Daily Discipline - Project Report

By Ludvig Krantzén and Filip Karlsson

Introduction

Many people know that having regular routines is healthy and beneficial. But creating and maintaining routines can be challenging. Whether that be small, simple routines such as “making your bed” or whether it be more difficult tasks such as “going to the gym”. The reason it is difficult might be because the task is easy to forget or that the task is difficult to make a habit out of. Either way Daily Discipline is here to help.

Daily Discipline seeks to aid in the everyday life of its users. Whatever routines one seeks to implement in one's life, Daily Discipline will help to keep track of what has been achieved every day, week and month. With Daily Discipline a user can always check on their daily, weekly and monthly tasks and upon completion the user can mark the task as completed in the app.

Daily Discipline displays tasks immediately when opening the app so it is simple to access daily tasks. This is to make it easy for the user to complete simpler daily tasks. To motivate the user to complete the more challenging tasks the user will level up whenever they complete enough tasks. Daily Discipline has the ability to add custom tasks, preset tasks and random tasks.

Graphical User Interface

Since the app was created by students who are not designers nor designers in training it is no wonder that the design phase was a struggle. Quite some time was spent in figma trying to get the app to look decent. The design was changed multiple times during the project. Design choices such as keeping it as minimalistic and using as few colors as possible were made to make it easier to make it look good. The app being more minimalistic also aids in making the app easier to understand for the user.

Figure 1-7 shows what the app looks like and its different views.

