Challenge Tracker Proposal

By Ye Ri Park

Contents

I. Motivation to Join the Mentorship Program

As a student who hopes to be an inspiring software engineer, I always try to practice software development and teach myself new tools that real software developers use. However, my questions only increase as I study more. Also, I believe that hands-on experience is very important for software engineers. However, when I am working on my project alone, I am always unsure if my approach is how real software engineers would do, and I want to know how my approach could be improved. This is especially why the mentorship program from Credit Suisse looks appealing to me. Through the mentorship program, I will have the opportunity to talk to real software engineers and discuss my programming project. I always strive to further hone my coding skills and learn new approaches to solving technical problems. Receiving both technical and non-technical mentorship would be a precious experience. I am eager to be part of the INSPIRE Women Program to embark on a career in technology with Credit Suisse and become an influential software engineer one day.

II. Proposed Approach – Challenge Tracker

We often set goals and find ourselves losing track of progress. This application, Challenge Tracker, helps us track goals, remind us when we are slacking off, and celebrate when we meet all the goals. It also has a collaboration feature with your friends where you can see rankings based on the completion of today's goals and where you can form challenge groups with your friends to reach 100% every day.

a. Use Case Diagram

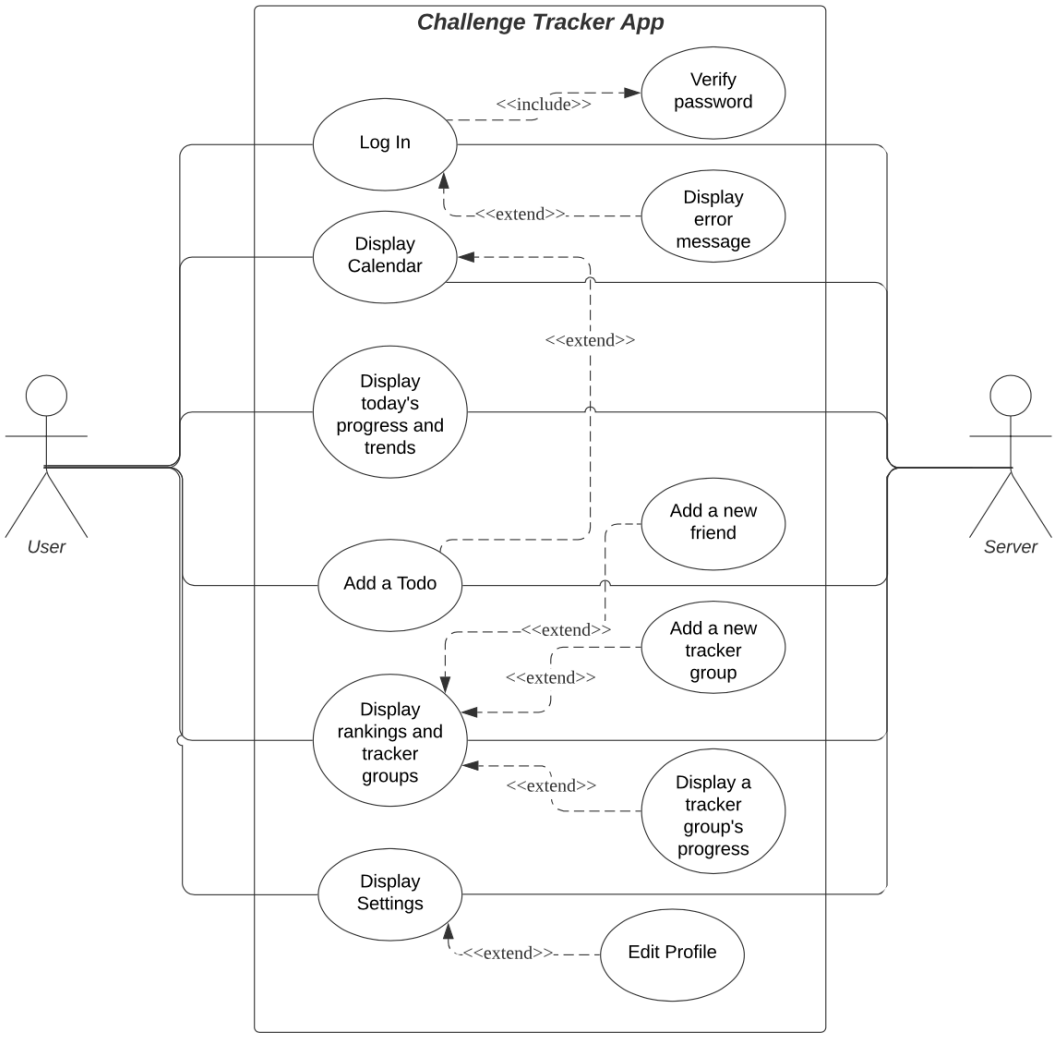

Figure 1. Use Case Diagram

The application consists of **six main activities** for users.

1) Every user should log in to view his/her calendar.
2) Users can view their calendar which has their to-do activities recorded.
3) Users can view their today's progress in percentages and their completion trends in a bar graph.
4) Users can add their to-do to their calendar.
5) Users can view today's rankings among their friends and themselves and view how their tracker groups are doing altogether.
6) Users can view their settings and edit if necessary.

b. Database Diagram

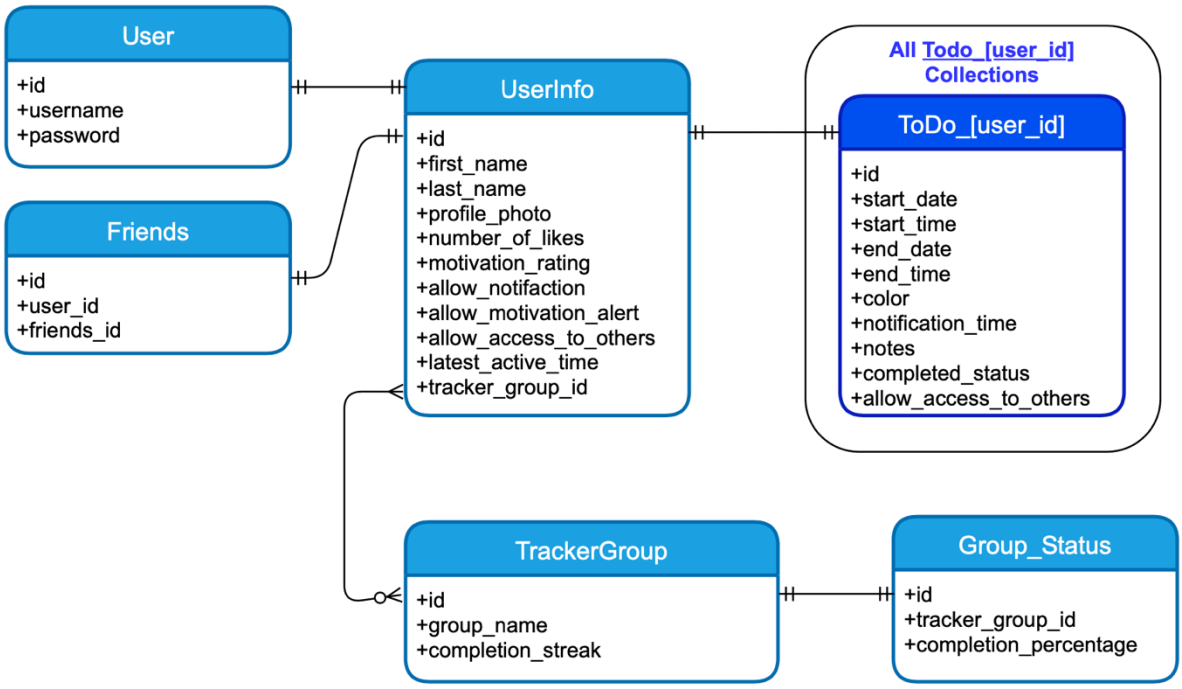

Figure 2. NoSQL Database Diagram

A NoSQL database will be used for the application since it supports multiple data structures and is more scalable than relational databases. Figure 2 shows the overview of the design of the NoSQL diagram for the Challenge Tracker application. Although this is not the final design, the current diagram consists of **six big collections**.

