Project report

Process report of Agile calendar Application

Sumeia Jemal Said
Martin Larsson
Abo Khalaf, Muhamed Kheer
31 March 2021

Table of contents

About us

We are three students in this project. Two of us studying Software Engineering and third one studying Web programming. We are second year students.

About project

In this project we were asked to develop a mobile application for LSS-boende. LSS-boende are having issues with adapt the time to the customer's needs because now days the the planning for the week is adapted after the staff time. Which VOF “Vård- och omsorgsförvaltningen “ wants to change this and allow the customer to be more involved in the planning of their efforts.

Scenario

Harald lives in an LSS home. He has a number of activities (cleaning, showering, shopping for food) that are planned during the week.

The problems that can arise are many for example:

- Sometimes Harald is not in the mood when it's time for cleaning
- If someone falls ill, it will be an hourly substitute that Harald does not know who will help him with the laundry, if Harald does not choose to skip the effort and then has to wait a week or so until the next opportunity
- If Harald's plans change, the schedule will be difficult to adapt

Expected result

A better opportunity for the user to be involved and plan their own week in a digital solution. The digital solution should have a transparency and connection for the three different users to get the information they need.

Mobile development tool

React native

We used react native as mobile development tool. Since react native is an open-source mobile application framework. React Native has a number of open-source libraries of pre-built components which can help you further speed up the development process. React Native is a JavaScript framework for writing real, natively rendering mobile applications for iOS and Android. It's based on React, Facebook's JavaScript library for building user interfaces, but instead of targeting the browser, it targets mobile platforms.

React native navigation

React Native Navigation is a module, dependent on and intended to be used alongside React Native. React Native Navigation provides 100% native platform navigation on both iOS and Android for React Native apps. The JavaScript API is simple and cross-platform.

UI Kitten

UI Kitten is a customisable React Native UI Library based on Eva Design System specifications, with 30+ UI components, 2 visual themes, and other supporting modules.

Expo

We used Expo react native and Expo is a toolchain built around React Native to help you quickly start an app. It provides a set of tools that simplify the development and testing of React Native app and arms you with the components of users interface and services that are usually available in third-party native React Native components. With Expo, you can build and deploy React Native apps for both iOS and Android with ease.

Firebase

We used Firebase for database and Firebase is Google's mobile application development platform that helps you build, improve, and grow your app. Firebase is a Backend-as-a-Service (Baas). The Firebase Realtime Database is a cloud-hosted NoSQL database that lets you store and sync data between your users in realtime.

Cloud Firestore

We chose to work with Cloud Firestore since Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud. Firebase Realtime Database, it keeps your data in sync across client apps through realtime listeners and offers offline support for mobile and web so you can build responsive apps that work regardless of network latency or Internet connectivity.

Firebase Authentication

We used Firebase Authentication for secure login to the user. Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more.

Platforms such us

Clubhouse

We used Clubhouse for planning out our sprints. Clubhouse is a project management tool that's specifically designed for software development teams. With a simple Kanban-style interface, it offers a clutter-free experience for better team collaboration.

And that's pretty much what Clubhouse's philosophy is about — provide a platform for software teams to work together and create products. It helps you:

- Plan project backlogs and sprints
- Automate and manage workflows
- Visualise project progress
- Streamline projects across software development platforms

Toggl track

We used Toggl to track our time budget. Toggl Track is a time tracking software operated by Toggl OÜ, headquartered in Tallinn, Estonia, that offers online time tracking and reporting services through their website along with mobile and desktop applications.

Git

We used Git to keep track on changing's for our code. Git is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development.

GitHub

We used GitHub to be able to work in team. Github is a highly used software that is typically used for version control. It is helpful when more

than just one person is working on a project. Say for example, a software developer team wants to build a website and everyone has to update their codes simultaneously while working on the project. In this case, Github helps them to build a centralised repository where everyone can upload, edit, and manage the code files.

Discord

We used Discord to communicate with team members. Discord is a VoIP, instant messaging and digital distribution platform designed for creating communities. Users communicate with voice calls, video calls, text messaging, media and files in private chats or as part of communities called “servers".

Microsoft teams

We used Microsoft Teams to communicate with customer. Microsoft Teams is a proprietary business communication platform developed by Microsoft, as part of the Microsoft 365 family of products. Teams primarily competes with the similar service Slack, offering workspace chat and videoconferencing, file storage, and application integration.

Google docs

We used Google Docs to report our project. Google Docs is an American word processor included as part of the free, web-based Google Docs Editors suite offered by Google which also includes Google Sheets, Google Slides, Google Drawings, Google Forms, Google Sites, and Google Keep.

