Enterprise Mobility

SE5070

Assignment I

Name: B. R. M. S. R. B. Rathnayake

Student ID: MS21911958

Application Name: *SLIITLIFY Masters*

Technologies: *React Native/Firebase*

Contents

Selection of Project Area

MS21911958 = 21911958

= 2 + 1 + 9 + 1 + 1 + 9 + 5 + 8

= 36 = 3 + 6

= 9 modulo 3

= 0 -> EduTech.

Overview

“SLIITLIFY Masters” is an react-native application built for students who are enrolled to SLIIT master’s degree. External users, referred as “Guests” can get an overview through this app.

Project idea

<u>User - Student</u>

May it be MSc, MBA, MPhil, in whichever stream this app serves the generic purposes for students and lecturers, to ease following up with attendance for lecturers, whilst making it easier for students to attend lectures and mark attendance through the app itself via a QR (quick response) code scanner. The student has direct access to Moodle (courseweb), student profile, SLIIT portal.

Students can login to the system using their MS Number as the username, and NIC as password (the same login is used to view results/grades in official SLIIT result board). This login will be validated by the backend servers. Authenticated students will login, invalid credentials will be rejected with a toast message. Student can slide the dashboard to the right, and view/select the menu options.

Main objective behind the project is to capture student's attendance and to allow the student to view their attendance progress. ***How?*** The lecturers project a QR code during the lecture session, embedded in the middle of the lecture slide. (To make things interesting, students must pay attention to the lectures because the QR can be projected on any random slide.) For any reason if the student was unable to use the app/smartphone, the student needs to inform the lecturer regarding the technical difficulty during the lecture itself. The scanned QR code will directly update the backend, which maintains the attended lectures, total lectures (3 lectures attended out 10 total lectures) along with the enrolled module(s) details.

User – Guest

This mobile app is not limited for just SLIIT masters students. If any outsider from the university or any student who doesn't follow a master's degree wants to use the app, they can simply use the Guest login. Guest login is authenticated with Google authentication with Firebase Auth. The guests can simply login to the app and browse through limited functionality shown on the left-hand side navigator pane. The reason for this login is to ensure that the app has a broad audience, and it could help outsiders to learn more about SLIIT Culture, learn about the offered master's degrees and explore SLIIT vicinities.

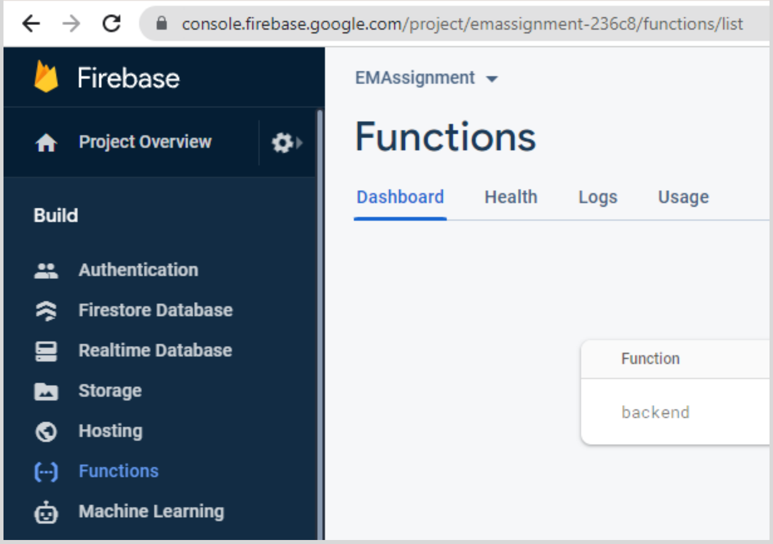
***Assumptions**: SLIIT administrators need to feed sensitive data (student profile pic, MS details, etc.) into the firebase backend, this app works assuming that data setup is readily available.

Project features

Student authentication against Firebase backend.
Ease of user Navigation throughout the application.
User preferences (dark theme/light theme) and app customizations along with animations
Native hardware used: QR code scanner/camera, vibrate, speakers.
Guest users can view faculty vicinities and classroom interiors in 360-degree mode with the aid of gyroscope. Graphical view for students to view current attendance status along with modules enrolled with a given semester/year. Depending on the user login (guest /user) the left-hand side navigator pane will show/ hide certain features and functionalities available for the respective user.

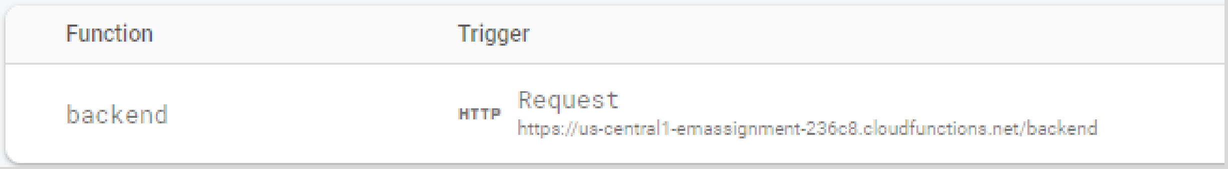
Firebase Clound Functions

Dedicated backend REST API, using cloud functions to perform CRUD operations, can be accessed over any device with an internet connection.



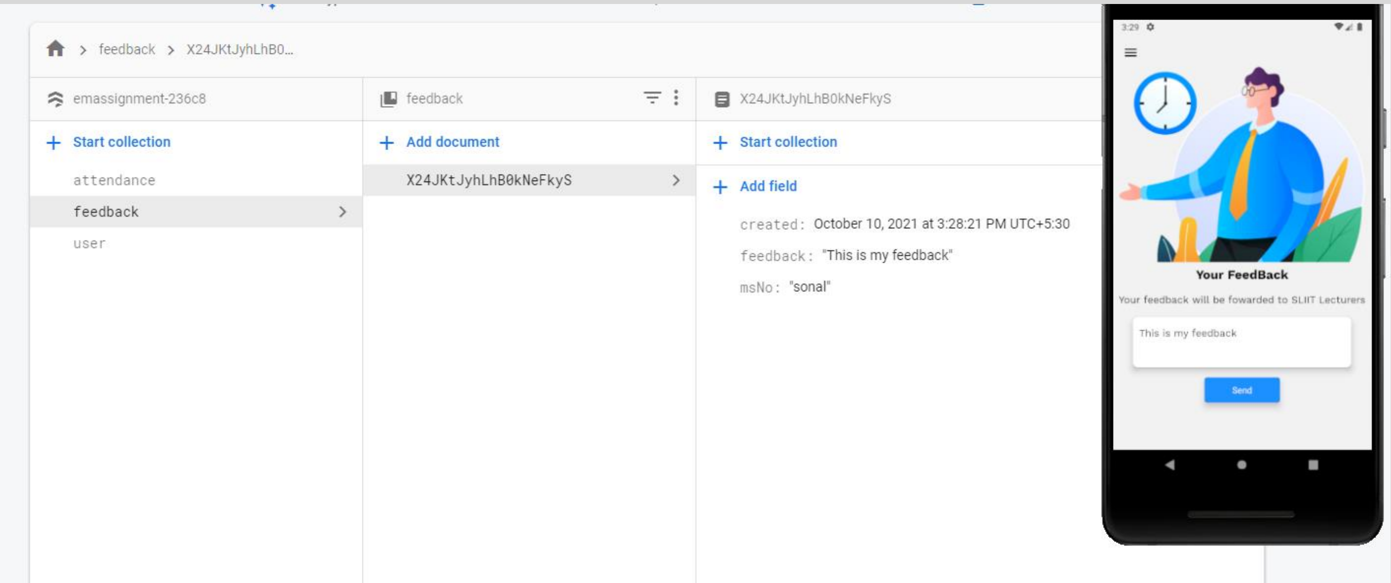
Function
backend

Exposed backend function



Function	Trigger
backend	HTTP Request https://us-central1-emassignment-236c8.cloudfunctions.net/backend