1) User – This collection is for storing user's username and password.
2) UserInfo – This collection consists of information that can be modified in the settings page.
3) ToDo_[User_ID] – There will be a collection for each user to keep track of his/her to-do list.
4) Friends – This collection is to keep track of friends for each user for displaying the rankings and allowing users to form tracker groups.
5) TrackerGroup – This collection consists of the basic group information that is supposed to appear on the third page of the application.
6) GroupStatus – This collection consists of data that appears when a user clicks on his/her tracker group to view how other members are doing.

c. App Features

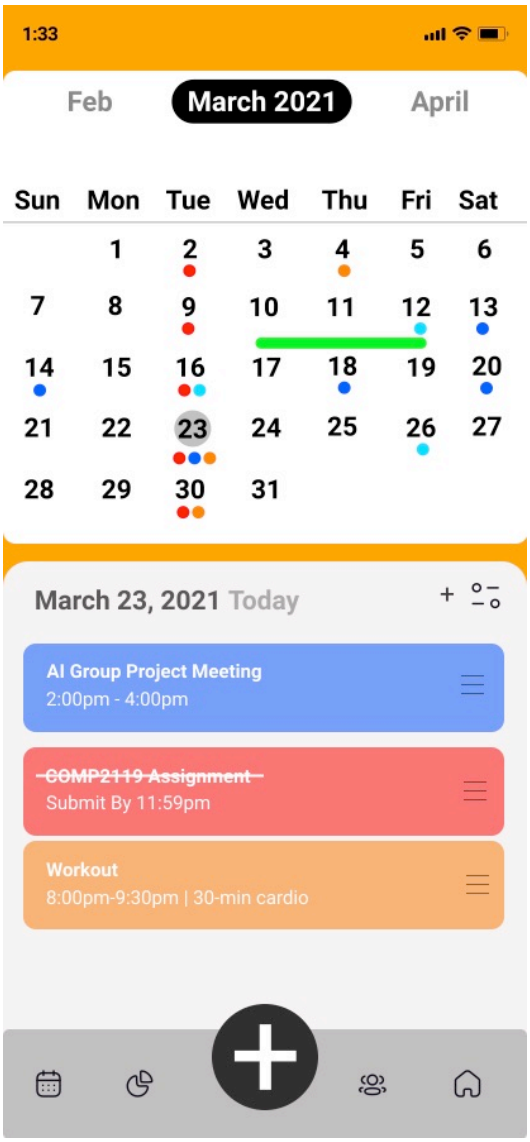

Figure 3. First Page of the Application

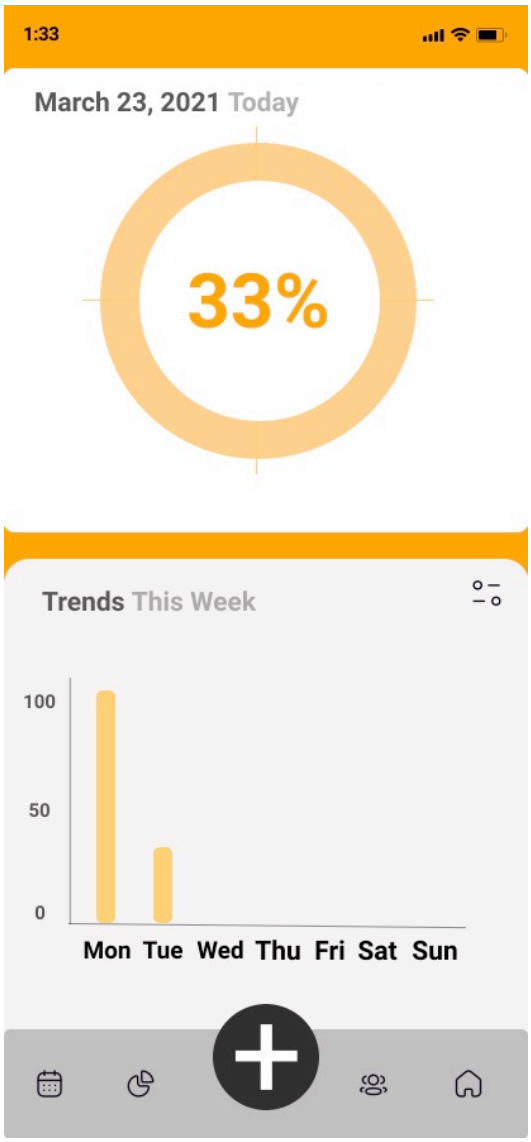

Figure 4. Second Page of the Application

Figure 3 shows the calendar page of the application. When the app opens, this page is the default page with today's to-do activities listed below. Users can change the priorities with the three horizontal lines on the right of each to-do block. The filters include by date added and by time. Users can also add new to-do with the plus button on the top right of the bottom block. On the calendar, the colors are set by the users where each color could represent a different type of goals.

Figure 4 shows the second page of the application. The circular graph shows how close a user is to complete today's goals. The percentage represents a user's progress. The second bar graph shows the trends of the completion percentages. Users can filter to view the trends daily, weekly, or monthly.

