time.

## ■ **4.3.1  User registration and login**

**Complexity:** High
**Priority:** High
**Description:** In order to safely store and access data in the cloud, user needs registration and login features.
**Note:** This requirement brings the need for thorough security configuration.

## ■ **4.3.2  User login with a facebook account**

**Complexity:** High
**Priority:** Medium
**Description:** Login through a Facebook account needed to minimize registration effort for a user.
**Note:** Meeting the Facebook's security criteria might take an additional effort.

## ■ **4.3.3  Friend list**

**Complexity:** Low
**Priority:** High
**Description:** Users need the ability to add each other to friend lists in order to share activities with each other.

## ■ **4.3.4  Application is usable without a registration**

**Complexity:** High
**Priority:** Low
**Description:** Ability to try the application without a registration improves experience for a user.

**Note:** This requirement will take a large amount of time for analysis therefore increasing overall complexity of the system.

### 4.3.5   Habits management

**Complexity:** Medium
**Priority:** High
**Description:** Users need an interface to manage repetitive activities.

### 4.3.6   Goals management

**Complexity:** Medium
**Priority:** High
**Description:** Users need an interface to manage long-term goals.

### 4.3.7   Goal plans sharing

**Complexity:** Medium
**Priority:** Medium
**Description:** Users need an interface to create and share goal plans.

### 4.3.8   Visibility of habit, goal, goal plan

**Complexity:** Medium
**Priority:** High
**Description:** Users need an ability to change visibility of their data. There will be two levels of visibility: private and public.

### 4.3.9   Activity history

**Complexity:** Medium
**Priority:** High
**Description:** Users need an interface to see their activities on habits.

### 4.3.10   Activity sharing

**Complexity:** High
**Priority:** Medium
**Description:** Users need an interface to see activities of other users.
**Note:** This requirement brings the need of visibility management for goals and habits.

### ◼ **4.3.11   Basic statistics calculation**

**Complexity:** Medium
**Priority:** Medium
**Description:** Users need an interface to see the statistics related to their habits and goals fulfillment.

### ◼ **4.3.12   Gamification elements by activity points**

**Complexity:** Low
**Priority:** Low
**Description:** Each executed activity related to the habit gives the user one point. Activity points displayed in user profile.
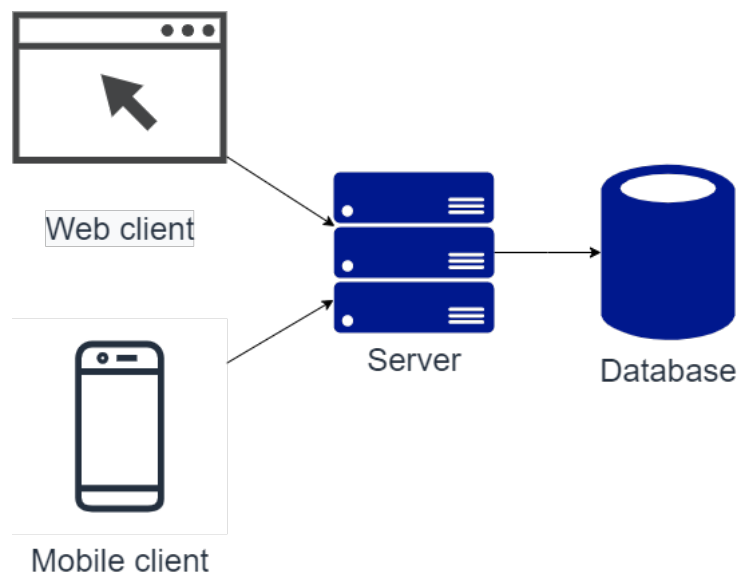
## ◼ **4.4   Software architecture**



**Figure 4.2:** High-level software architecture [author]

Software architecture has been proposed with scalability and multiple types of client applications in mind. As shown in figure 4.2, server is a standalone application separated from client applications. Such separation allows sharing business logic across different client applications without the need of a rewrite. Another benefit is the simplicity of scaling. New instance of the server application might be deployed behind load balancer as needed.