Creating dummy feedback Post requests into the backend



X24JKtJyhLhB0kNeFkyS

created: October 10, 2021 at 3:28:21 PM UTC+5:30
feedback: "This is my feedback"
msNo: "sonal"

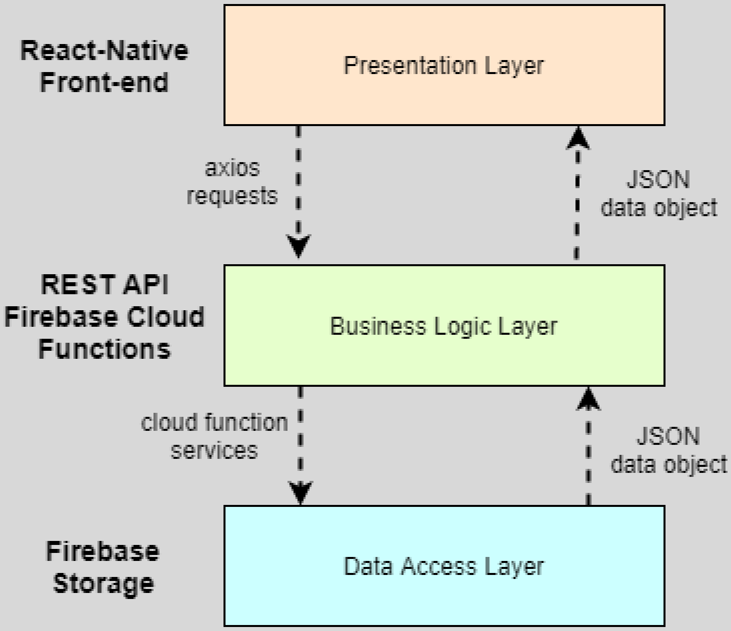
This is my feedback

Application architecture

Major Decisions

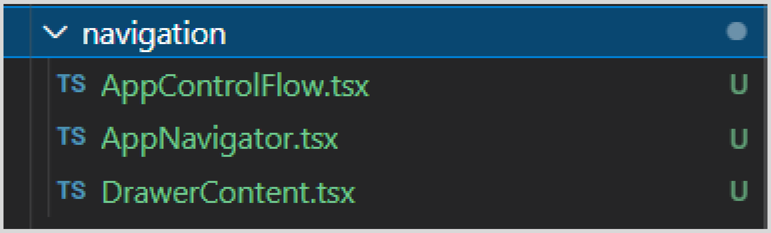
Maintaining a separate backend API to have server-side processing. SLIITLIFY Masters application currently several business logics and functionality, data validations written on the app itself. But in the long-term picture, these validations and processing can be gradually moved to the server side, in hopes of maintaining a dummy client as for the front end.

SLIITLIFY Masters follows three tier layered architecture.



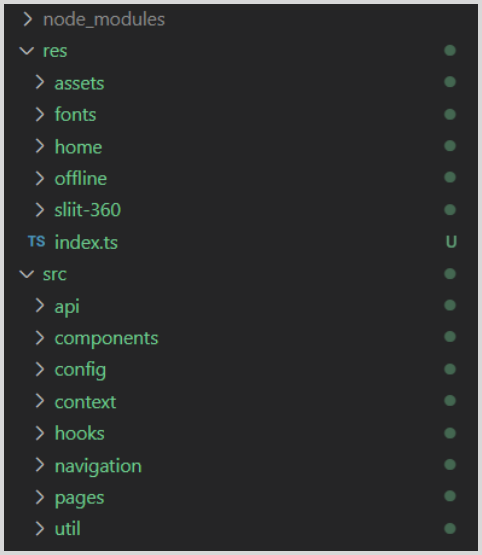
Intent Routing

For routing purposes, static routing has been used inside DrawerContent.tsx



Source code Organization

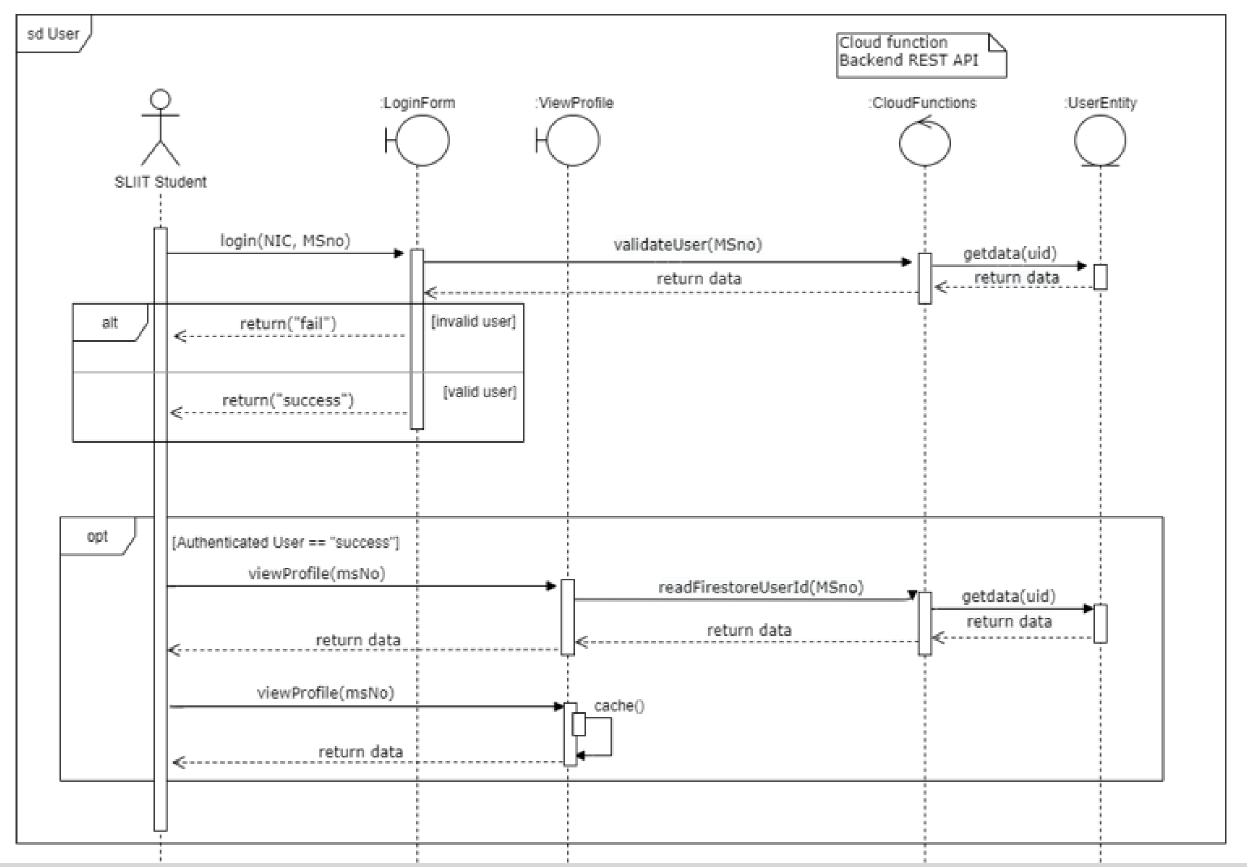
Source code folder structure has been structured based on **Type.**



