IM3080 Design and Innovation Project

(AY2021/22 Semester 1)

Project Report

Title: myHealth

Github: https://github.com/naboleh/myHealth

Submitted by: Group 5

Supervisor: Assoc Prof Erry Gunawan

Table of Content

1. Introduction

1.1 Background

With the increase in use of technology in healthcare, the industry has turned to mobile applications to complement the supply of personalised healthcare [1]. These applications have the potential to bring convenience and interactiveness to various healthcare services such as booking appointments and health events. One such example would be the HealthHub application developed by the Ministry of Health (MOH) and the Health Promotion Board (HPB). The main aim of HealthHub is to use technology to connect citizens with healthcare institutions. Users can access their personal health-related information such as past prescriptions, lab results, hospitalisation discharge information and appointment history. It encourages engagement between patients, caregivers and medical providers. Another aim of the app was to promote health literacy with access to health & wellness content written for local audiences. However, while applications like HealthHub provide various services for users, our team recognised a lack of intuitiveness when using the applications, particularly for the older generation. Thus, our team intends to create a healthcare application that aims to use technology to extend health services to users in a simple and intuitive manner.

1.2 Motivation

Phone applications and smart devices have been developed to ease the process of acquiring medical services and improve health self-management.

Furthermore, the COVID-19 pandemic has created a rapidly changing environment in terms of healthcare protocols. Singapore's battle with the latest wave of infections has put pressure on healthcare systems. Safe management measures and other restrictions have resulted in a longer waiting time which may cause difficulty for seniors and the immunocompromised to go out and physically visit a doctor.

Currently, there are no government-backed teleconsultation services available in Singapore. There is a need for more centralised access to public healthcare services. With that in mind, our team plans to develop an engaging and user-friendly mobile application which provides online medical services as well as health self-management features. We aim to achieve a higher rate of medical care for Singaporeans.

2. Objectives

As our health application was inspired by HealthHub, our team tested it to figure out what needed to be improved on while keeping our target audience, Singaporeans, in mind.

We decided on 4 key design principles for our project. The first principle is intuitiveness. This allows users to have an easier time navigating and completing the key processes within the application. The second principle is making it easy for users to find what they need in the application. The third principle

is having a lightweight application that has valuable features. The fourth principle is ensuring the application has a nurturing tone.

Therefore, the aim of this project is to develop an application, myHealth, with an improved navigation and overall structure compared to HealthHub. We intend to achieve this by revising the user interface (UI) and design, as well as implementing new features that will be helpful for users in light of the current COVID-19 situation.

3. Literature and Technology Review

3.1 React Native Through Android Studio

React Native is an open-source UI software network created by Facebook. Android Studio is used to set up the necessary environment to build the React Native application on. React Native goes by the 'Learn Once, Write Anywhere' approach [2]. This means that by using a single Javascript code base, it is possible to build an application for IOS, Android and the web on React Native.

The software renders to the native platform which makes it highly dynamic and responsive to user input. On top of that, this enables access to platform features such as the phone camera and user's location and supports the 'live reload' feature which allows users to immediately see the result of the latest changes that they have made to the code, hence there is no need to recompile after each change. Since React Native is an open-source software, it uses open source packages, therefore, it is possible to import building block components including buttons, view, texts and images.

3.2 Vonage and SQLite

Vonage is a cloud-based communications provider which allows developers to integrate real-time messaging, voice and video into applications without building backend infrastructure. This is done by implementing Vonage's comprehensive suite of programmable communications APIs.

SQLite is a serverless, self-contained database management system. As it is serverless, the application is allowed to directly access the database files and no configuration is needed for files. Being self-contained means very little support is required from the operating system.

3.3 HealthHub and Other Telemedicine Applications

HealthHub is an initiative by the MOH Singapore to provide a one-stop portal for Singaporeans to access their health records and services. The purpose of HealthHub is to allow Singaporeans to take charge of their own health and wellness "through the online access of personalized health records, better health literacy and adoption of healthy lifestyle practices." [3]

However, unlike other telemedicine mobile applications that are currently available, HealthHub does not have an e-consultation service. Applications like Doctor Anywhere and MaNaDr provide online consultation with doctors so that users do not need to leave the comfort of their own home. Users are able to access a variety of healthcare and wellness services to cater to their healthcare needs including opting for medication delivery.

4. Design and Implementation

4.1 Design Teardown and Consideration

While navigating through HealthHub, our overall consensus was that the application design lacked user friendliness and had poor style consistency. These were mainly reflected in terms of clarity and layout of the functions, fonts and colour scheme. The application was also used to deliver plenty of information but lacked the use of graphics to balance out the content.

We identified seven colours across the pages of Healthhub (Refer to Appendix A for full teardown). Although these colours were generally vibrant and reflected the friendliness of the application, the number of hues can confuse the eye, resulting in users being unsure of where to begin [4]. Furthermore, the icons used were different for the various pages. The icons in the bottom navigation bar do not provide a clear description of its functions unless selected. The lack of information makes it difficult for users, especially seniors, to navigate intuitively across frequently used features, defeating its purpose of easier navigation.