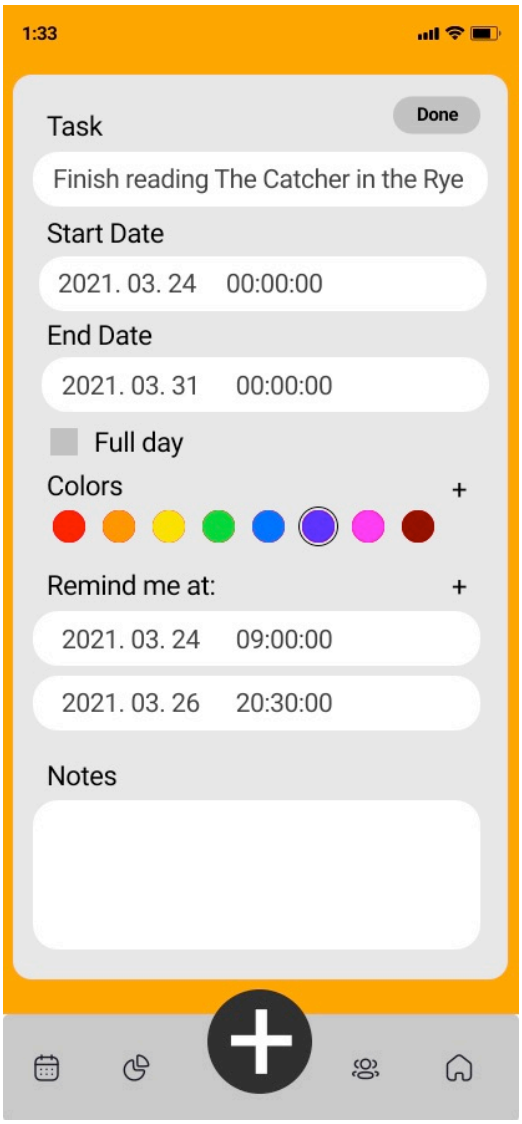

Figure 5. Adding To-Do Page of the Application

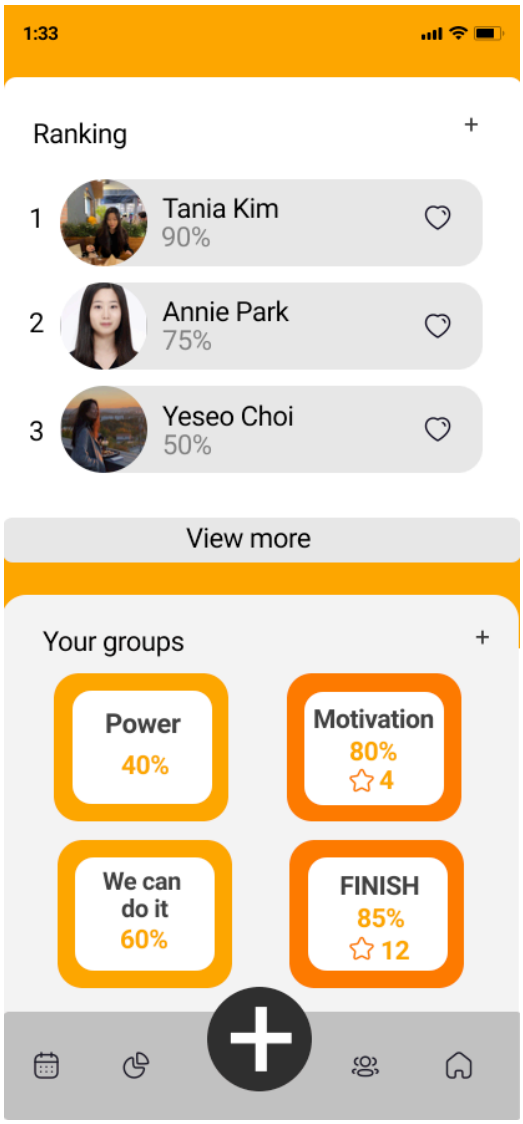

Figure 6. Third Page of the Application

Figure 5 is the page that appears when a user tries to add a to-do. Users could either press the plus button from the first page or press the big plus of the bottom panel. As shown in Figure 5, users can include their tasks, start dates, end dates, and colors. Users can also add multiple

reminder times and notes for each task. When completed, users will have to press the Done button, and the tasks will be added to the calendar.

Figure 6 shows the rankings and tracker groups for each user. The rankings are to hope users to be motivated by seeing how other friends are doing today. Users can also like their friends' rankings as a way of celebrating their progress. This page also shows the tracker groups where each tracker group can reach 100% when all members complete their goals. This is a way to make users feel responsible for their set goals. The number next to the star shows the streak, which is the number of consecutive days that the group has reached 100% together. Users can add friends and create new tracker groups on this page.