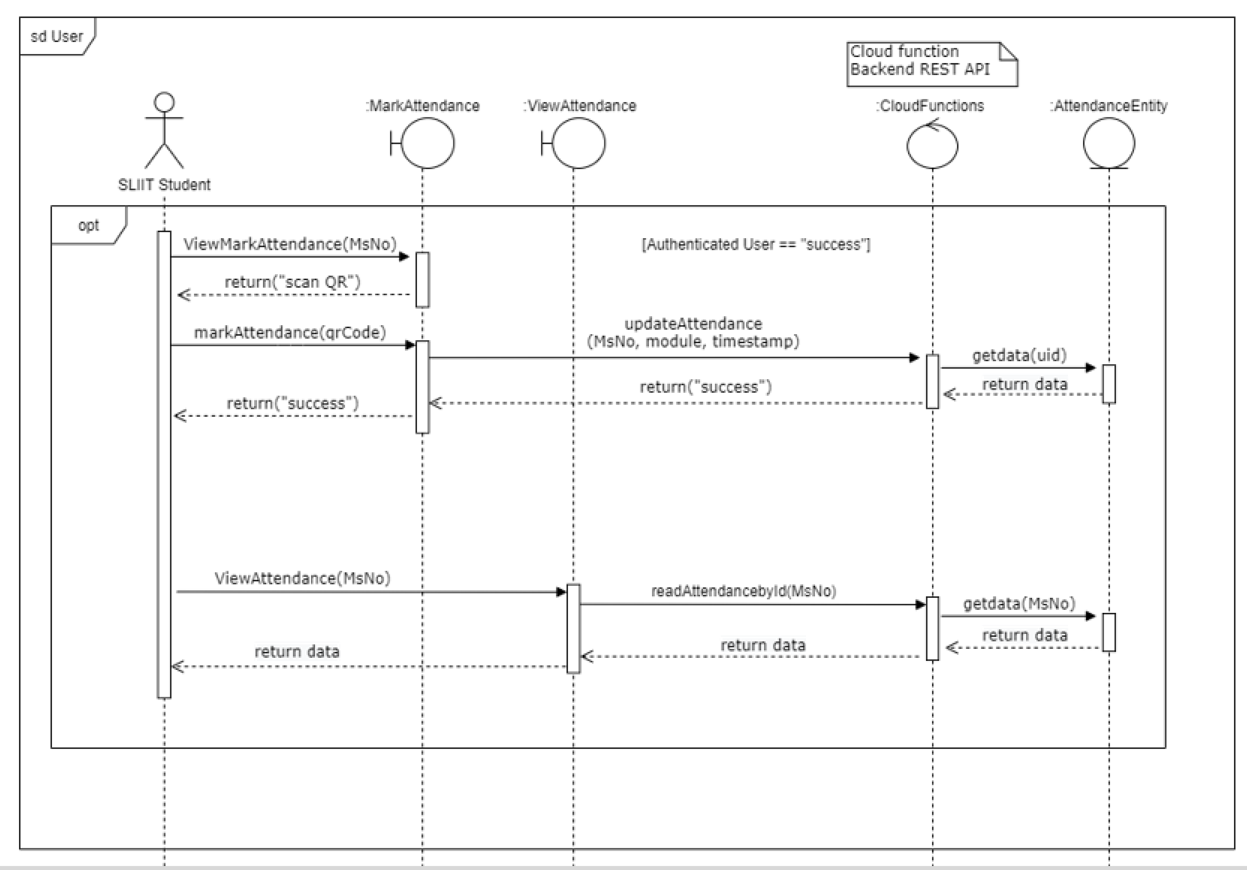
```
> node_modules
v res
  > assets
  > fonts
  > home
  > offline
  > sliit-360
  TS index.ts                U
v src
  > api
  > components
  > config
  > context
  > hooks
  > navigation
  > pages
  > util
```

Data Flow

User Sequence Diagram

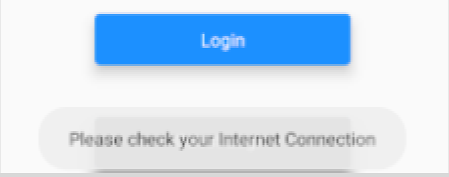


Attendance Sequence Diagram



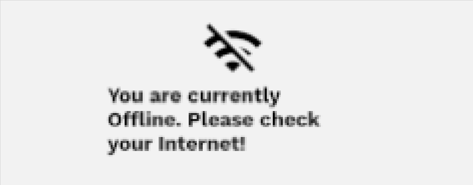
Offline access technology

React-native-offline was used to handle users that are offline. A toast message is notified to the user once they are offline,



Please check your Internet Connection

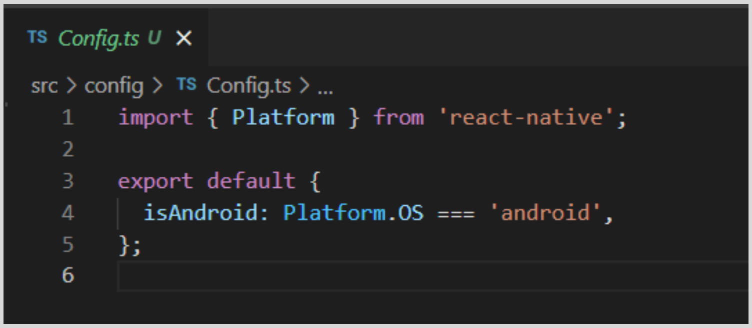
Separate offline custom component **<OfflineApp/>** which can be wrapped around data fetching elements. If the internet connection is stable, the element (children within a fragment) will be displayed, else the below offline text message is displayed. Once the connection is active, data will be cached with firebase persistence.



You are currently Offline. Please check your Internet!

Design constraints and principles followed

Currently, the project is configured and tested to run on Android platforms inside the **Config.ts**



```
import { Platform } from 'react-native';

export default {
  isAndroid: Platform.OS === 'android',
};

```

From a development perspective, the front-end application was built using Typescript, JavaScript and React which is then compiled to Native code to support android/iOS platforms. For iOS purposes, change the **Config.ts** file from android to iOS. Such instance is shown below to adapt to Apple smartphone display sizes.

```
const marginTop = Platform.OS === 'ios’? 20 : StatusBar.currentHeight;
```

Efforts for a thin mobile client approach can be improved in further steps, the backend server can handle most of the business logic. As an improvement, we can allocate load balancers to enhance the response times. Data synchronization methodology used was On-demand and cached for later used to prevent excessive loading.

User Agility – One Handed usage for the app, the navigator drawer can be accessed from a default thumb zone. Simple thumb gestures with easy memorization and ease of use

Buttons are fed with motion feedback, animation transition added on user picture to denote layout changes and sound emitted on screen menu tap. Simple clear icons to enhance visibility, with globally followed conventions. Anti-pattern Hamburger menu was used for quick user navigation since there are several menus

Communication protocol used - HTTP REST with AXIOS Library

Conditional return statements used to show user a Loading screen until all data is fetched from the database.

Other Constraints – The application is ideal for a portrait mobile device usage. Minimum of Android version 4.4.4 above is required, 2 GB RAM (considering other applications running in the background) and bare-minimum of 100MB storage space to install the application.