In terms of layout, the margin between the buttons on the homepage are very narrow, requiring careful navigation by users. In addition, the chosen functions such as “Children’s Health” and “Family and Friends Health” might not be applicable for all users hence should be replaced with a function of higher priority.

Our team decided that we wanted the myHeath application to portray a welcoming and friendly look and feel to it. This serves the purpose of encouraging users to use the application and ensure that checking on their health need not be something to be afraid of. According to [5], most patients tend to neglect their health due to the lack of time, as well as the fear of obtaining bad news when visiting a doctor. Thus, we wanted to create an application environment that gives convenience and allows users to be comfortable with their health issues.

4.2 Functions Teardown and Consideration

Apart from the UI, we felt that the user experience of the application could be improved. The functions of the application are not well organised and could not be navigated intuitively by the user. There could be better classification of functions and the visual hierarchy of functions in the application should be made more distinct to facilitate a smooth and pleasant user experience throughout using the application.

As stated in our objectives, visiting a physical doctor has become more inconvenient during the COVID-19 pandemic. Thus, we decided to enhance users' experience and improve convenience by proposing an e-consultation function that is able to improve efficiency and cost-effectiveness apart from convenience. Moreover, we established that the application would be a one-stop portal for all things health, thus we decided to keep the following functions inspired by HealthHub:

1. Appointments - booking appointments for physical consultations
2. Payments - paying for medical bills
3. Caregiver - checking records of other members of the family who may not be as tech-savvy

4.3 Final Design and Functions

After much discussion and revision, our team finalised on the designs and functions of myHealth (Figure 1). We wanted to ensure that the designs and features of this new application complemented one another and that the 4 key principles (Refer to Objectives section) were addressed.

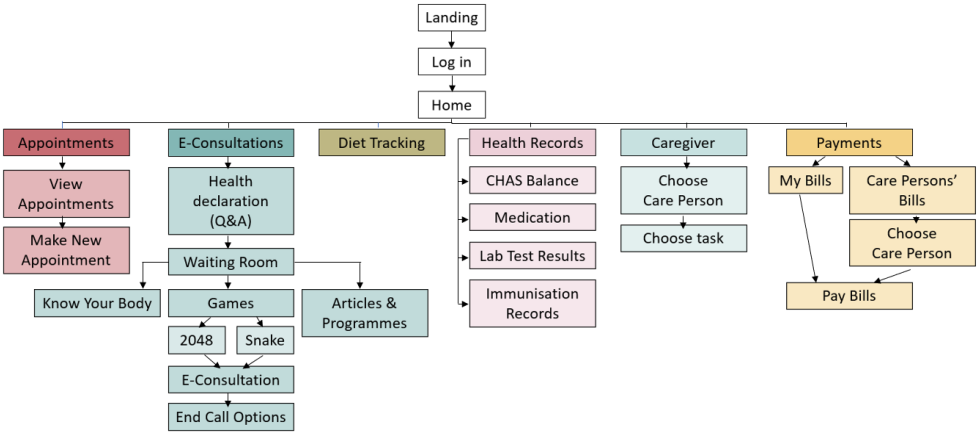


Figure 1: Block Diagram of main flow of application

4.3.1 Final Design Assets

Our team decided on a muted pastel colour palette of five colours (refer to Appendix B for final design assets) to add vibrancy but not overwhelm users. These colours were also used for the various main functions on our homepage to provide clarity in differentiating them (Figure 2).

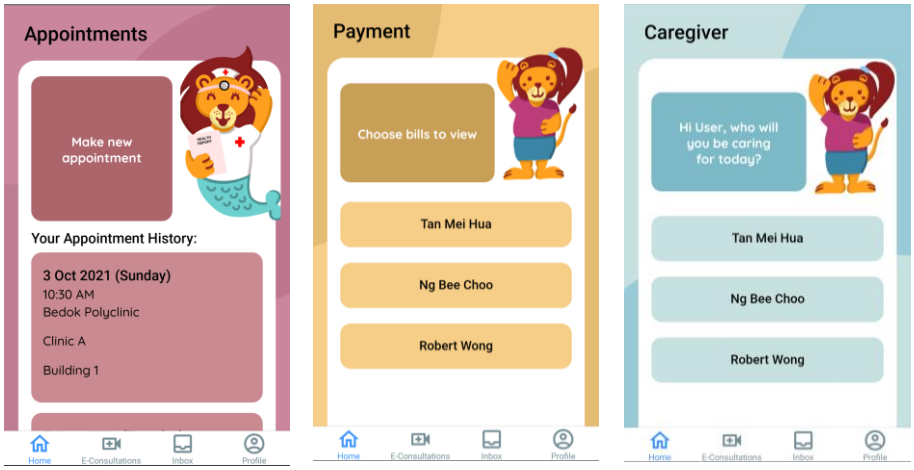


Figure 2: Use of colour palette for different functions

The icons were redesigned to have a consistent modern style. In particular, for the bottom navigation bar, the icons were accompanied with captions below so users can easily identify and access the different functions.

In terms of typography, we picked sans-serif typefaces, Roboto and Quicksand, for a modern and crisp finish. We used Roboto Medium for content title and Quicksand for body text as it is legible to use in small sizes. These fonts complement each other and also ensure readability with their clean lines and sharp edges. Moreover, they give off an approachable and friendly feeling [6], further emphasising the application environment our team intended for.

