Introduction

OVERVIEW

Chat App is a JavaScript mobile app that allows users to communicate with each other. This app uses Google Firestore to store the chat messages and demonstrates React Native development.

PURPOSE and CONTEXT

Chat App is an exercise project to get hands-on experience on React Native development for mobile apps. As more and more people are involved with their phones, we as developers need to be able to adapt and create mobile versions of web apps. This used to be troublesome because you would have to specialize for these different platforms (iOS and Android), but now React Native, among others, have made this incredibly easier to build and maintain.

OBJECTIVE

Chat App is a project that focuses on using React Native to build a chat app for mobile devices. It includes a chat interface and options to share images and their location. Also, messages will be stored using Google Firestore Database.

DURATION

This project took about 3 days to complete.

CREDITS

I am the main developer for this project, but I received feedback and assistance from my tutor and mentor from CareerFoundry.

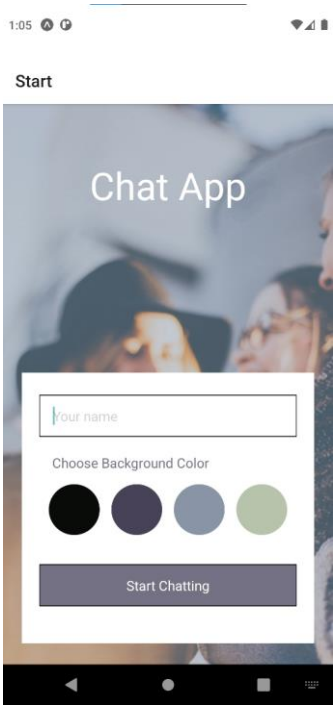
TOOLS, SKILLS, METHODOLOGIES

- React Native
- Expo
- Google Firestore Database/Cloud Storage
- Google Firebase Authentication
- Gifted Chat library

Approach

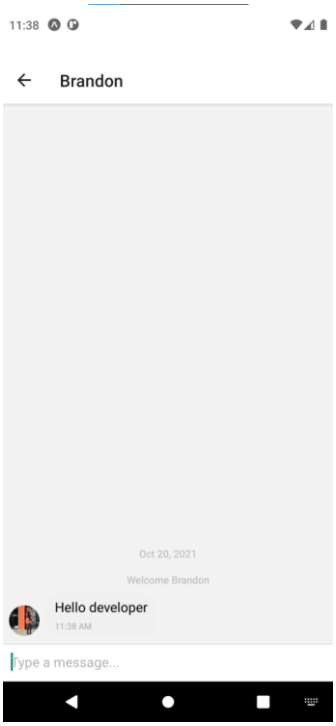
STEP 1: Create the general layout and style for the start screen (home) and the chat screen

I started with understanding the design specifications for the home screen component. Then mapped out each section of the component to implement these designs. I would then repeatedly test the functionality of this component by using Android Emulator checking background colors and name matching.



STEP 2: Using Gifted Chat, create a chat user interface

First, I rendered in Gifted Chat in my chat component so that I can display static messages for the app. This would also allow me to send messages on my end (Android Emulator), but they would not be stored yet.



STEP 3: Using Google Firestore Database, store user messages

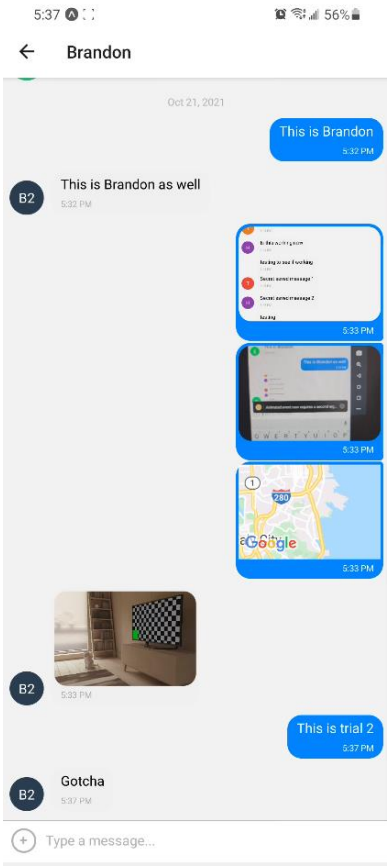
First, I created a Firebase account and created a project with Cloud Firestore. I then followed the steps to create user messages list collection. Finally I implemented Firestore into my project by installing and using the firebase config information to update and fill the database with user messages.

STEP 4: Using React Native's asyncStorage, store user messages offline locally

First, I installed asyncStorage and imported it into my app. I then created functions that store and retrieve messages from Firestore (previous step). I also created a delete function in case I need to delete any messages stored in asyncStorage. Finally, I used NetInfo to check if the user is online. If the user is offline, then it will load the saved messages stored in asyncStorage. If the user is online, they would be put in the authentication process via Firebase then, assuming they pass this process, the app would load the saved messages from Firebase and store new messages locally via asyncStorage.

STEP 5: Implement ability for users to send images and current location

There are guidelines that need to be followed when apps want to use their user's camera/gallery/location/data. It usually falls on the developers to create applications that makes user experience secure when using their applications. Taking these guidelines into consideration, I followed a permission procedure by requesting the user to allow any of these features.



Conclusion

The idea of Chat App was to create a mobile app that allows users to send messages, images, and their location while also being able to store, retrieve, and access their previous messages. Chat App taught me how to use React Native to achieve these features and further my knowledge of JavaScript and database interactions. Future updates would include private chatrooms for users and their requested friends.