Integration hardware

Hybrid mobile application development that compiles into native code which can then access native features and device hardware. SLIITLIFY Masters delegates certain features to mobile’s native hardware to take use of them.

Camera Usage

To mark attendance, the application takes use of the camera device, to scan the QR code projected by the lecturer during the lecture. This QR would contain the module id, date/time slot, lecturer name and capture’s the student’s MS number. All these data combined, the app would send a POST request to the backend API to update the student’s attendance for that module, which increments the “attended” column by 1.

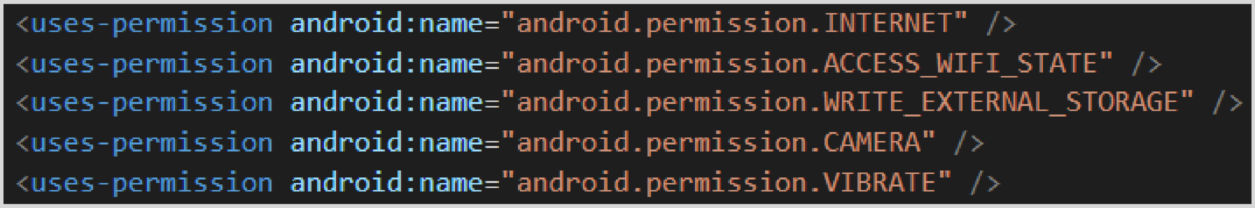
Speaker/Vibration Usage

User interactions have mini vibrations embedded and sounds enabled on tapping on the main drawer. User interactions when tapping on the main drawer

Gyroscope Usage

Guests can view SLIIT premises in 360-degree mode. As they tilt their phone, the image rotates. This feature is very helpful for outsiders who have not physically visited SLIIT vicinities.

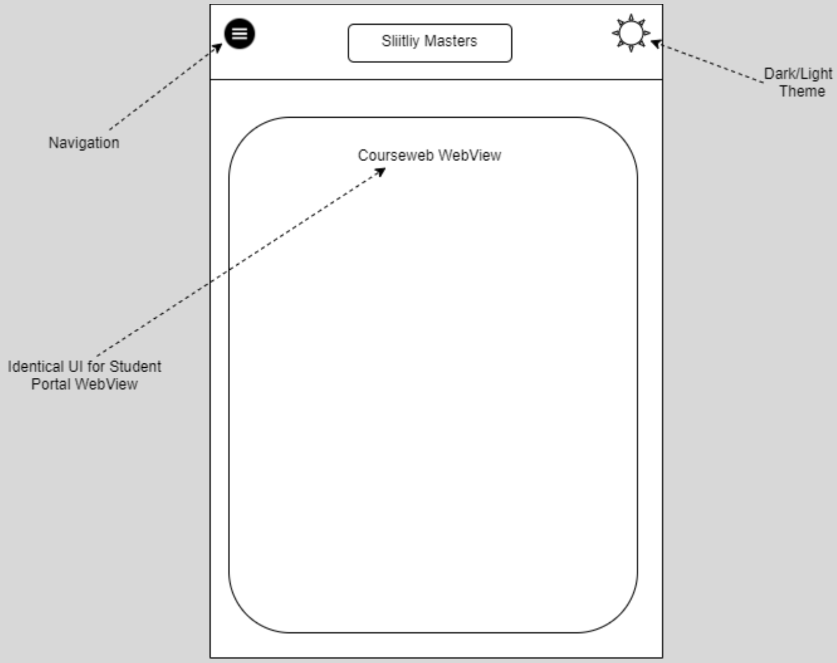
Permissions grants for AndroidManifest.xml



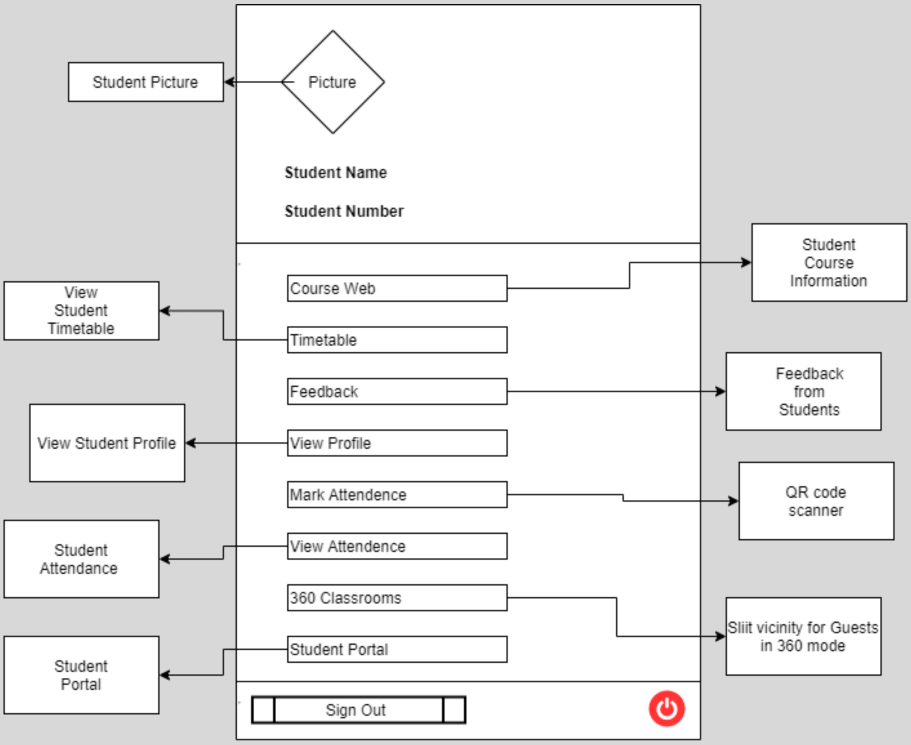
```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.VIBRATE" />
```