As myHealth is designed for Singaporeans, we added a friendly local touch to it and chose Singapore Identity as our overarching theme. This includes a landing page with familiar infrastructures like HDB flats and the iconic mosaic dragon playground (refer to Appendix B for graphics).

We also created our mascots based off Singapore's national animal, the lion, so users can easily build a connection with it. Our main mascot, Merly, is a merlion nurse that welcomes the user on the landing page (Figure 3) and appears on multiple pages to mimic face-to-face interaction with users as they navigate through the application.

Figure 3: Landing page of myHealth with graphics

Other mascots include a family of lions: father lion, mother lion, boy lion, girl lion, grandma lion (Refer to Appendix B for mascots). This family represents the members of a typical Singaporean household which ties in with the "caregiver" function in the application.

4.3.2 Final Function

With the feedback given by our supervisor, our team decided to focus on the E-consultation function and deemed it as the main selling point of myHealth. The flow of the function mimics a real-life visit to the doctor, which consists of a Waiting Room and a Question & Answer (Q&A) section to gather information on the users' symptoms before connecting them to a doctor via video call and chat.

With the idea of making visiting a doctor less dull for patients in mind, we expanded on the Waiting Room by adding three components: Know Your Body, Articles and Programmes, and Games (Figure 4). These components appear after answering the Q&A, while users are put on hold for the server to connect them to a doctor.

Figure 4: Waiting room under E-consultation function

Each of the three components increase interaction with users through various means (refer to Appendix B for screenshots of the three components). When waiting for physical consultations, patients are able to pass the time by using their mobile phones to surf the Internet or play mobile games. However, when already using their phones, they may not exit the application in fear of disconnecting from the e-consultation session and hence do not have other means to consume time.

Know Your Body provides information on the human body, along with various health statistics in Singapore. This provides guidance and details on how users can take care of their body.

Articles and Programmes consists of health-related articles and ongoing events conducted by the Singapore HPB. This allows users to catch up on the latest updates relating to Singapore's healthcare system.

Finally, the Games component includes two games: Snake and 2048, a more engaging alternative. Elements of health are added to the game to ensure consistency with the objective of the application.

When users are connected to a doctor, a pop-up appears for users to start the video call. Users can also choose to converse with the doctor via chat (More information on the implementation and functionality of video call and chat in section 4.4).

Once the consultation is over, disconnecting from the call brings users to a page with the bills for the consultation. Users choose to pay through various methods before exiting the function and returning to the Homepage.

4.4 Implementation

As mentioned in the Literature and Technology Review section, we made use of React Native with JavaScript for our coding of the general development of myHealth (Refer to Appendix C for screenshots of source codes). As it is widely used by many software developers for application development, resources could be easily found online. In addition, due to its open-source nature, basic building blocks available could be used for the implementation of our features. These speed up the process of our application development, allowing us to focus on more challenging parts of the application.

Vonage APIs are used for the implementation of the chat and video call functions in our E-Consultation feature. The video call function developed enables users to turn on and off the camera according to their level of comfort. With respect to the situation, users could also toggle to the chat function easily to communicate through texting when needed.

For our database system, we used SQLite DB Browser to set up and configure our database tables. The backend connection with react-native was established using TypeScript and SQLite storage packages. User details such as name and current symptoms will be collected from the function pages respectively and stored in the database through SQL CRUD instructions.

To facilitate collaboration between the team members to develop myHealth, Github is used to enable all members to work on various parts of the application remotely.

4.5 Discussion

4.5.1 Challenges Faced

At the start of our project, our team was unsure as to which software we will be using to build our app. We discussed and decided upon using OutSystems, a low-code platform that provides tools for the development and deployment of apps. Using OutSystems came with its pros and cons. The pros are that not much coding is required and building applications is relatively simple. Many simple features such as app registration and login were automatically generated with just a few clicks. However, there were not many available resources that we could rely on to help implement the features we needed. Moreover,

creating the UI for our application was also a challenge as there were limitations on positioning the UI elements such as not having a simple way to overlap elements. Having no answers to the UI part, we ended up switching to the React Native framework in favour of its large documentation and open-source resources that we can take advantage of.

That leads to the second challenge that we faced. Since React Native is new to everyone in the team and uses JavaScript as its coding language which not many are familiar with, we have to essentially start learning from scratch. Having no prior knowledge, it was quite challenging at the start. At times we were doubting if we should again change platform and use Android Studio instead since mostly everyone has some experience working on it. However, with the help of the many available online resources, we managed to push forth and adapt to the new framework to develop our app.

Another challenge that we faced during our project was the limited time we have to implement the many features that we initially planned to achieve. Originally, we planned to add 4 new functions to the 1.0 app. That being the E-Consultation, Health Tracking, MyResponder-like feature, and a scanner that lets you find out more information on certain medications. Not knowing how tight the schedule is going to be, we were quite ambitious on our goal. However, advice from the professor to narrow down our scope to just 1 or 2 main features along with the gradual realization of how much time we have made us reconsider our choice. In the end, our team decided to focus on implementing the E-Consultation feature.