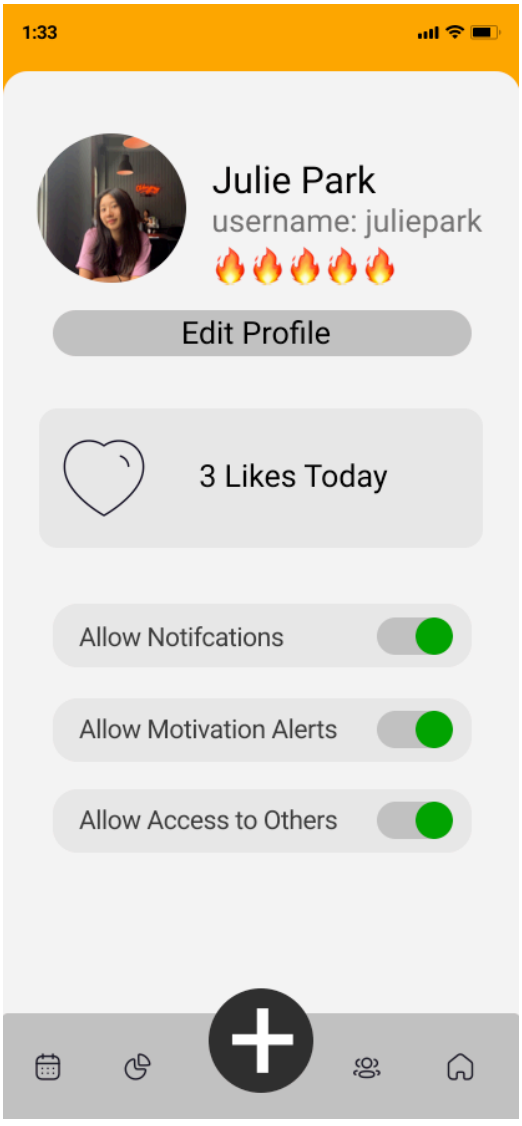

Figure 7. Fourth Page of the Application

Figure 7 shows the settings page. Users can edit profiles to change their profile photos and the fire ratings which is to represent how motivated they are. The maximum number of fires each user can have is five. This page also shows the number of likes the user got today in a way to encourage users to work hard and keep up their work. Users can also change some settings on this page. The app sends notifications based on the reminder times that users set for each task. The app also sends notifications as a way to motivate users regardless of the reminder times they set. The app allows users' friends to view their to-do lists and trends. These three could be turned off depending on how users turn these on or off.

III. Possible Technical Solutions

	Possible Technical Problems	Possible Technical Solutions
1.	Slow response rate for particular activities in the application	The database may need to be modified to be queried more efficiently for each activity. I will have to reconsider how to reduce the queries to improve the response rate.
2.	Inaccurate information displayed on the application	I should build a system to make sure all activities by users are updated on the server, so the database consists of correct and updated information.
3.	Data security	It is important to secure the firewall, but keeping all data encrypted could also be a way to ensure cyber-security. It is also important to back up my data regularly so that no data is lost.

- Poorly written code could cause memory leaks, synchronization issues, and many more. When writing code for both frontend and backend, it is important to review code and figure out solutions to all errors.
- To avoid technical problems and to be able to solve every error I encounter, it is crucial to keep researching and understand every step I am working on. I should not merely focus on completing the project but rather think about how I could make a more efficient application.
- I should also keep track of what I have done and be able to explain every choice I make when designing an application.

IV. Potential Tech Stacks

	Potential Choice of Tech Stacks	Reasons