UI/UX flow

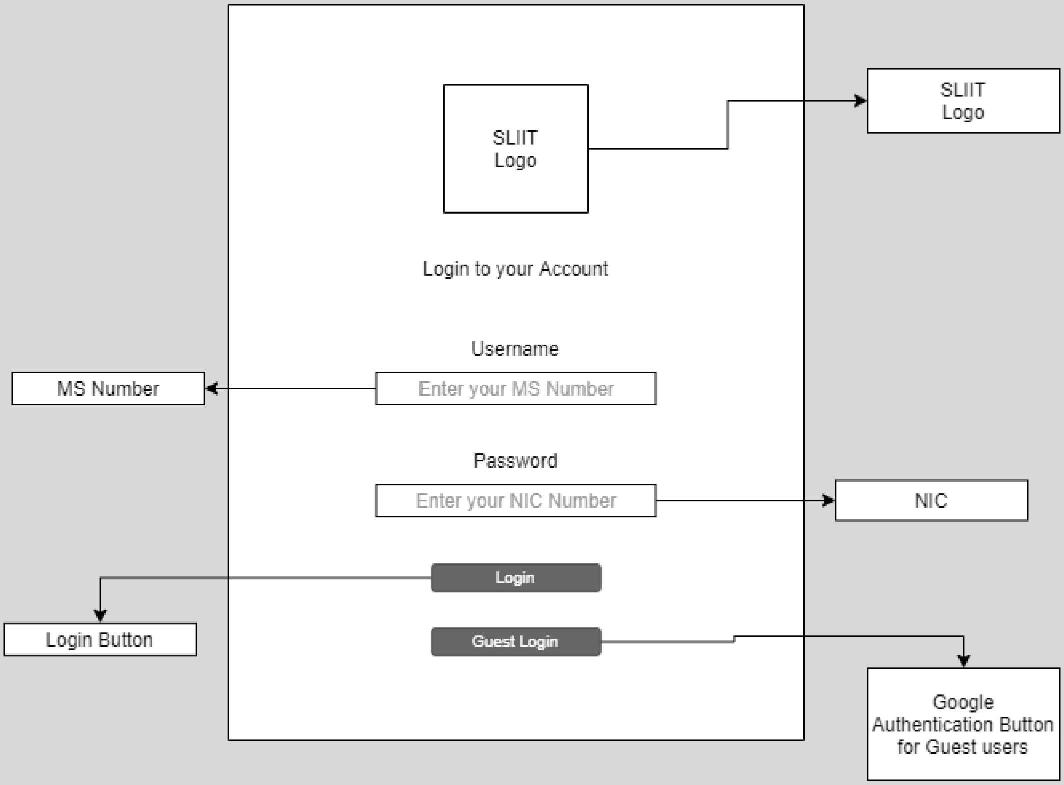
Wireframe 1 – Main User Interface



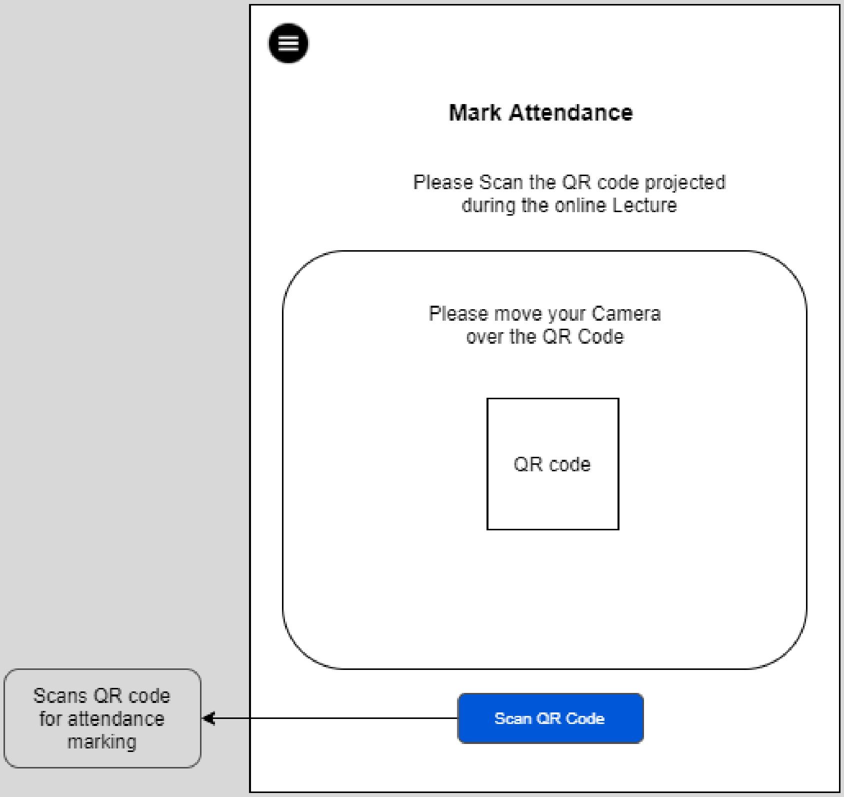
Wireframe 2 – Navigation Pane



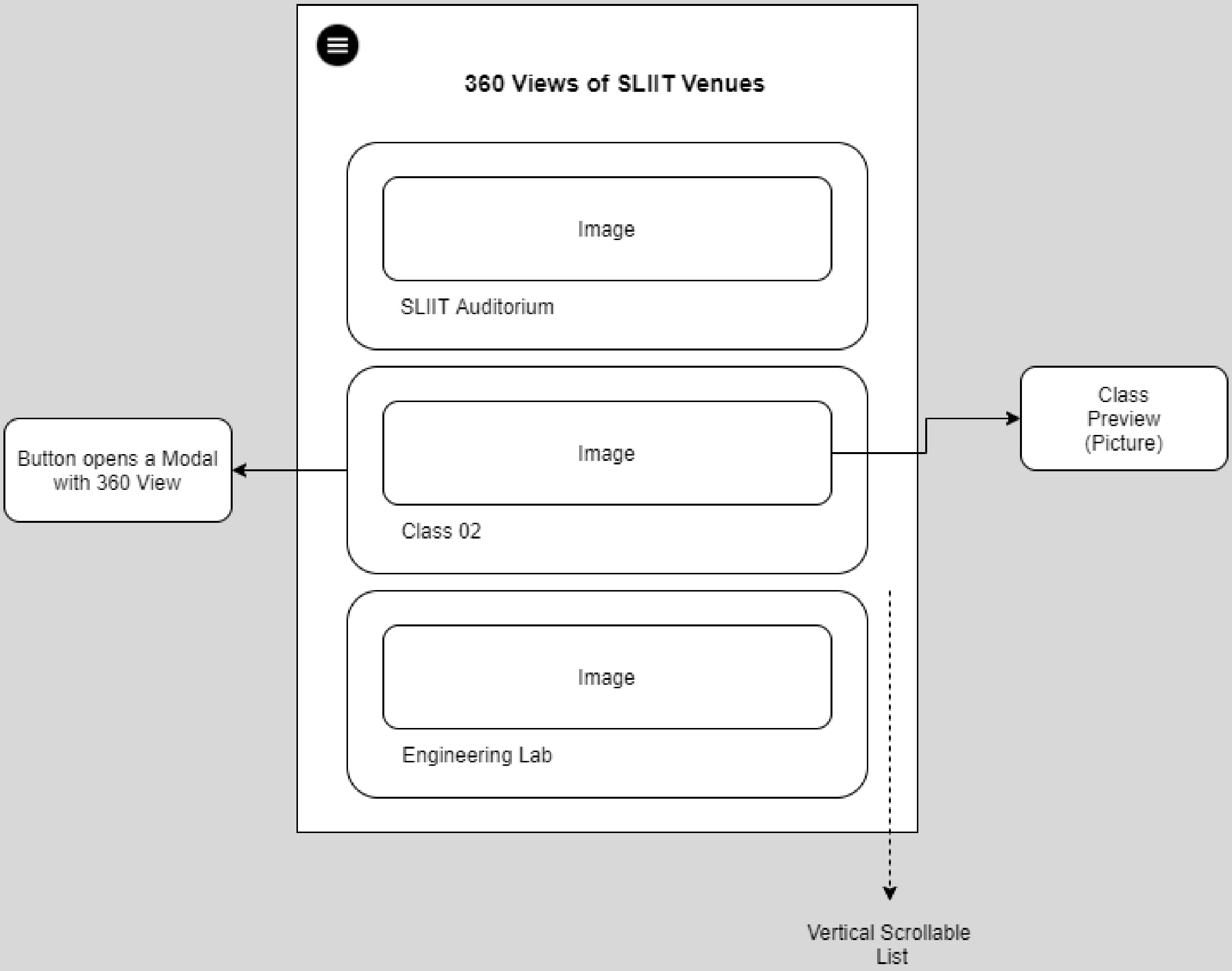
Wireframe 3 – Login Interface



Wireframe 4 – QR Scanner Interface


Mark Attendance
Please Scan the QR code projected during the online Lecture
Please move your Camera over the QR Code
QR code
Scan QR Code
Scans QR code for attendance marking

Wireframe 5 – 360 Guest View



Hamburger menu is considered as an Anti-pattern, nevertheless for this application purpose a hamburger menu navigation seemed fit. Students might find it frustrating to swipe left and right continuously to navigate to a certain page. However, for the 360 Classroom view for the guest, a vertical scrollable view was proposed to enhance user experience.