More specifically application is built using Model-View-Controller (MVC) architecture.[15] View component in MVC is a representational layer. User interacts with the application through mentioned layer. In this instance, application View component is represented by client applications. Model

component represents business logic and Controller serves to bind View and Model. Model and Controller components are encapsulated in server part of this application.
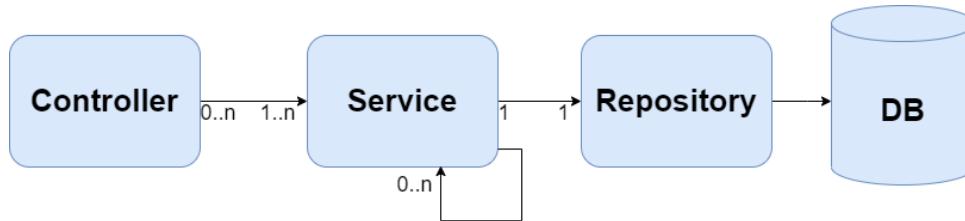


**Figure 4.3:** Server architecture [author]

Detailed server architecture displayed in figure 4.3. Repository serves as a Data Access Object (DAO). Each DAO defines the set of allowed operations over a specific entity of the domain model. Service is an implementation of business logic. Each Service has exactly one related Repository. Other entities Repositories should be accessed via their Service component, hence 0-to-n cardinality from Service to Service. Complex business logic scenarios with several Service components calls might be encapsulated using the facade design pattern. Controller creates API endpoint mapping, e.g. in www.domain.com/users '/users' is an endpoint mapping.

Described architecture is supported out of the box by major web development frameworks as Spring (Java/Kotlin), Django (Python), Rails (Ruby).[16, 17, 18] etc. This, alongside with preservation of the single-responsibility principle [19] enforced by the idea of MVC, allows for rapid development of maintainable applications.

## 4.5 Technological stack

Technological stack was considered with developer experience and free of use in mind. Popularity, community and performance were also taken into account.

### 4.5.1 Server side

Kotlin with Spring framework was selected as the server side language and PostgreSQL as the database.[20, 16, 21] Kotlin is a modern programming language that compiles to Java compatible bytecode. This allows to use any Java library with Kotlin.

There are many reasons to use Kotlin over Java. Some of them are null safety, final variables and classes. This helps to write more robust code. Boilerplate code is significantly reduced in comparison with Java. Code written in Kotlin is shorter.

Following code snippet is written in Java.

```java
public class Person {
    private String firstName;
    private String lastName;

    public Person(String firstName, String lastName) {
        this.firstName = firstName;
        this.lastName = lastName;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
}
```

Kotlin equivalent for that snippet would be the following.

```kotlin
class Person(var firstName: String, var lastName: String)
```

Another benefit of Kotlin is the strong influence of functional programming, resulting in convenient features such as scope functions, high order functions and lambda expressions.

In regard to downsides, Kotlin's clean builds take longer to compile.[22]

### 4.5.2 Frontend

React was selected as a frontend framework with Material-UI as a design library.[23, 24] React is developed by Facebook and positions itself as a UI library, which means further libraries such as Axios (client-server communication) and Redux (state management) might be needed.[25, 26] Material-UI is a library with React components built according to Google's UI design guidelines. Combination of React with Material-UI is responsive out of the box, which means web client will be displayed correctly on mobile devices without an extra amount of work.

Also, it is relatively less complicated to prototype a mobile client using React Native in the future.[27] React Native is a UI library that allows the creation of UI for Android and iOS mobile devices. Code written in React Native is very similar to React code, some components could even be reused. React Native code then transpiled to platform native UI code for Android or iOS.