1.	(Frontend Development) React Native and JavaScript	React Native is a JavaScript framework for cross-platform apps. I want to develop a cross-platform app,

		so the app can be adapted to the majority of devices. Both Android and iOS users could download the app. However, Compared to the native app development, it has relatively poorer UX, lack of flexibility, and performance delays.
2.	(Frontend Development) Swift, XCode, and iOS SDK	Swift is a programming language created by Apple for building apps. Swift contains dynamic libraries and supports for manipulation of text strings and data, which reduce the size of the app and increase app performance. Many iOS developers use Swift because Swift is more functional than Objective-C, it is less error-prone. Swift is also known for its higher speed and lesser memory consumption. The iOS SDK has an API that helps linking software applications and the platform they run on. As an iOS user, I will have a better understanding of how the features and functionalities are implemented on iPhones. The disadvantage of using Swift would be that only iOS users would be able to use the app.
3.	(Backend as a Service) Firebase	A Backend as a Service is a platform that controls backend side development and cloud infrastructure. It includes features like scalable databases, APIs, file storage, and push notifications. Firebase is a BaaS backed by Google. Firestore is a cloud-hosted, real-time and offline-first document database Cloud functions also allow users to set up their own server. Cloud messaging provides an infrastructure to send and receive push notifications between the server and devices at no cost. Other important features include hosting and cloud storage.

		The biggest advantages of using Firebase are productivity gains and outsourcing cloud management responsibilities.
4.	(Database) NoSQL - Firebase	The advantage of NoSQL over SQL is that NoSQL does not have a fixed schema. NoSQL requires much less structure than SQL as each data is self-contained and independent. Having more flexibility within the database would be more efficient since it is easier to make changes to my database. Also, being able to scale horizontally is a huge advantage because I would not have to migrate data to a larger server like most SQL databases.