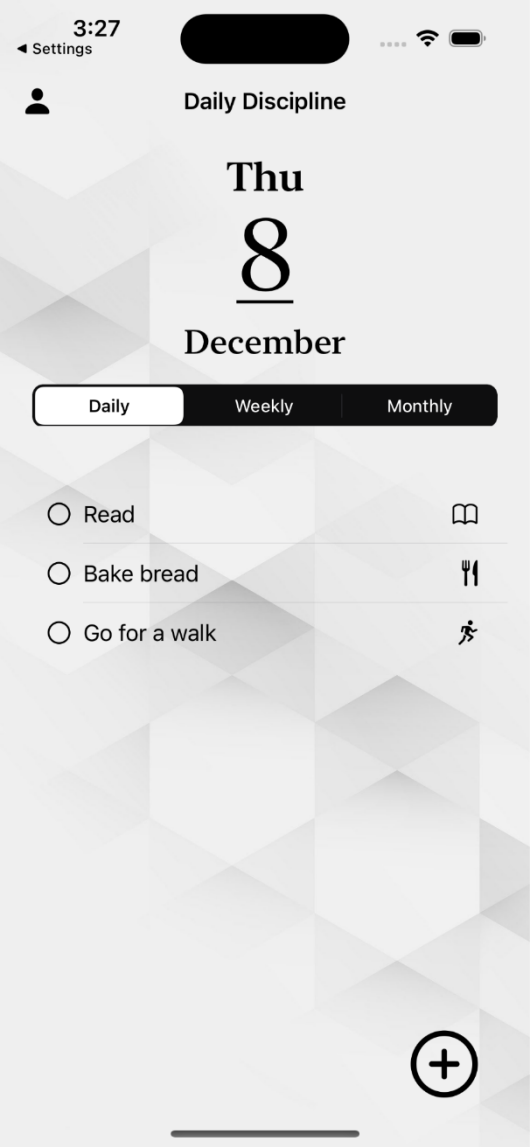

Figure 1, Task List

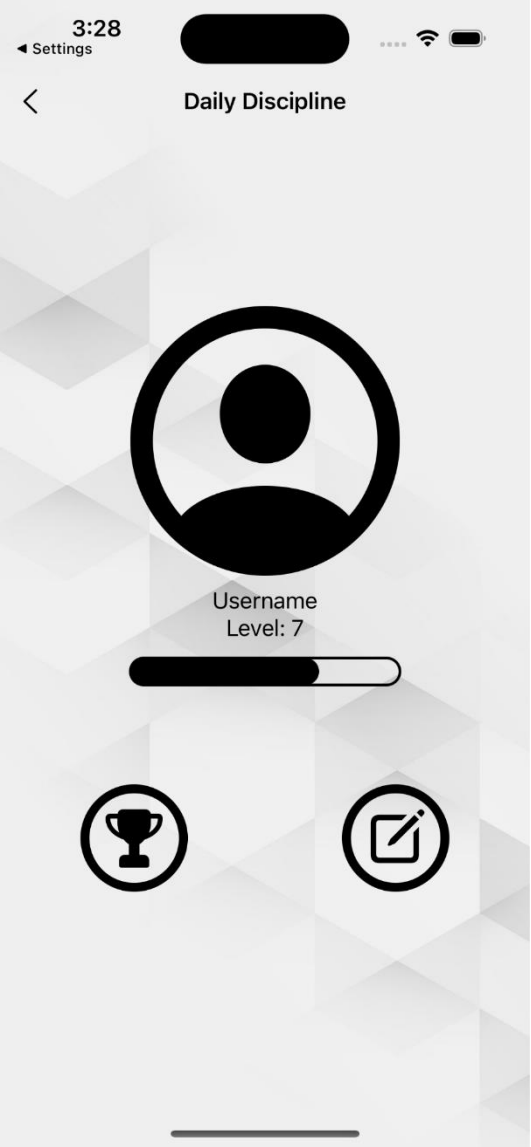

Figure 2, Profile View

Figure 3, New task view, Custom task

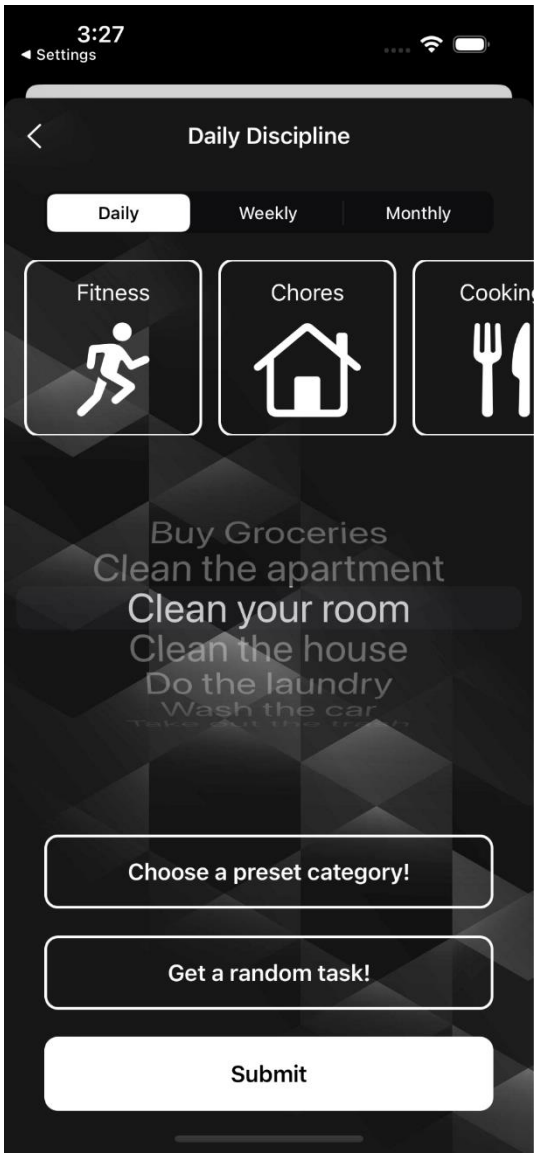

Figure 4, New task view, Preset task

Figure 5, New task view, Random task

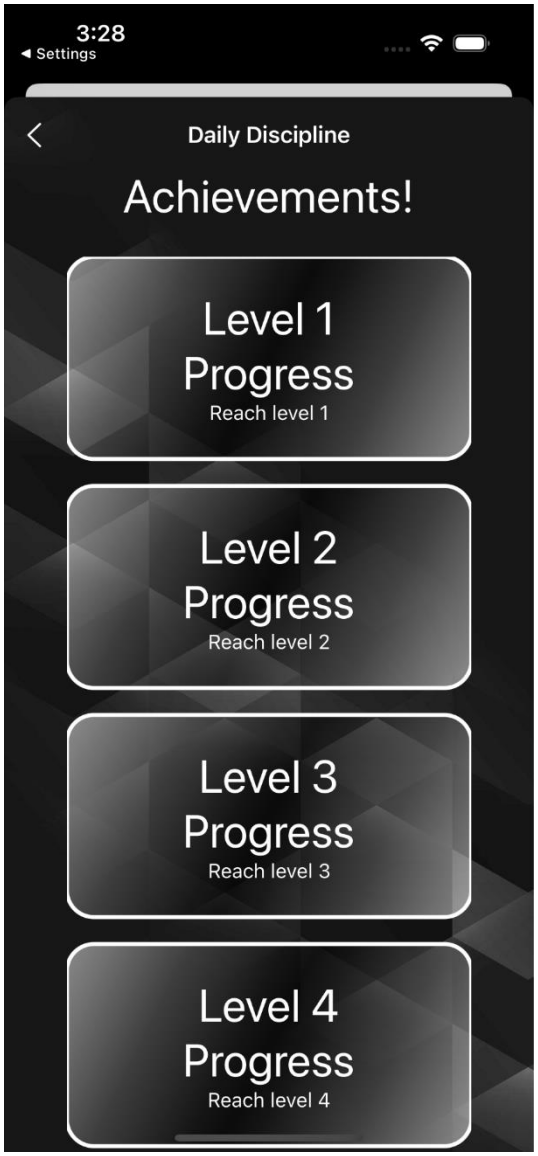

Figure 6, Achievements view

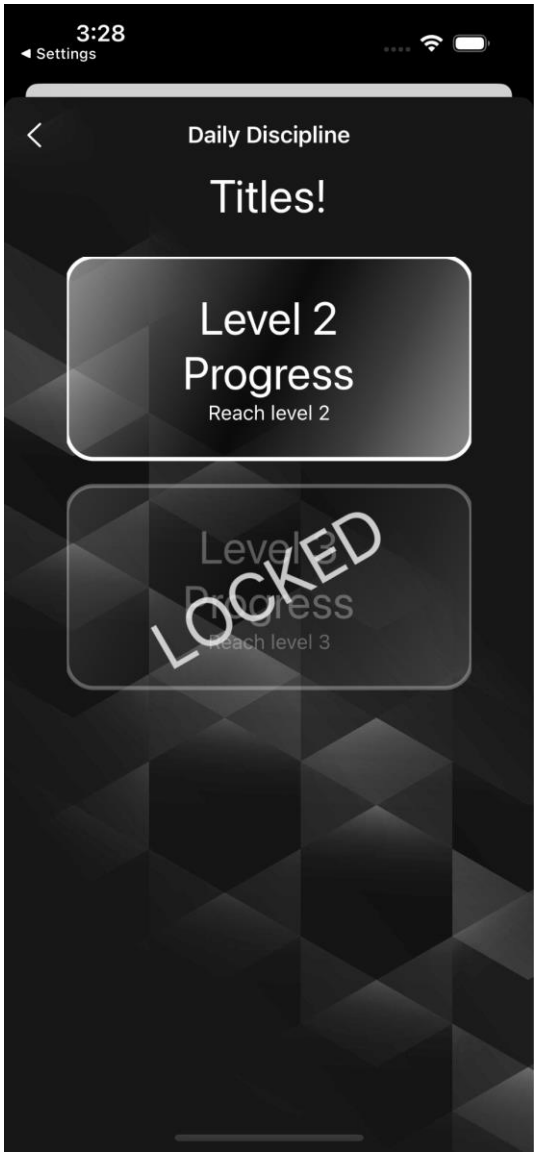

Figure 7, Titles view

Architecture

Originally the code structure was supposed to follow the MVVM pattern. However due to lack of time and experience regarding design patterns it ended up not being as obvious which design pattern was used. The project was divided into Views as much as possible, separated some logic into Models and used ViewModels when it was deemed necessary. This is one of those areas that could have been improved if there was some more time left when the rest of the project was done.