Google meet

We used Google Meet to have customer meetings. Google meet is a video-communication service developed by Google. It is one of two apps that constitute the replacement for Google Hangouts, the other being Google Chat.

Process

We started with learning what is React native since we chose to work with it. The main reason why we chose react native is because the customers wants a iOS mobile application and we are just two people of four that have Mac operating system. Since developing an iOS mobile application will require to have a Mac operating system. So we decided to work with react native since in react native we can develop the app for both iOS and Android so we all can participate on coding part. The reason we chose to develop a mobile application and not web application is that the customers wanted to have a notification extension or more like a warning notice for upcoming event on mobile phone. And from a previous experience we have noticed that it won't be possible to have the notification on web application in the way the customer wished. So after discussing and searching we have come to a conclusion of having a mobile application will be more appropriate to customer wishing. Since adding a notification to mobile application is more appropriate and easier to implement than adding to web application. Since one of our group has experience with react native so we started directly with coding and with setting up react native project. We had whole sprint for setting up the development environment then we had another sprint for learning how to work with react native. We started with creating accounts in different platforms. Such as toggle, clubhouse.io, GitHub. To help us planning sprints and tracking our budget and manage our code files. We started with creating a simple login screen so we can move on with the other features. And then on learning process we stared creating other screens. And connecting to Firestore firebase database. And then we started focusing on customer feedback. Which we had some late requirements that a took alot of time than expected. The most thing that we focused on was the home screen and how to create "insats". Insats is a daily action that's the user needs.

The customer had some detailed requirements about how should the user create an “insats” or plan for the week which “insats” should take action in which day. And being able to move the “insats” freely also took a lot of time. We started lately with the styling. Which we recently got from customer information about how should screens look. And what colors the “insats” should have and what should they include .

Problems

First problem we faced in this project was setting up the react native in iOS operating system because two of our group members have iOS os.

Second problem we faced was one of our group members left the project.

Third problem we faced was we didn’t get any API from customer that we have been expected to get as soon as possible.

Fourth problem we faced was dropdown menu connecting to firebase database. We got this requirement after discussing the features with customer.

Fifth problem we faced was firebase authentication. Because expo does not support some firebase modules. Which took time to found out that.

Sixth problem we faced was drag and drop extension. It was also late requirement that we got. The problem was using a complete module.

Seventh problem we faced was react native component pan responder.

Eighth problem we faced was creating seven days of week and specially Swedish days.

Ninth problem we faced was trying to fit insats to specific time.

Tenth problem we faced was that array which saves all the positions and keys etc has created duplicates of those that are changed.

Eleventh problem we faced was screen size problem.

Twelfth problem we faced was the setting timer issue. Located in "App.js" it is there to not show the warnings for firestore timers that are running all the time. This warning: Setting a timer for a long period of time, i.e. multiple minutes, is a performance and correctness issue on Android as it keeps the timer module awake, and timers can only be called when the app is in the foreground.

Solutions

First problem, we found out that the problem was iOS does not support some react naive latest version. Which it took us hours and hours to figure out and to try to solve the problem or to come up with another idea.

Second problem, we solved by contacting the customer so they can lower down the requirements or to be more specific which features are most priorities.

Third problem, we solved by using firebase firestore so we can at least start to backup our datas for application.

Fourth problem, we solved by try out many different ways of connecting react native to firebase database.

Fifth problem, we solved by hardcoding the extensions that have been imported from module. Unfortunately the the errors are so vague. Troubleshooting the problems was not easy at all. It was also a problem that we didn't start with the secure login and we developed our project til a certain point.

Sixth problem, we solved by hardcoding the pan responder for the drag and drop instead of using the complete module which it didn't not support some extensions.

Seventh problem, we solved by changing our code to classes instead of functions. Since UI kitten module use functions in their documentation so we had to remove that module because we decided to continue with classes.

And also alot of example codes or YouTube tutorials use classes which many developers highly recommend that.

Eighth problem, We solved by adding moment module to be able to have Swedish days we had to import the “moment-with-locales-es6" but then it was better hardcoding the Swedish days. So we removed the module.

Ninth problem, We solved by finding out specific block size for adding the insats.

Tenth problem, We solved by calling a function from the input components that run in home screen (where the array is updated), to then be able to run splice in the 2 places in the array where the duplicates are located.

Eleventh problem, we solved by getting the exact device that the customer will use for this project.

Twelfth problem, we solved by ignoring setting timer which was guide we found from may developers.

Progress

We decided to continue working with expo react native because running an application with expo is much faster than the react native only. Plus with expo we could run the app from different devices.

