

SWEN325 – Assignment 2

cherryryan, 300339331

Written Report

Architecture:

I built my app using 3 layers. There's the Presentation Layer, the Business Layer, and the Data Layer. Each layer has a different purpose and handles different parts of the application.

Presentation Layer

This is the front of my application, it's what defines the look of my application, and how the user can interact with my application. Each screen has its own unique look and style, defined by its render method and style document.

Business Layer

This is where all of my data processing happens. Updates from the Firebase Real Time Database are sent to my application, including new messages/channels that have been created. These updates are handled by the JavaScript code that defines my applications behaviour. Each page has its own unique set of JavaScript methods, which define what happens from user interaction, and how to interact with the Data Layer.

For the above layers, each different page has its own unique JavaScript file, and Stylesheet file (some pages share style sheets to reuse the same code).

Data Layer

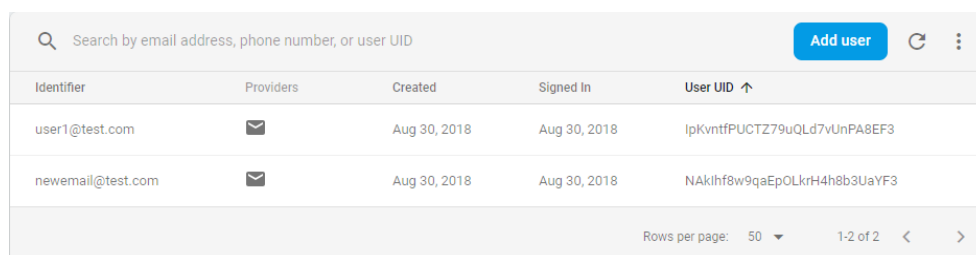
The final layer is my data layer. Nothing has changed since the last assignment, everything is stored in the same format on firebase. This is where all the information created by my application is stored. Messages/channels are sent from the application to Firebase, where they get stored, and then sent back out to each instance of the application. This allowed for real time updates, which is required for a good UX in messaging applications.

External Component - Firebase:

In order to make my app function, I used the Authentication and Real Time Database services provided by Google Firebase. To integrate it into my ionic app, I made use of the firebase npm module, which allowed for me to send and retrieve data from my firebase service.

Authentication:

In order to use my app, users have to create an account with an email and password. I made use of the Email/Password sign in method from Firebase's Authentication module to get this to work. My app will ask the user for an email address, and password, and will then verify this with the Firebase app. Here is what it looks like on the Firebase Console.



Identifier	Providers	Created	Signed In	User UID
user1@test.com		Aug 30, 2018	Aug 30, 2018	IpKvntfPUCTZ79uQLd7vUnPA8EF3
newemail@test.com		Aug 30, 2018	Aug 30, 2018	NAkih8w9qaEpOLkrH4h8b3UaYF3

Rows per page: 50 1-2 of 2

The user can then sign in to the same account, and my app will remember which messages have been sent by you to display them differently to messages sent by other users.

Database:

I store my apps messages and channels on a Real Time Database provided by Firebase. When a new channel is created, the app pushes a new entry into the database under the path channels/. Inside the channels path, there will be an entry for each different channel that has been created, which contains only the name of the channel. This value is used so that my app can display all the available channels and allow for the messages structure to work.

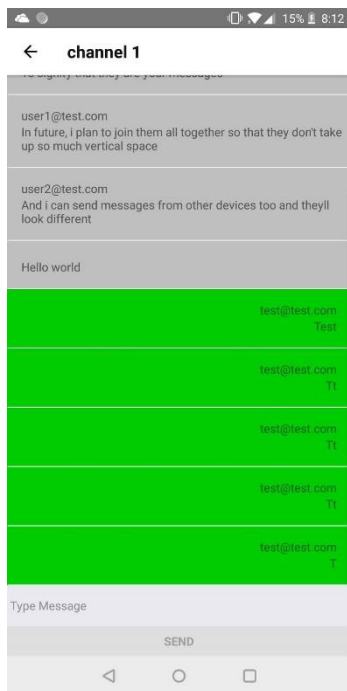
The messages are stored in a similar way, under the path messages/[channel name]/. There is a different path for each channel, which is under its name. Inside each channels path is an entry for each different message that has been sent in that channel. With this update of the application, I changed how messages are stored. Instead of containing the message, userID, and username, the database now only cares about the message and username. This is because the application no longer allows you to update your email or password, so the users email will stay the same throughout their use of the application.