The placement of files follows a certain principle. Firstly, the Views, Models, and API are placed in their separate folders. ViewModels have been placed in the Models folder. In the Views folder, the main views are placed separate to each other, and all have the relevant sub views in a folder called “MainViewNameSubViews”. Whenever a sub view is used on more than one main view it is placed in a folder called “GlobalViews”. This is all to make the different views and sub views easy to navigate.

Implementation

As the main focus of the Daily Discipline app was the tasks, this was considered the most fundamental part and the foundation for the app. The decision was made to brainstorm and research the best way to build this foundation. As well as building different solutions to see what suited the app the best. After some time, several different foundations had been built and soon after one foundation was considered appropriate for the project.

With the increasing size of the project a general principle was introduced. It was to create functions, UI, etcetera in isolation of the main project branch. The purpose of this being mainly two things. Firstly, creating the functions themselves was easier to do on smaller, more easily navigable projects. Secondly, the function was more safely implemented into the project since it was first created then recreated and integrated.

The Daily Discipline app is built around CoreData. It relates to almost everything in the entire app since it stores all the tasks and the user's level and experience. Just as it is very difficult to add walls to a house that does not have the foundation built yet it is also very difficult to add anything that relates to CoreData when CoreData is not yet fully implemented. It was very difficult getting it to work as intended and it had to be redone multiple times during the project. Majority of the time spent on the project was spent implementing CoreData which left little time for other things. It now works as intended. It was solved through just spending a lot of time on it and simplifying it, removing the previous CoreData relationships that were implemented before.

The plans were structured in a timeline with notes that kept track of what should be done next and what had already been done. All logic but mainly CoreData took longer than expected which delayed everything else which made the plan timeline to become inaccurate and hard to follow. The timeline was structured into 3 weeks with every week being a “major” goal while the notes were smaller goals and all the things that had to be done. The app was meant to be completed at the beginning of the last week but ended up being done the same day as the deadline. For the deadline to be met some aspects of the project needed to be scrapped.

Learnings

Early in the project there were attempts to implement the UI in full. This was later considered a mistake, mainly for two reasons. Firstly, the app's look and functionalities had yet not been completely realized. A lot of the UI was later changed to make the app more aesthetic and to better reflect the app's functionalities. This resulted in a lot of time spent on something that later was going to be scrapped in favor of the better looking, more intuitive UI. Secondly, it became difficult trying to implement functions and logic into a UI that was already designed. This made the process of creating functions and logic more time consuming than they needed to be. It also made it more complicated since the design was remade multiple times during the project, leading to a lot of time wasted.

Outlook

As stated previously. The architecture could use some work and make it clearly follow a design pattern. Even though the developers might understand the code perfectly, other people might struggle if it does not follow a common way to structure the code.

All hard coded data is stored in a file called StoredData. This includes all categories that a task can have, all achievements available, all preset tasks, etc. This is a file that could be expanded, adding even more achievements, preset tasks and all other things that the file stores. It could also be effective to add an API that stores all the tasks so we do not have to add everything manually.

Daily Discipline could use some more content in order to motivate users to continue using the app. Such as a more complex leveling system with extra EXP for streaks and tasks having different difficulties and EXP that reflect the difficulty. Another example of more content such as this is more achievements relating to categories or certain tasks as well as fun and interesting titles unlocked in conjunction with the unlocked achievements.

If the UI had more animations and some minor adjustments here and there this could also majorly improve the user experience.