TypeScript was also considered as frontend technology but idea was dismissed due to insufficient time resources. Benefit of TypeScript is strong typing which JavaScript lacks.[28]

### 4.5.3 Deployment

Heroku cloud platform was selected as a deployment option.**??** Simple, minimalistic interface and straightforward deployment via command-line interface are the benefits over Amazon AWS or Google Cloud. Pricing is affordable only for proofs of concept, MVPs and small projects. Heroku is not the best choice for large application or high throughput application as the pricing goes 2 to 3 times more expensive than with Amazon AWS.[30]

## 4.6 Wireframes

Wireframes serve as a schematic representation of the user interface. The main target is to display what functions will be available on which pages. Wireframes are not intended to represent the looks of the future application. Implemented user interface is slightly different as it has to be mobile friendly. The example of a wireframe and several application's screens is located in the Appendix A.

# Chapter 5

## Implementation

This chapter will describe the development process of the first version of the application. All high priority functional requirements were implemented. Also, some medium priority requirements, such as statistics calculation were implemented.

## 5.1    Backend

The most useful resources during the development were Spring Framework documentation and Baeldung.[31, 32]

Project was configured as a multi-module Maven project. The next step was security configuration. Spring Security uses the chain of responsibility principle for security filters. After implementation of Json Web Token provider, new security filter was created and added to the chain.

Development started from the server side. First entities of the domain model were implemented. Using Hibernate via Spring database schema was generated automatically. Spring Data repositories with default operations were defined over entities. Services with the core parts of business logic were implemented and unit tested. Controllers were created to access created services. Postman was used for integration testing of created endpoints.

## 5.2    Web client

The most useful resources for the web client development were documentations for React and Material-UI.[23, 24]

Web client serves as a representational layer over server endpoints. React web applications are built based on a single-page application (SPA) model. The main idea of SPA is to have one HTML page that will dynamically change its content using JavaScript. Benefits of this model are faster transitions and more fluent feel overall. Downside of SPA is more complicated search engine optimization. Search engines web crawlers were historically designed to work with HTML served content, but with SPA content is served with JavaScript.

In React application there is one HTML file where the main "App" component is placed. Using React Router different page components might be

served. Page components are built by smaller reusable components, e.g. list of habits for today might be reused for upcoming habits. These reusable components might be also reused in mobile application prototyping.

The implemented client was ad-hoc tested during the development against local instance of the server. Later it was tested by users.

## ■ 5.3 Deployment

Solution was deployed to the Heroku cloud.[29] Applications there are deployed to "dynos". Dyno on Heroku is a virtualized Linux container. Containers provide isolated environment which brings several benefits. The first is separation from the infrastructure. This simplifies the process of deployment. The second is the ease of scalability. New containers might be deployed as needed to balance the incoming traffic.

Process of the deployment is straightforward. Given that Heroku command-line interface is installed and user is authenticated. First Heroku should be initialized in the project's root directory.

```
heroku create
```

This will add new Git remote called "heroku" by default. Deployment process will start simply by pushing code changes to the heroku remote.

```
git push heroku master
```

The entire process might be monitored through server logs.

```
heroku logs --tail -a $APP_NAME$
```

Deployed web client was available on https://pda-web.herokuapp.com/ at the time of writing.

# Chapter 6

## Testing

This chapter will describe testing methods applied during the development and post deployment.

## 6.1 Development testing

The purpose of development testing is to verify expected behavior and ensure that changes to the application will not break existing functionality.

Development testing included unit testing, integration testing and ad-hoc web client testing. For server testing in-memory H2 database was used. This ensured data consistency every test run. Unit tests covered Service components functionality to verify expected behavior.

Integration testing was done using Postman which is an application for REST API testing. This application allows joining multiple requests to test scenarios. Benefit of this approach is that the server first can be tested locally and then again after the deployment in production.

## 6.2 User testing

The goal of the user testing is to identify usability issues and critical application design flaws. In given context, a critical design flaw would mean that some core features of the application will make application unusable for a significant amount of users. Five users were familiarized with the application under supervision using steps from the list below.