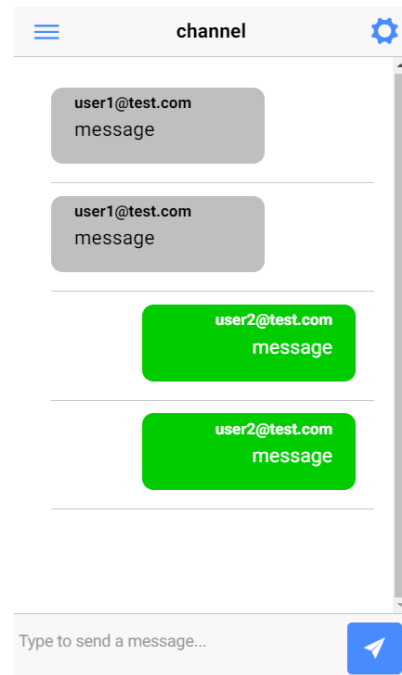
UX Decisions:

Updated Message Design:

I wasn't too happy with the final design for the messages presented by my Ionic App. The bubbles presented a familiar style for messaging apps, such as Facebook Messenger, and SMS apps on smart phones. However, they take up a lot of vertical space in their current form, which means that the user can't see many messages on the screen. I wanted swap back to a better implementation of my original plan of a Slack/Discord style message, which would take up less vertical space on the screen. In my Ionic App I played around with the placement of messages, but I never touched the color of the messages.



The result, is the right. The messages are easy to distinguish, easy to read, and display more messages on the screen at once. It improves the ux more than what I had before it, and it brings it in line with how I originally wanted the application to look as well.

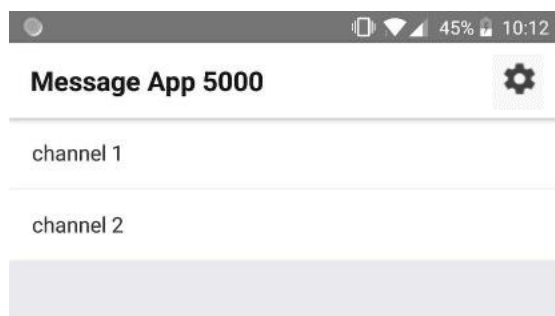
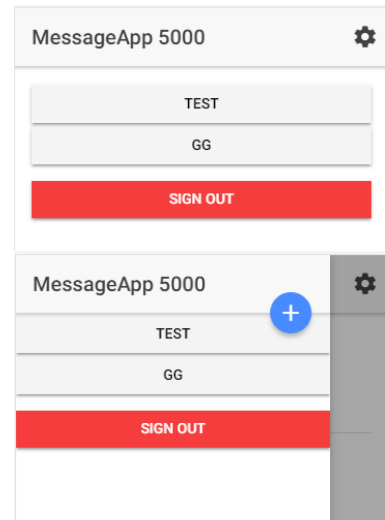


The first thing I tried was the Discord/Slack style of messages, but with the colors used in my Ionic Application. The result is more messages on the screen, but the green makes it hard to read the messages you have sent. To improve this, I played around with the colors that were used, to try and make things easier on the users eyes.