4.5.2 Limitations

The application is still just a prototype with many dummy elements such as buttons and pages that are only for show. While it does provide a relatively complete view of the flow of the app, many functions from the 1.0 application are still not a working implementation. For users looking for a complete experience of a one-stop health app, the myHealth app still falls short.

A limitation with the E-Consultation feature is that while it is working, there are still occasional bugs that plague the chat and video-calling function. Since the application was not built for release, the same token and session key for the communication API used to implement the E-Consultation feature is shared amongst users. The token and session key has to be manually renewed every 30 days for it to continue working. Since everyone using the app shares the same session key, there is essentially only 1 room for everyone. A chat message sent through the app is received by all users of the app and a video call can be accepted by any user. The E-Consultation feature also by right requires the implementation of 2 sides: doctors and patients. However, due to the limitation of time, only the patient side is implemented which is to say that the feature is not complete. Due to this, a timer has to be implemented to simulate the doctor's call on the app.

Another limitation of the app is regarding the games. Since React Native is not optimised to handle games, there are very limited choices as to what kind of games can be implemented in our app. While it is not that big of an issue for those with low requirements such as puzzle games that only require a

few elements, other choices are not an option. Games that require many things to be rendered at once or fast-moving animations will heavily impact the performance of the app. Hence the app is limited in terms of the game choices that can be implemented on it.

5. Conclusion and Recommendation

5.1 Conclusion

Our initial objectives were to create a one-stop health services application, with four additional features, that is intuitive, easy to navigate, lightweight with valuable features and has a nurturing tone.

Overall, we improved on the layout and structure of the application so that the navigation is more intuitive for the users. Of the above four additional features, we were only able to fully implement one. myHealth cut down on the information offered in the original HealthHub application to be more valuable. The introduction of mascots and personalisation by addressing the users by name contribute to create a nurturing tone.

Planning is a vital part of the development cycle, however, with the consistent advice and feedback from the team's supervisor, many changes were made throughout the development process. The incremental model of software development, in which the product is designed, implemented and tested incrementally, was a good choice for the development of this application. The team was able to split the project into smaller functions to develop and improve it in concurrent iterations which allowed more flexibility in adding features that were later decided on. It also allowed the team to learn from previous iterations and improve on the next round.

5.2 Recommendation for Future Work

5.2.1 Feedback from target users

The team used reviews of the inspiration applications from the Google Play Store to better understand and improve the application. As many of the reviews mentioned the poor UI and the counter intuitiveness of the application, the team ensured that design was done with the user in mind. These reviews ensured that the team implemented friendlier UI/UX across myHealth, including design consistency, personalisation and breaking down user actions. To further improve the accessibility of our application, feedback from user testing and surveys provided more insight on pain points with the original application.

5.2.2 Develop the appointments feature fully

The appointments feature is marketed as the main function of the original HealthHub application. However, one of the main concerns that users have according to its reviews on the Google Play Store

is that this function has many broken links and is not very responsive, leading to long waiting times to book an appointment. The team redesigned the structure of the appointments feature in myHealth to make the flow more instinctive, by prioritising one main action per screen and reducing the number of steps to book an appointment. However, myHealth does not currently use a database, and as such, users can book any slot, regardless of whether another booking has already been made. As such, an integral part of the original application remains incomplete.

5.2.3 Create an onboarding process

The team's aim was for the application to be usable by everyone in Singapore, even those who may not be tech-savvy. The application reduces the amount of content and unnecessary elements greatly, and includes some direct instruction. For example, the pre-consultation form includes direct instruction so that the information required from the user is explicit and straightforward, with no distractions. However, the application could be improved with an optional onboarding process that details the processes with a clear walkthrough for beginner users for parts of the application that are not as standardised with the layout, like the Waiting Room.

5.2.4 Improve on launch speed

As React Native is not optimised to handle games, the launch time of the application increased with the addition of two simple games. As the decision on adding games happened later in the project cycle, it was not realistic to change our development tools with the approaching deadline. Since speed is not a critical part of the application, improving the launch speed was not prioritised. However, if myHealth were to be launched for public use, it would be better to defer delivery and devote more time and research into a tool that can better handle the game feature.

References

[1] G. Kalem and Ç. Turhan, “Mobile technology applications in the healthcare industry for Disease Management and wellness,” *Procedia - Social and Behavioral Sciences*, vol. 195, pp. 2014–2018, 2015.

[2] React Native, https://reactnative.dev/, 2021

[3] HealthHub, https://www.healthhub.sg/about-us, 2021

[4] T. Hornor, “10 troublesome colors to avoid in your advertising,” *SitePoint*, 12-Aug-2020. [Online]. Available: https://www.sitepoint.com/10-troublesome-colors-to-avoid-in-your-advertising/. [Accessed: 10-Nov-2021].

[5] “5 reasons why you're avoiding a check-up, but shouldn't,” *RSS*, 2019. [Online]. Available: https://www.careand.ca/post/5-reasons-why-youre-avoiding-a-check-up-but-shouldnt. [Accessed: 10-Nov-2021].