1. Go to https://pda-web.herokuapp.com/

2. Create an account

3. Sign in to your account

4. Create a habit

5. Complete some habits for today

6. Try to edit your habit (change days of week, times a day etc.)

7. Create a goal

8. Mark some goal tasks as finished

9. Try to edit your goal (add tasks, change order etc.)

10. Add a new friend (try typing username 'artem')

11. Check your friend requests on "Friends" page

12. After friend request accepted check "Activities" page for friend's activities

13. Check "Statistics" page

14. On "Statistics" page generate a habit to fill the graph

15. Delete generated habit "Statistics test"

After familiarization session users were left to use the application for the next week. Some bugs related to streak reset and statistics calculation were found during user testing. Below is the list of selected user notes order by priority from the most important. Some was not included as they were caused by the lack of "Profile" page which was in the design, but was not yet implemented.

1. **Edit progress from previous days.**
   This request will help to keep users engaged in case they forget to mark their progress. They will lose the streak and might as well lose interest in the application otherwise. It is possible to implement this feature with the current domain model, as the information about each habit fulfillment for each day is kept in "Habit daily progress" entity.

2. **Leaderboards of habits fulfillment.**
   This request works well with the idea of social engagement. Deeper analysis of habits with shared progress between users is needed.

3. **Groups of users.**
   This request is a natural development of the friend list idea. Users should be able to create groups where they could share habits, work on them together and see each other progress. The concept of leaderboards works well with this request.

4. **Counter on habits.**
   Currently, when habit is repeated multiple times after marking one as done, the following will popup. Adding a counter that will show how many repeats are done and how many more to go will improve user experience. This request could be simply implemented with the current domain model.

5. **There should be a feature of occasionally doing more repeats of a habit.**
   There is no simple way to occasionally mark more repeats on a habit. This request requires additional analysis before implementation.

In conclusion, user testing helped to discover some bugs and gave several ideas to improve the application. All users agreed on the fact that the main way of using this application would be via the mobile client. No critical design flaws were identified during user testing. Collected information would be valuable for the future implementation of the mobile client as well as improvement of current web client.

27

# Chapter 7

## Conclusion

The domain research provided an understanding into the personal development process. This understanding allowed proposition of the data management. The second part of the domain research justified that the competitiveness might be used as a motivator for personal development.

Analysis of existing solutions allowed detecting common pitfalls and gave ideas for features that might improve overall user experience. This analysis also demonstrated uniqueness of the application idea and therefore proved valuable for further solution analysis and implementation.

Based on the research and analysis of existing solutions, the solution analysis was created. Research of the domain served as the basis for domain model. Functional requirements were mostly derived from the analysis of existing solutions.

Created analysis served as a basis for the first version of the application. User testing verified created concepts and provided valid ideas for the future analysis and development.