We started planning features in sprints and estimate the time so we know at the end which features we will be able to deliver.

We connected our application to firestore firebase. So that we have a backup of when the user create, delete and edit an “insats”.

Discussing the issues with team members helps alot on progressing the project.

We decided to research drag and drop separately so we can have many solutions. We had two solutions. One to work with complete module and just add some extensions to it and other solutions was to work with help pan

responder and hardcode everything. We decided to test separately then decided to continue with better solution. We faced alot of problems working with complete module and it took hours to try to fit in the module with our application. After alot of researching we figured it out the problem which the module it self does not support some extensions. Which we needed the extensions for our application. So we decided to work with pan responder instead and hardcode everything. So we can add what ever extension we want.

We are not currently handling the case where a user goes back in time and adds insats.

Since customer feedback changes while we are developing the app which menas we had to change the features based on the feedback we get every sprint delivery.

Since we have been working the most on the home screen which was the most evolve screen. In the beginning we had the add screen which was about creating insats but we have replaced that screen with drag and drop extensions. Which we have mixed home screen with drag and drop extension and completely deleted the adding insats screen which was the solution we had after discussing with the customer.

So the adding instas screen was about a screen with calendar and drop down menu time and insats plus adding the name of staff.

Along with adding the drag and drop extension plus calendar with weeks and hours has solved the problem of having another screen to be able to add insats. So now in the home screen the user can drag an insats and drop it in the specific hour and day.

Another extension we added in home screen was swapping. Considering that the customer wished the user should be able to edit or delete insats.

Which in the beginning we had edit and delete screen which the user would be able to change day, time and insats and staff. But with swapping

extension we succeed to edit time and deleting insats. So the user could simply drag the wished existed insats which is swapping and drop it to another time or day or could just be time and day or same time but another day. And we have an empty place in the home screen which we call it trash so that the customer can drag insats and drop it in trash which simply will be deleted. But we managed to turn the deleted insats back to insats moln so that the user would be able to reuse the deleted insats. So by using swapping extension we deleted the edit and delete screen.

From the latest feedback from customer was that when the user click on insats it should popup a window with extra information about the insats. So we created modal extension which modal is sort of view component that allows popup window above enclosing view.

We have been working on the feature which was to prevent impossible bookings with regard to staff resources. We got a schedule from customer about when are the staff available which according to that we should limit out bookings of insats. So, for occasion we have for example if the user book insats on Wednesday at 11 oklock, another user won't be able to book that time. The user will get a popup window about the time in the specific day is not available. Since there aren’t any staff available at that time.

This feature took us a little bit more time because of async and await which we had to be carful with.

We have aslo worked on style as customer wished. They wanted to have all insats in a sort of cloud at the top of the page so the user would drag an insats from there. They wished to have different colors for the weeks, specific colors which we have fixed also.

As the customer wished the app needs a notification in which when the user create an insats, the user should get a notification about the created insats. Which we have notifications right now and they are only being sent to

the user that creates the insats, since it's simpler to check if you receive the notification this way.

We have also added another account for staff only since the customer wished the staff needs to see the insats for the day. Which we have fixed another account for staff only so that when the staff login with specific login info and it will show screen with the insats for the day only.

We have been also working on tests since we knew that there would be another developers which will continue with this project so we had to make sure that our functions are running correctly. We already have manual testing and our application works as it should.

Since we are working with react native expo and the appropriate testing framework would be jest because react native documentation and expo documentation highly recommend jest. Jest is javascript testing framework and it primarily designed for React-based applications. There would be another detailed documentation about how to use jest on our project.

We have started with deploying the application. Some issues we faced were that as developer you need to have an account on google play console to be able to deploy an app on play store. Which we don't have in our team but after contacting the customer we got invited to account that the customer already have. But another issues was that since we were just guests on the account, so we couldn't create an app there. After contacting the customer again, the customer made one of us admin. So now we are in process setting the app before releasing it.

Another problem we faced with deploying the app was the apk version which we needed to upload the apk file with another version. Being carful with the size for the app icon is also an issue with deploying the app. Plus creating a privacy policy url was an issue also.

Now the app is published but google play console are experiencing longer than usual review times so the app hasn't reviewed yet.

Suggestions

This suggestions are just from what we have experienced during developing this agile calendar project.

First suggestion, before using any module read about issues that people had while using the module and research alot to see what does module support and which extensions it does support it.

Second suggestion, it is very important with coding in which operating system. So it is much better to code on same operating system, if not always be aware about the versions while adding a new module. So that finding the errors will be much easier if always keeping eye on versions. Since sometimes the error are so vague which being aware of adding module will help solve massive problems.