[6] J. Rinaldi, “Sans serif vs serif font: Which should you use & when?,” *IMPACT Inbound Marketing Agency*, 02-Nov-2021. [Online]. Available: https://www.impactplus.com/blog/sans-serif-vs-serif-font-which-should-you-use-when. [Accessed: 10-Nov-2021].

Appendices

Appendix A

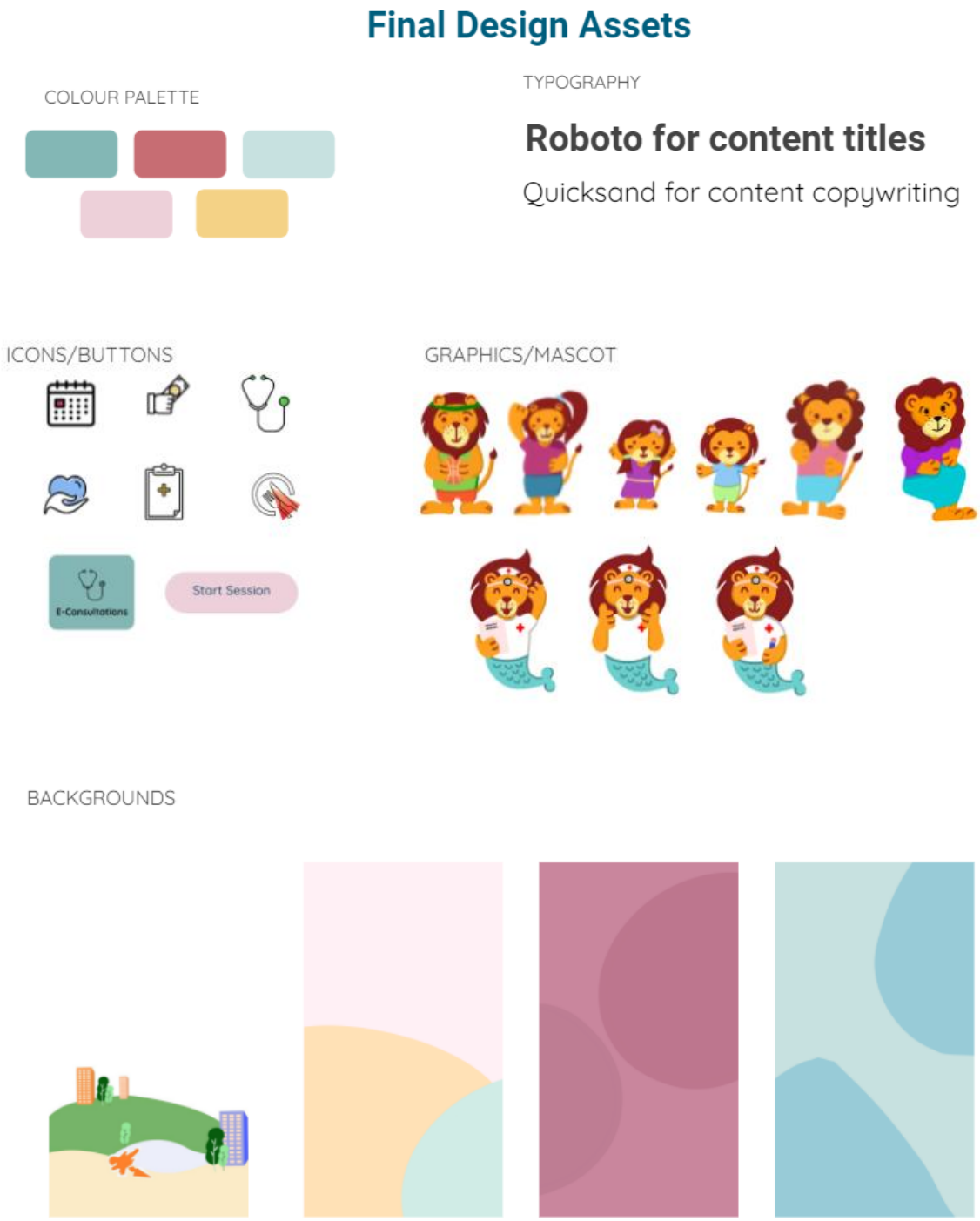
Link to Teardown of HealthHub:
https://docs.google.com/presentation/d/1i28PlDyO7NGq9MLcKhxzc-VjmInfqUoaxcoLuHhjFlk/edit?usp=sharing

Link to Ideation of Application:
https://docs.google.com/presentation/d/1PaVbXs252OGTIfx99j376QKb8nyKq2EWY5Zlu8nTduQ/edit?usp=sharing

Link to Features Prototype Figma:
https://www.figma.com/file/WgKqwYf0I1WzyyMegTZDLC/Presentation-Wireframe?node-id=2291%3A6033

Link to Design Prototype Figma:
https://www.figma.com/file/GXDln9fn071ulmdm5SZBgF/Healthhub-1.0?node-id=0%3A1

Appendix B


Final Design Assets
COLOUR PALETTE
TYPOGRAPHY
Roboto for content titles
Quicksand for content copywriting
ICONS/BUTTONS
E-Consultations
Start Session
GRAPHICS/MASCOT
BACKGROUNDS