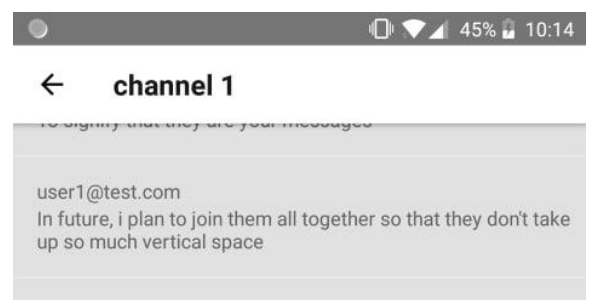
Removal of the Side Menu, Using the Channel List instead:

Looking at how my Ionic app functioned, the Channel List that greets you upon logging in to the application wasn't used very much. Its functionality was exactly the same as the side menu that you accessed through the Channels pages. Without changing the UX too much, I figured that the two could be combined and have all its functionality be merged.



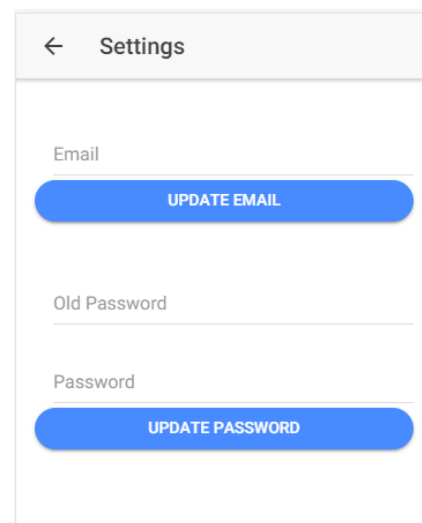
My first merge idea was to remove the side menu entirely. This presented a problem, however, as there was no good place to put the ability to create a channel. In Ionic, this functionality was only in the side menu, and as the settings button was taking the top right corner. This option needed more work to be finalised.

For my final design, I decided that it would be better to have the add channel button on the top right, and to rework how I handled the settings menu. Now the Channel List Screen has a way to create a channel, a feature that was lacking on the Ionic Design, and you access the Channel List the same way you would with the Ionic App.

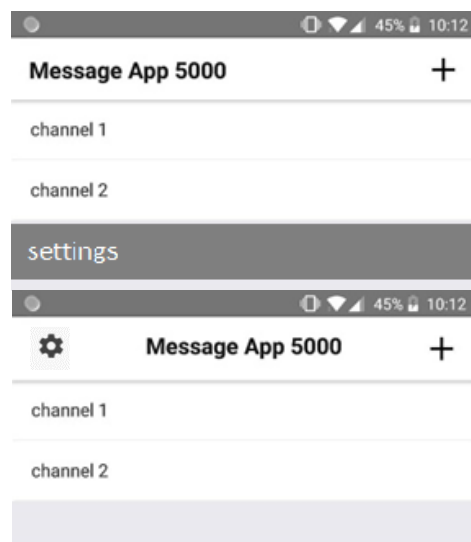


Changes to Settings:

The settings menu in the Ionic Application was kind of tacked onto the application at the last minute. Due to the changes to the Channel List, it needed a new, and better home.



A mockup of a settings screen. At the top is a header bar with a back arrow and the title "Settings". Below the header are two input fields: "Email" and "Old Password". Under the "Email" field is a blue button labeled "UPDATE EMAIL". Under the "Old Password" field is another blue button labeled "UPDATE PASSWORD".



I came up with two solutions to the placement of the settings button. First, was an inline button to take you to the settings menu under the channel. It would have a different color to signify that its not a channel. The second was to move the header to the center, and have a settings wheel on the top left. While playing around with ideas though, I started to question whether the settings menu was still required.

In the end, I deemed that I was altering too much of the user experience by trying to place the settings menu, that it didn't actually benefit me enough to place it in. The email is never verified, so the user can lose the email and still have access to their account. The most important functionality lost was that the user can no longer update their password. However, because of the way users sign in, they can never alter the password without first knowing the password anyways. So in the end, if they had forgotten their password, their account can never be recovered. In future, reworking how user account work would be a high priority, as having a secure, recoverable account is important for the user experience.