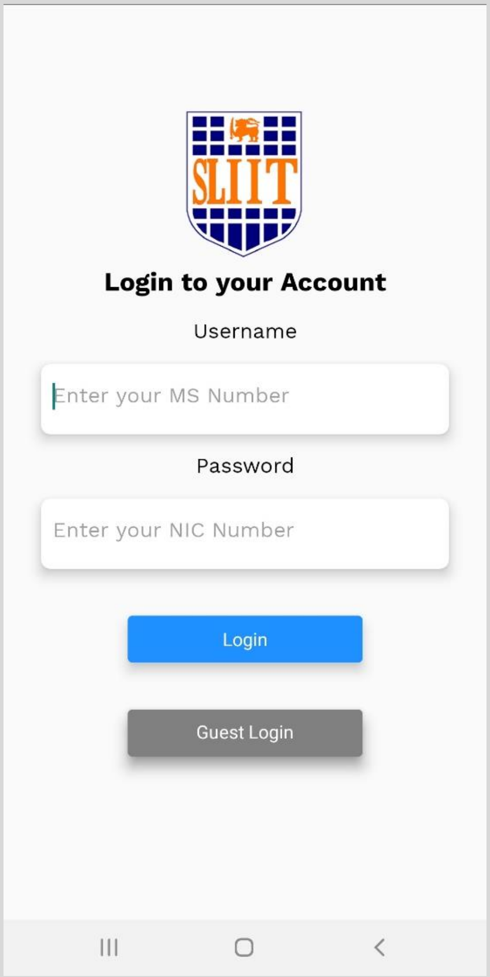
All interfaces have eye soothing colors that were coordinated with international color pallets, since the major target audience for this application is SLIIT students, some of SLIIT color themes were integrated as well. Cool animations, icons, sound effects, vibrations that follow convention were integrated so the UX is not frustrating. User can customize the application as they wish such as the light/dark theme, this would make the student feel like they are engaging with the application in a customized preference.

User Interface Libraries

MaterialIcons library was used to access the icons listed on https://oblador.github.io/react-native-vector-icons/, some icons/images that were not available could be found under the res/ folder. Conventional design icons used universally have been used in this application. App Icon badge was designed with AndroidAssetSets [3].

User Interface Flow

SLIIT student logs into SLIITLIFY Masters using their MS number and NIC. Guest users who do not possess such credentials, could use the Guest Login which is validated with Firebase Google Auth.



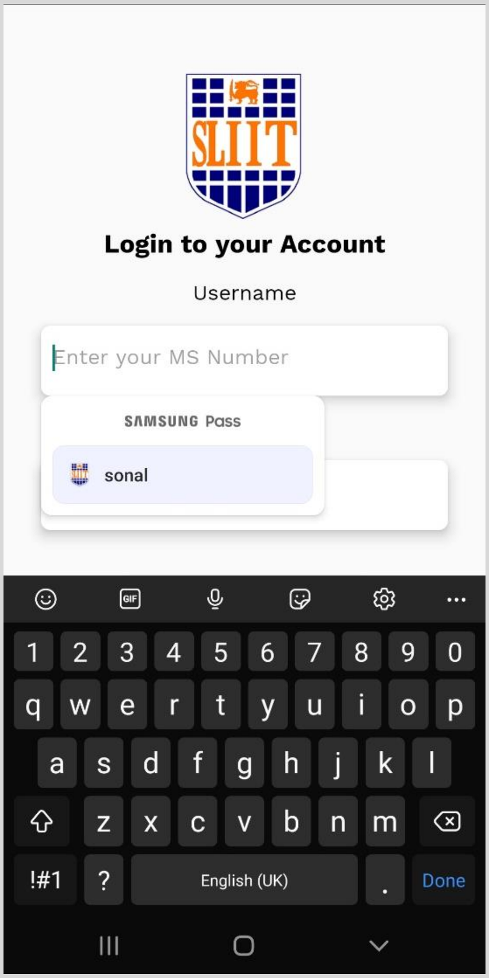
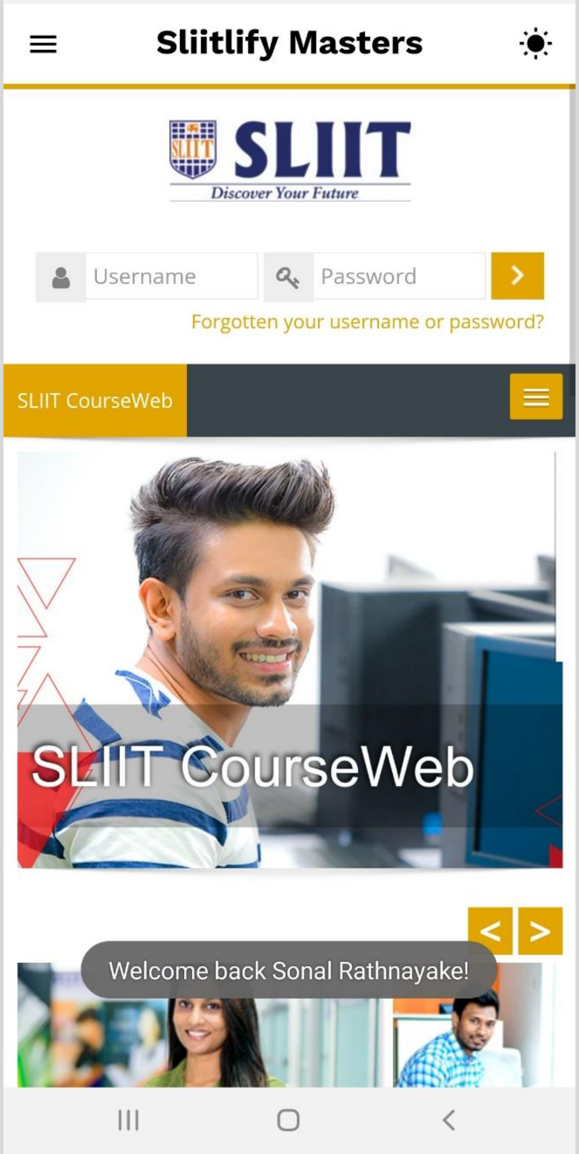


Figure 1 – User Log

Figure 2 - Login Supports Samsung Pass

Upon successful user login, student will be redirected to the Homepage, which is the courseweb itself.



Welcome back Sonal Rathnayake!