Appendix C

GitHub (for full source codes): https://github.com/naboleh/myHealth

Screenshots for some frontend codes

README.md App.js

```
/*
Stack navigator to move between the pages
1. import the page in here (e.g. import HomePage from './src/pages/HomePage';)
2. enter it as a <Stack.Screen ../>
3. in the page's .js file, the touchableopacity/button should have an action (e.g. onPress={() => navigation.navigate('HomePage')})
*/
import * as React from 'react';
import {StyleSheet, View, Text, Image, TouchableOpacity} from 'react-native';

import {NavigationContainer} from '@react-navigation/native';
//import {createNativeStackNavigator} from '@react-navigation/native-stack';
import {createStackNavigator, TransitionPresets} from '@react-navigation/stack';
import {createBottomTabNavigator} from '@react-navigation/bottom-tabs';
import {createDrawerNavigator} from '@react-navigation/drawer';

import LandingPage from './app/pages/LandingPage';
import SingpassLogin from './app/pages/SingpassLogin';
import HomePage from './app/pages/HomePage';
import Appointments from './app/pages/Appointments';
import NewAppt from './app/pages/NewAppt';
import Payment from './app/pages/Payment';
import Bills from './app/pages/Bills';
import EConsultBill from './app/pages/EConsultBill';
import BillsLO from './app/pages/BillsLO';
import Caregiver from './app/pages/Caregiver';
```

(App.js)

```
import HealthRecords from './app/pages/HealthRecords';
import CheckMeal from './app/pages/CheckMeal';
import DietTracking2 from './app/pages/DietTracking2';
import Profile from './app/pages/Profile';
import EConsultsChat from './app/pages/EConsultsChat';
import EConsultsQnASymptoms from './app/pages/EConsultsQnASymptoms';
import EConsultsQnAMedication from './app/pages/EConsultsQnAMedication';
import EConsultsQnAMedName from './app/pages/EConsultsQnAMedName';
import EConsultsQnADrugAllergy from './app/pages/EConsultsQnADrugAllergy';
import EConsultsQnADrugName from './app/pages/EConsultsQnADrugName';
import EConsultsQnAFever from './app/pages/EConsultsQnAFever';
import EConsultsDataPage from './app/pages/EConsultsDataPage';
import EConsultsVideo from './app/pages/EConsultsVideo';
import EConsultsLandingPage from './app/pages/EConsultsLandingPage';
import EConsultsQnAPainPoints from './app/pages/EConsultsQnAPainPoints';
import EConsultsWaitingRoom from './app/pages/EConsultsWaitingRoom';
import EConsultCallEnd from './app/pages/EConsultCallEnd';
import KnowYourBodyLanding from './app/pages/KnowYourBodyLanding';
import KnowYourBody from './app/pages/KnowYourBody';
import KnowYourLungs from './app/pages/KnowYourLungs';
import KnowYourLiver from './app/pages/KnowYourLiver';
import NewsPage from './app/pages/NewsPage';
import Article1 from './app/pages/Article1';
import GamesLandingPage from './app/pages/GamesLandingPage';
```

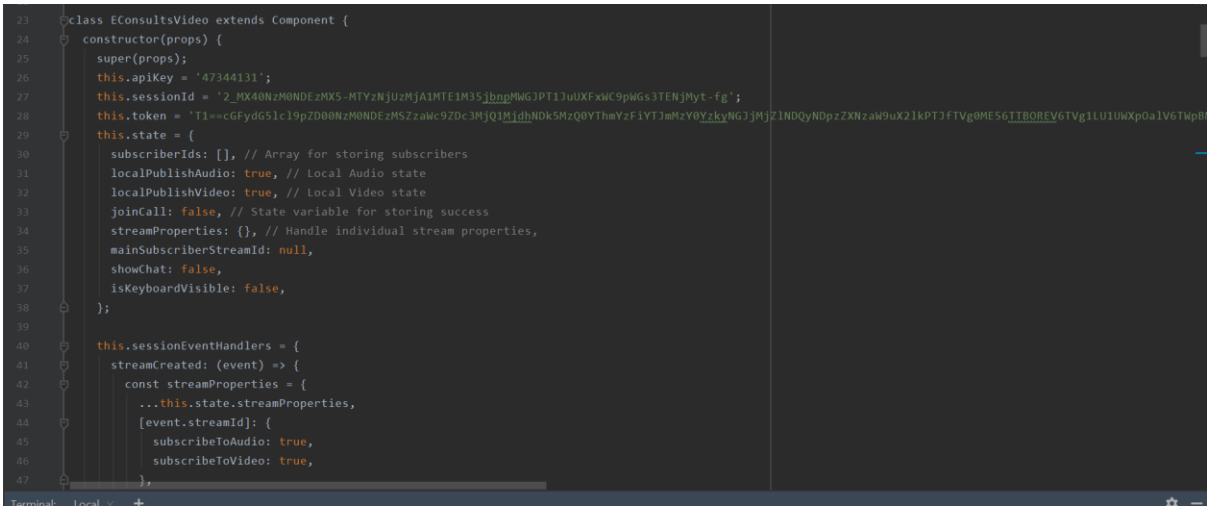
(App.js continued)

```
          },
          headerShown: false,
          drawerActiveBackgroundColor: '#fff',
          drawerActiveTintColor: '#748c94',
          drawerLabelStyle: {color: '#748c94', fontSize: 15},
        }}>
        <Drawer.Screen name="Home" component={HomeStack} />
        <Drawer.Screen name="Settings" component={Settings} />
        <Drawer.Screen name="Logout" component={LandingPage} />
      </Drawer.Navigator>
    );
}


// Put the pages where you want the bottom navbar included, here
//const ScreenWithBarStack = createNativeStackNavigator();
const ScreenWithBarStack = createStackNavigator();
function BarStack() {
  return (
      <ScreenWithBarStack.Navigator>
        <ScreenWithBarStack.Screen
          name="HomePage"
          component={HomePage}
          options={{headerShown: false}}
```