The next step would be the analysis of information gathered from the prototype development and testing. Deeper analysis of UI/UX part should be done. Later, a prototype of the mobile client might be created.
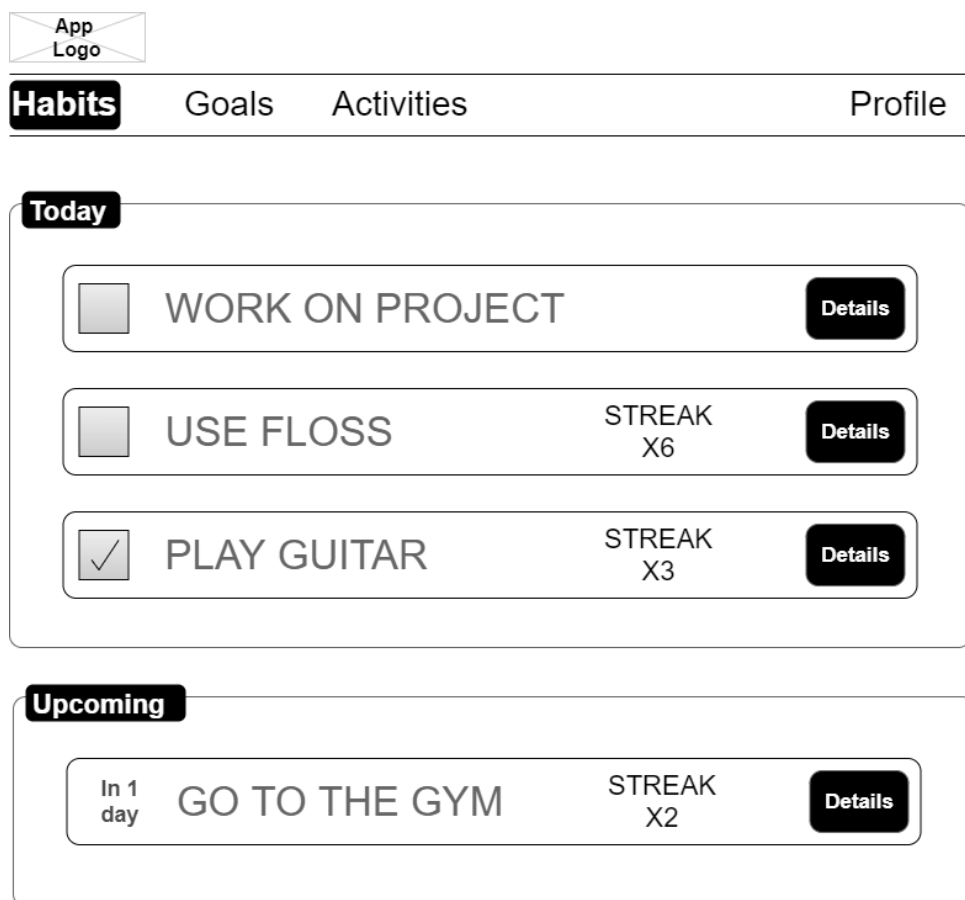
# Appendix A