Figure 3.1 – Home Page (Light mode)

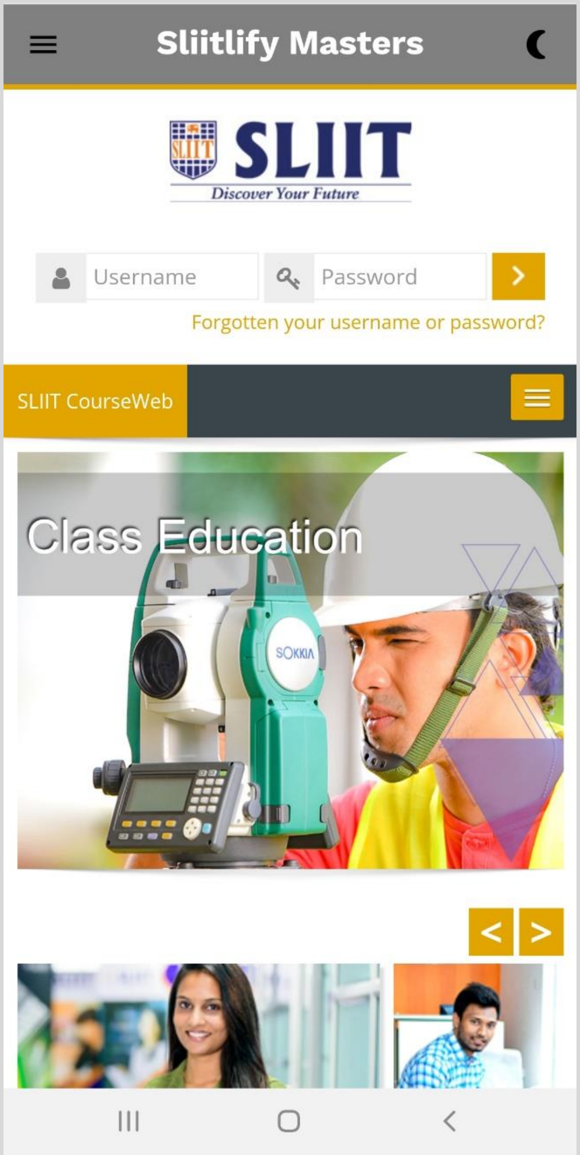
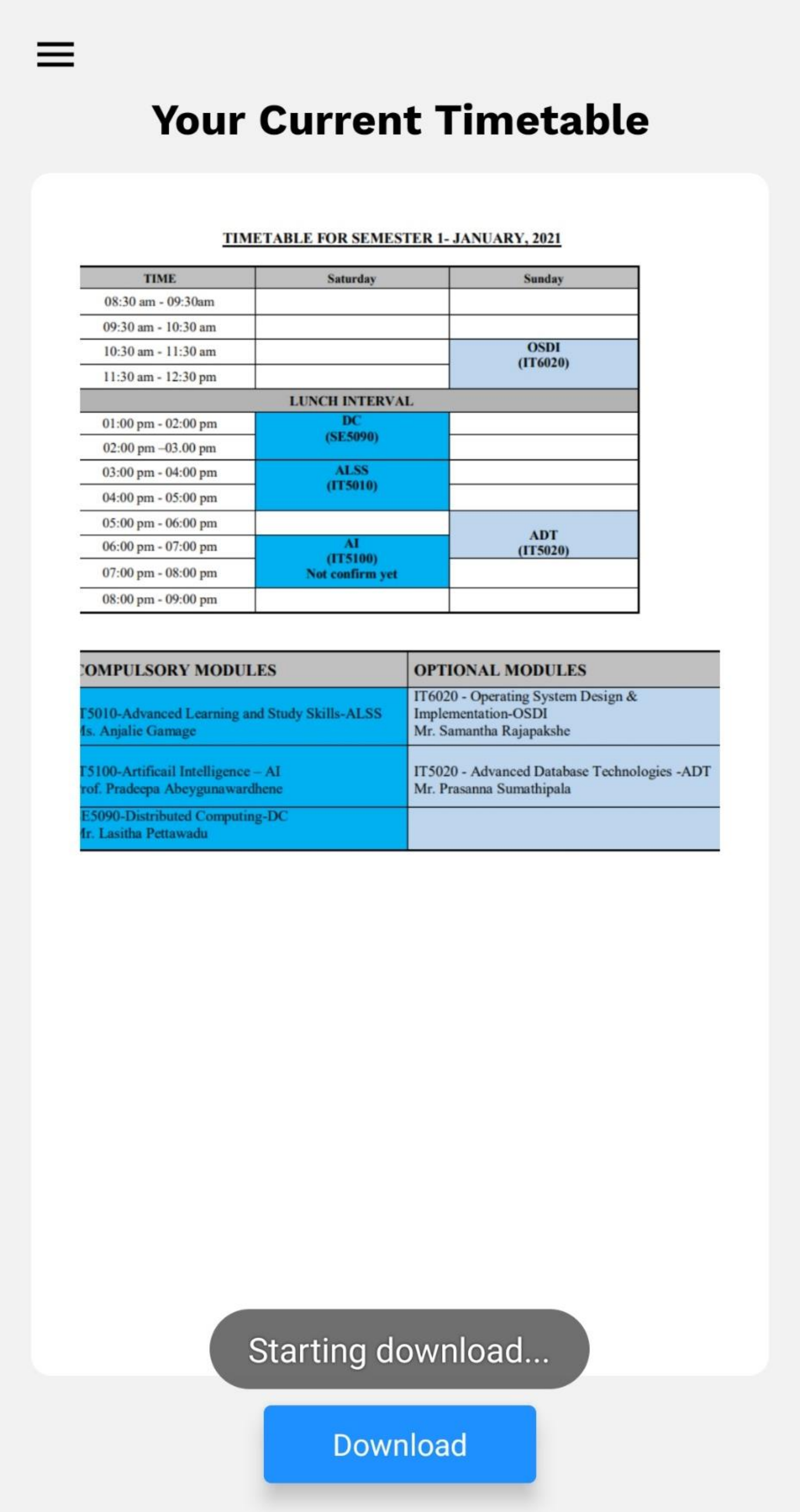


Figure 3.2 – Home Page (Dark mode)

Students can view their timetable for the corresponding Year/Semester. Downloadable PDF version will be stored on Firebase storage, so the student can refer offline if required.



Your Current Timetable

TIMETABLE FOR SEMESTER 1- JANUARY, 2021

TIME	Saturday	Sunday
08:30 am - 09:30am		
09:30 am - 10:30 am		
10:30 am - 11:30 am		OSDI (IT6020)
11:30 am - 12:30 pm		
LUNCH INTERVAL		
01:00 pm - 02:00 pm	DC (SE5090)	
02:00 pm –03.00 pm		
03:00 pm - 04:00 pm	ALSS (IT5010)	
04:00 pm - 05:00 pm		
05:00 pm - 06:00 pm		ADT (IT5020)
06:00 pm - 07:00 pm	AI (IT5100) Not confirm yet	
07:00 pm - 08:00 pm		
08:00 pm - 09:00 pm		

COMPULSORY MODULES	OPTIONAL MODULES
T5010-Advanced Learning and Study Skills-ALSS Ms. Anjalie Gamage	IT6020 - Operating System Design & Implementation-OSDI Mr. Samantha Rajapakshe
T5100-Artificail Intelligence – AI rof. Pradeepa Abeygunawardhene	IT5020 - Advanced Database Technologies -ADT Mr. Prasanna Sumathipala
E5090-Distributed Computing-DC Mr. Lasitha Pettawadu	

Starting download...

Figure 4 – View Timetable UI

When the home screen is swiped right, a Navigator menu slides in from the left-hand corner.