(App.js continued)

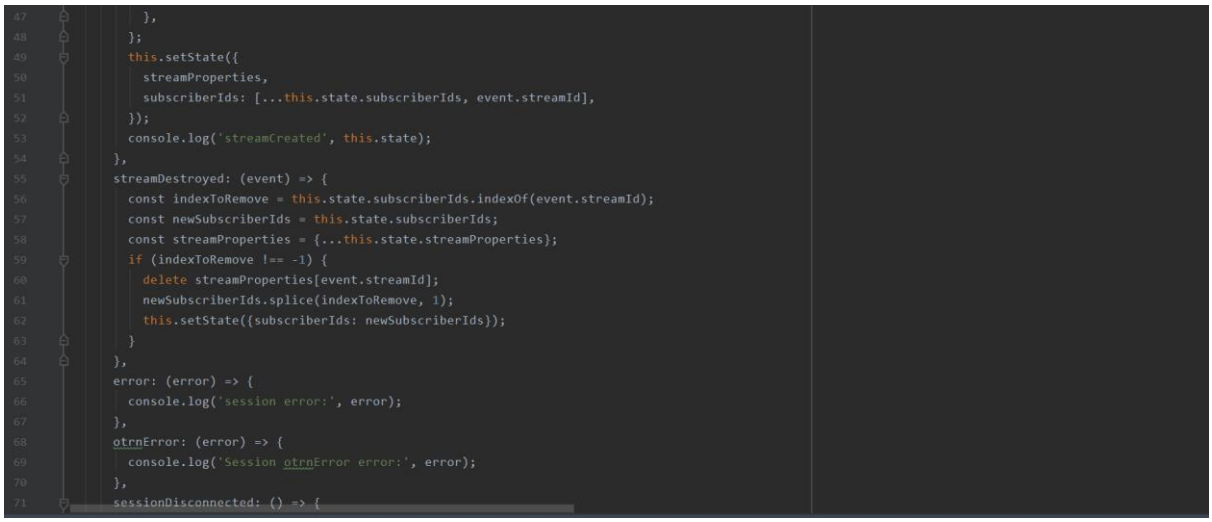
```
import React, { Component } from 'react';
import { SafeAreaView, StyleSheet, ScrollView, View, Text, Dimensions, Button, TouchableOpacity, Image, ImageBackground } from 'react-native';
import { OT, OTSession, OTPublisher, OTSubscriber, OTSubscriberView } from 'opentok-react-native';
import EConsultsChat from './EConsultsChat';
import { Modal } from 'react-native';
import { Keyboard } from 'react-native';

/*
This import is only needed if we want to store API key, Session ID and Token in a separate config.js file.
import * as credentials from './config';
*/

import Icon from 'react-native-vector-icons/MaterialIcons';

const dimensions = {
  width: Dimensions.get('window').width,
  height: Dimensions.get('window').height,
};

const mainSubscribersResolution = {width: 1280, height: 720};
const secondarySubscribersResolution = {width: 352, height: 288};

class EConsultsVideo extends Component {
  constructor(props) {
    super(props);
    this.apiKey = '47344131';
```

(EConsultsVideo.js -- video call component under E-Consultations)



```
class EConsultsVideo extends Component {
  constructor(props) {
    super(props);
    this.apiKey = '47344131';
    this.sessionId = '2_MX40NzM0NDEzMX5-MTYzNjUzMjA1MTE1M35jbnpMWGJPT1JuUXFxWC9pWGs3TENjMyt-fg';
    this.token = 'T1==cGFydG5lcl9pZD00NzM0NDEzMSZzaWc9ZDc3MjQ1MjdhNDk5MzQ0YThmYzFiYTJmMzY0YzkyNGJjMjZlNDQyNDpzZXNzaW9uX2lkPTJfTVg0ME56TTBORFV6TVg1LU1UWXpOalV6TWpB
    this.state = {
      subscriberIds: [], // Array for storing subscribers
      localPublishAudio: true, // Local Audio state
      localPublishVideo: true, // Local Video state
      joinCall: false, // State variable for storing success
      streamProperties: {}, // Handle individual stream properties,
      mainSubscriberStreamId: null,
      showChat: false,
      isKeyboardVisible: false,
    };

    this.sessionEventHandlers = {
      streamCreated: (event) => {
        const streamProperties = {
          ...this.state.streamProperties,
          [event.streamId]: {
            subscribeToAudio: true,
            subscribeToVideo: true,
          },
```