# Application wireframe and screens



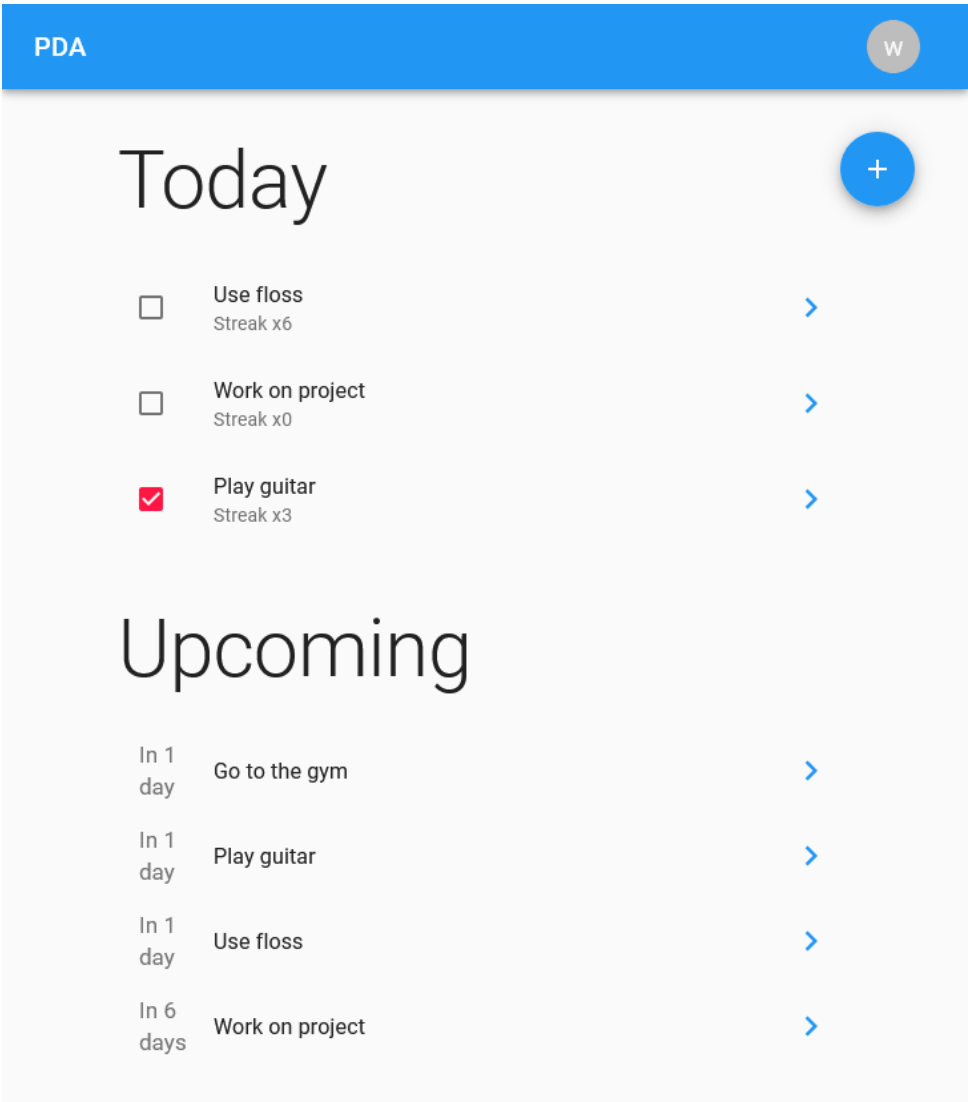**Figure A.1:** Habits page wireframe [author]
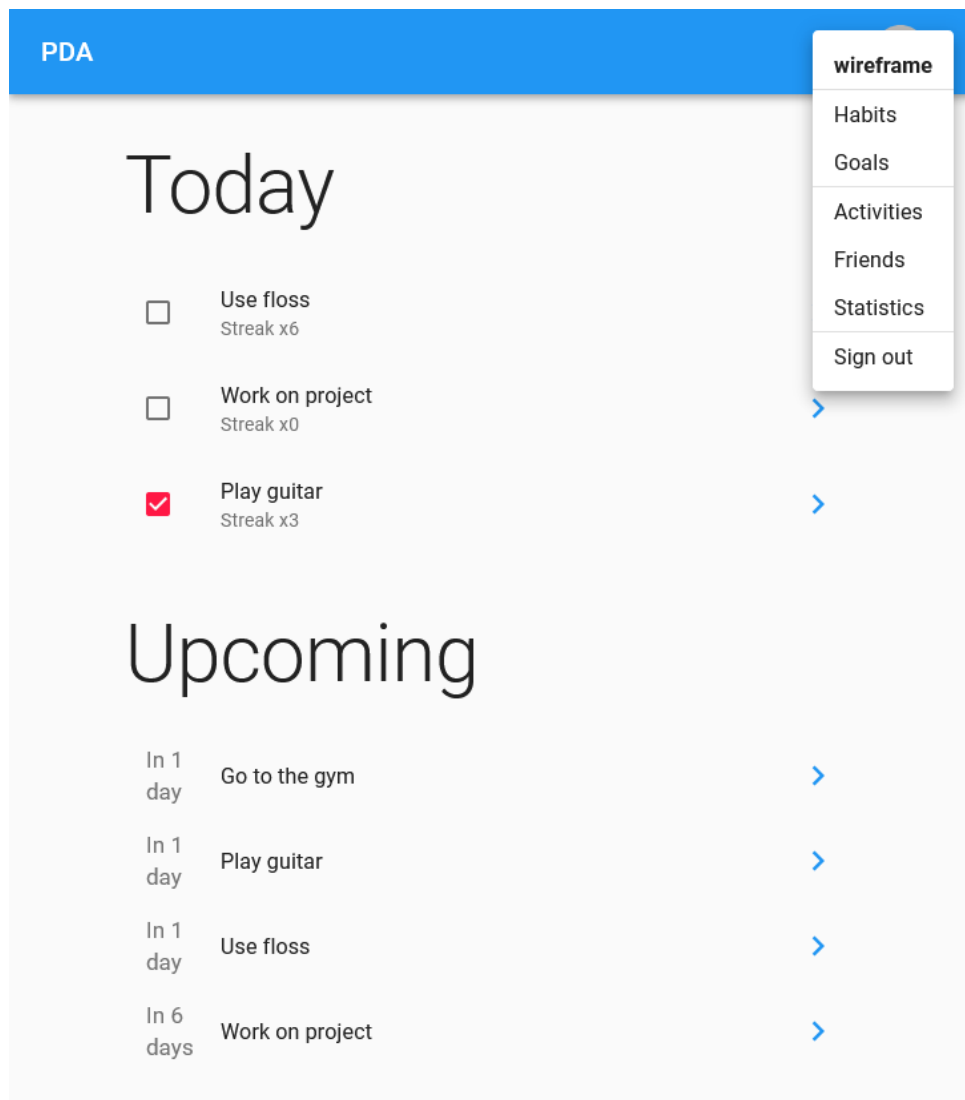
**Figure A.2:** Habits page [author]