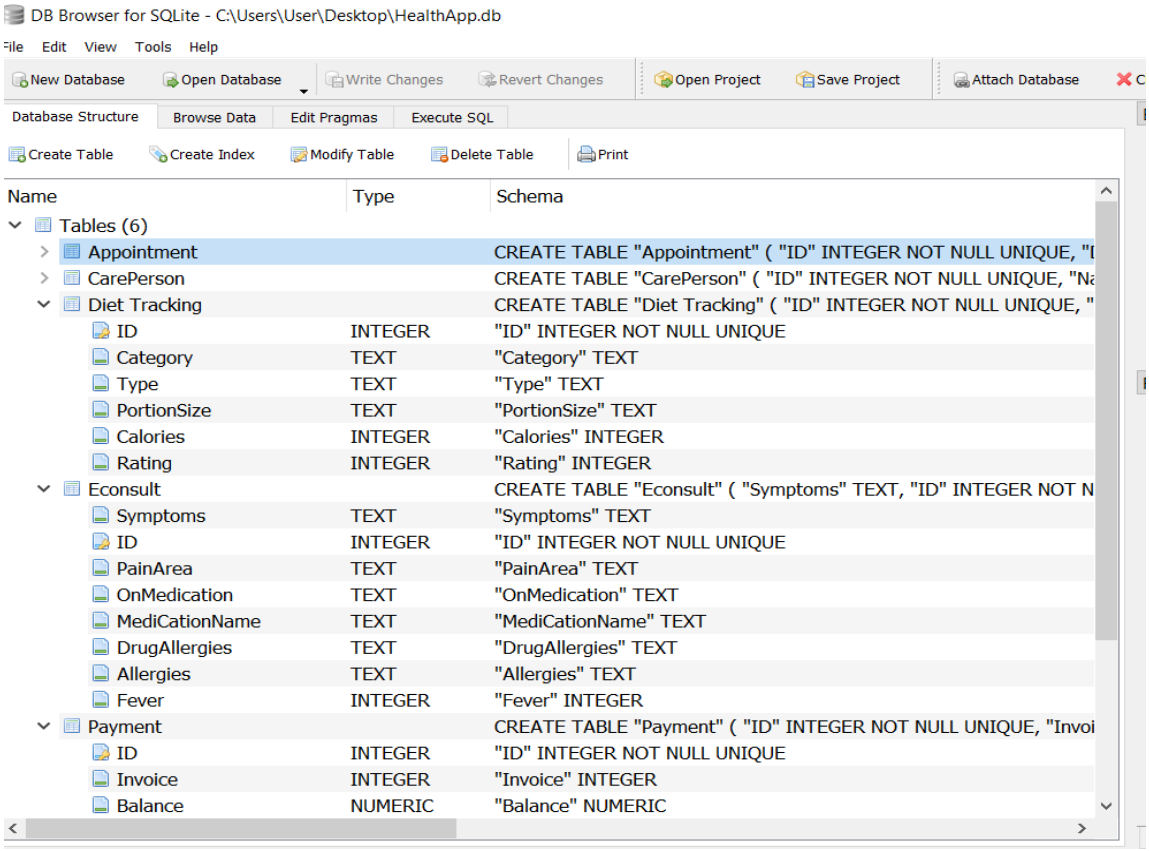
(EConsultsVideo.js continued)



```
          },
        };
        this.setState({
          streamProperties,
          subscriberIds: [...this.state.subscriberIds, event.streamId],
        });
        console.log('streamCreated', this.state);
      },
      streamDestroyed: (event) => {
        const indexToRemove = this.state.subscriberIds.indexOf(event.streamId);
        const newSubscriberIds = this.state.subscriberIds;
        const streamProperties = {...this.state.streamProperties};
        if (indexToRemove !== -1) {
          delete streamProperties[event.streamId];
          newSubscriberIds.splice(indexToRemove, 1);
          this.setState({subscriberIds: newSubscriberIds});
        }
      },
      error: (error) => {
        console.log('session error:', error);
      },
      otrnError: (error) => {
        console.log('Session otrnError error:', error);
      },
      sessionDisconnected: () => {
```

(EConsultsVideo.js continued)

Screenshots for some backend codes



(SQLite Database tables)

```
mport { openDatabase } from 'react-native-sqlite-storage';

onst db = openDatabase({
 name:'MyHealth',
);

xport default function SingpassLogin({navigation}) {

 const [UserName, setUsername] = useState('');
 const [Password, setPassword] = useState('');

 const [name, setoutName] = useState('');
 const [password, setoutPassword] = useState('');

 const createTables=()=>{
   db.transaction(txn =>{
     txn.executeSql(
       'CREATE TABLE IF NOT EXISTS userInfo (id INTEGER PRIMARY KEY AUTOINCREMENT, Name VARCHAR(30), Password VARCHAR(30), Symptoms VAR
       [],
       ()=>{
         console.log('creating table successfully');
       },
       error =>{
         console.log('error on creating table '+error.message);

       },
     );
   });
 };

 const addUser = () =>{
   createTables();
   if(!UserName || !Password){
     alert("Enter category and Password");
     return false;
   }
```

(Backend Connections)