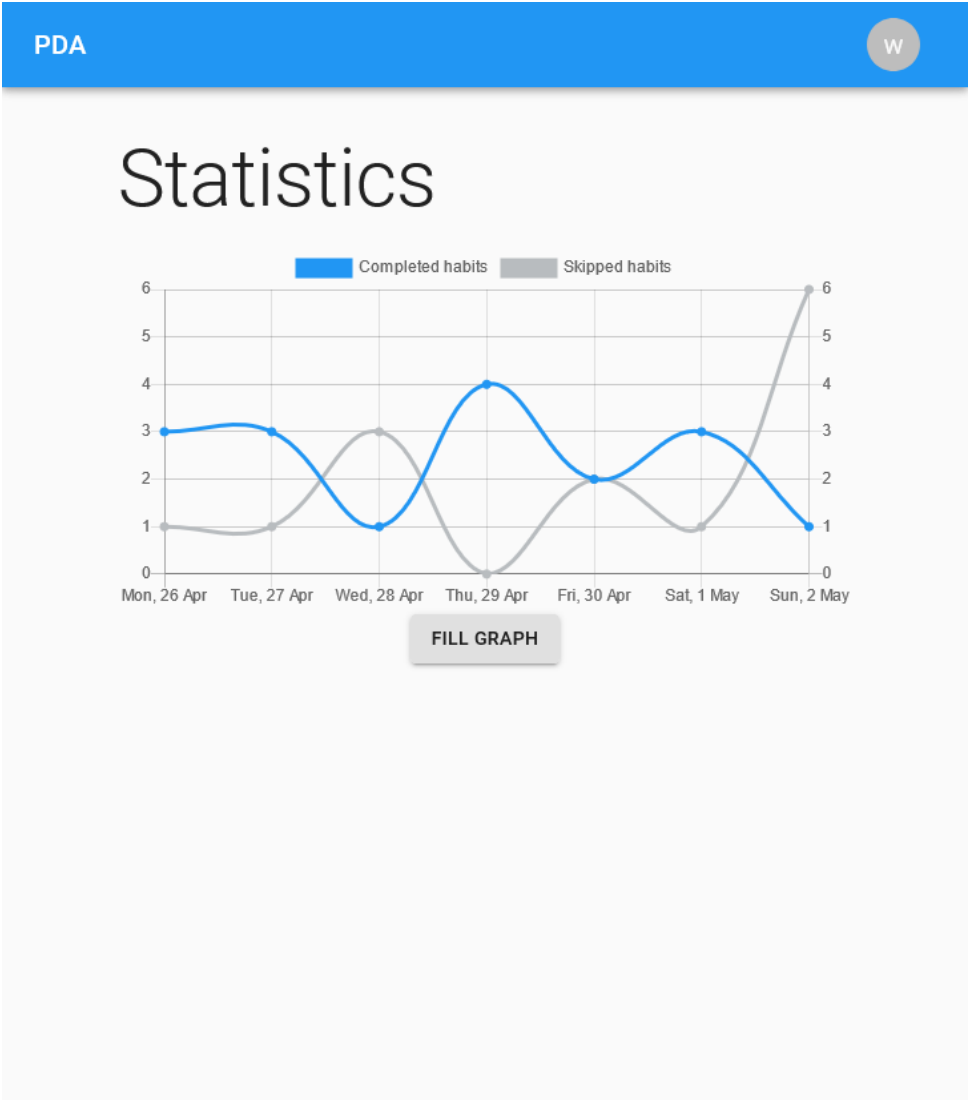**Figure A.3:** Habits page with menu opened [author]

**Figure A.4:** Statistics page [author]

# Appendix B

## Bibliography

[1] A. Maslow. *"A theory of human motivation."* Psychological Review, 1943.

[2] *"Maslow's hierarchy of needs."* Wikipedia.org, 2020.

[3] *What is Personal Development?* skillsyouneed.com, 2020.

[4] Orosz G, Tóth-Király I, Büki N, Ivaskevics K, Bőthe B and Fülöp M *"The Four Faces of Competition: The Development of the Multidimensional Competitive Orientation Inventory."* Front.Psychol., 2018.

[5] Ryckman, R. M., Libby, C. R., van den Borne, B., Gold, J. A., and Lindner, M. A. *"Construction of a Hypercompetitive Attitude Scale."* Journal of Personality Assessment, Vol.62, 1994.

[6] Ryckman, R. M., Hammer, M., Kaczor, L. M., and Gold, J. A. *"Construction of a personal development competitive attitude scale."* Journal of Personality Assessment, Vol.66, 1996.

[7] Ryckman, R. M., Thornton, B., and Gold, J. A. *"Assessing Competition Avoidance as a Basic Personality Dimension."* The Journal of Psychology, Vol.143, 2009.

[8] Ryckman, R. M., Thornton, B., and Gold, J. A. *"Search Engine Market Share Worldwide 2020."* statcounter.com, 2021.

[9] BroadbandSearch.net, *"Mobile Vs. Desktop Internet Usage (Latest 2020 Data)."* www.broadbandsearch.net, 2020.

[10] Uloo GmbH *"Uloo logo."* Google Play Market, 2020.

[11] HabitRPG, Inc. *"Habitica logo."* Google Play Market, 2020.

[12] Unstatic Ltd Co *"Habitify logo."* Google Play Market, 2020.

[13] Appest Inc. *"TickTick logo."* Google Play Market, 2020.

[14] Luke Bickston *"HabitShare logo."* Google Play Market, 2020.

[15] *"Model–view–controller."* Wikipedia.org, 2021.

[16]  *"Spring framework."* spring.io, 2021.

[17]  *"Django framework."* djangoproject.com, 2021.

[18]  *"Ruby On Rails framework."* rubyonrails.org, 2021.

[19]  *"Single-responsibility principle."* Wikipedia.org, 2021.

[20]  *"Kotlin programming language."* kotlinlang.org, 2021.

[21]  *"PostgreSQL Relational Database."* postgresql.org, 2021.

[22]  *"Kotlin vs Java: Performance Drill down & Which to choose."* medium.com/@johnkorly, 2019.

[23]  *"React JavaScript library."* reactjs.org, 2021.

[24]  *"Material-UI React library."* material-ui.com, 2021.

[25]  *"Axios JavaScript library."* github.com/axios/axios, 2021.

[26]  *"Redux React library."* redux.js.org, 2021.

[27]  *"React Native."* redux.js.org, 2021.

[28]  *"TypeScript programming language."* typescriptlang.org, 2021.

[29]  *"Heroku cloud platform."* heroku.com, 2021.

[30]  *"Heroku vs. AWS – Which PaaS Hosting to Choose?"* railsware.com, 2018.

[31]  *"Spring framework documentation"* spring.io, 2021.

[32]  *"Baeldung - Spring tutorials"* baeldung.com, 2021.