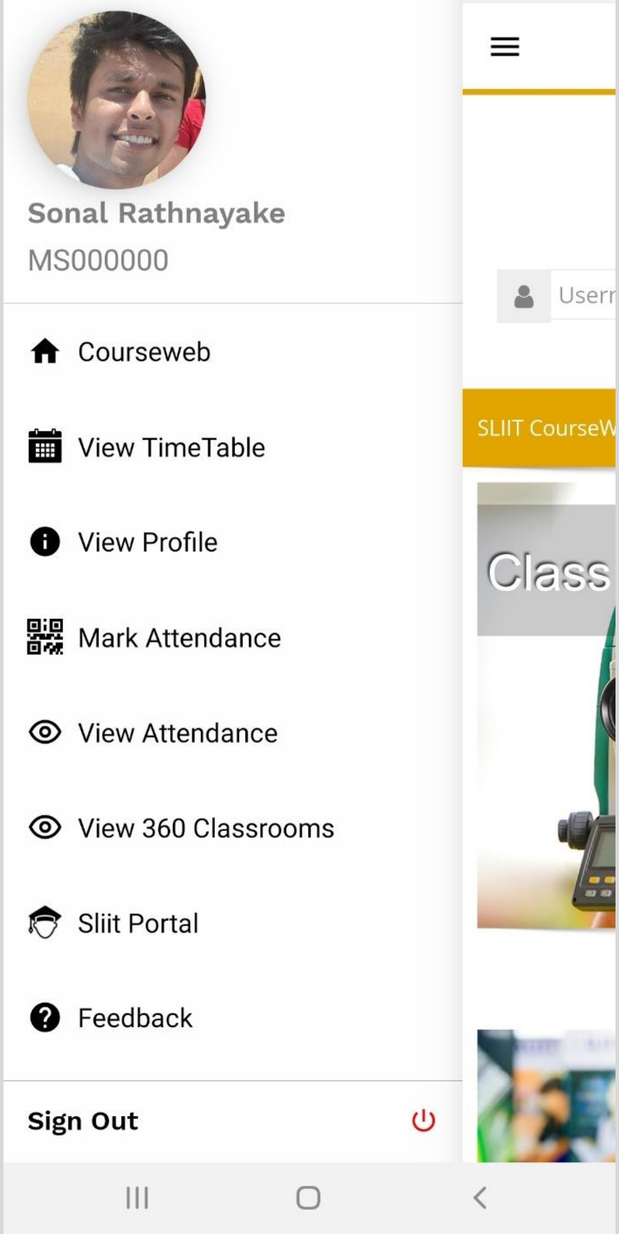
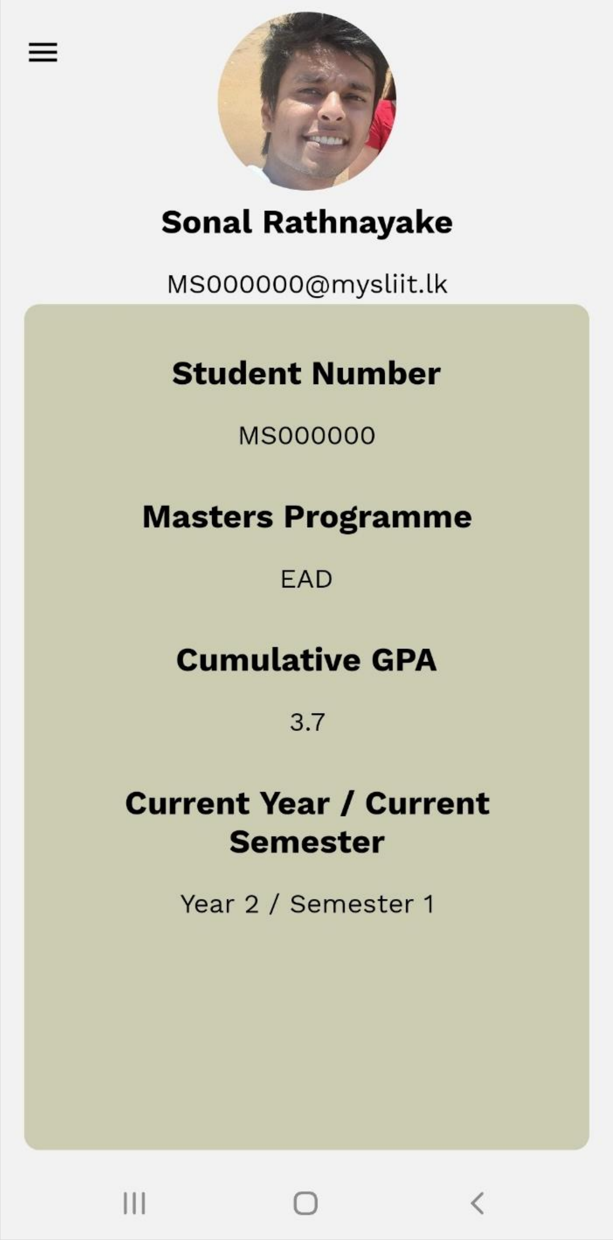


Figure 5 – Menu Navigator



Sonal Rathnayake

MS000000@mysliit.lk

Student Number

MS000000

Masters Programme

EAD

Cumulative GPA

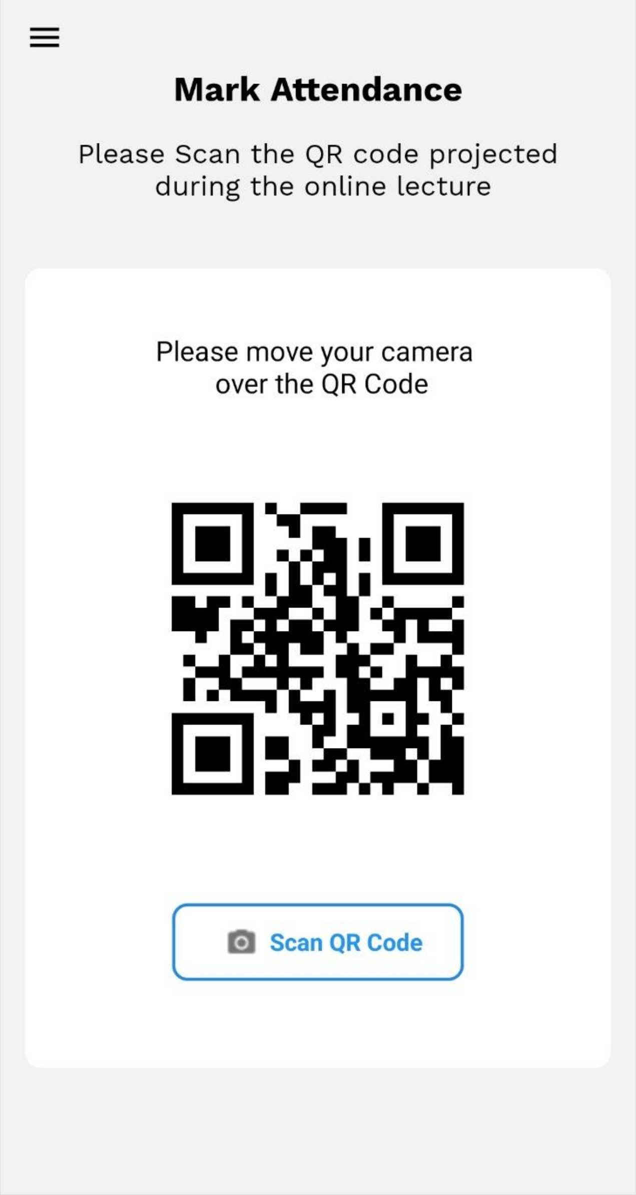
3.7

Current Year / Current Semester

Year 2 / Semester 1

Figure 6 – Student Profile UI

Students can open “Mark Attendance” Menu to record attendance when the respective lecturer publishes the QR code during the lecture.

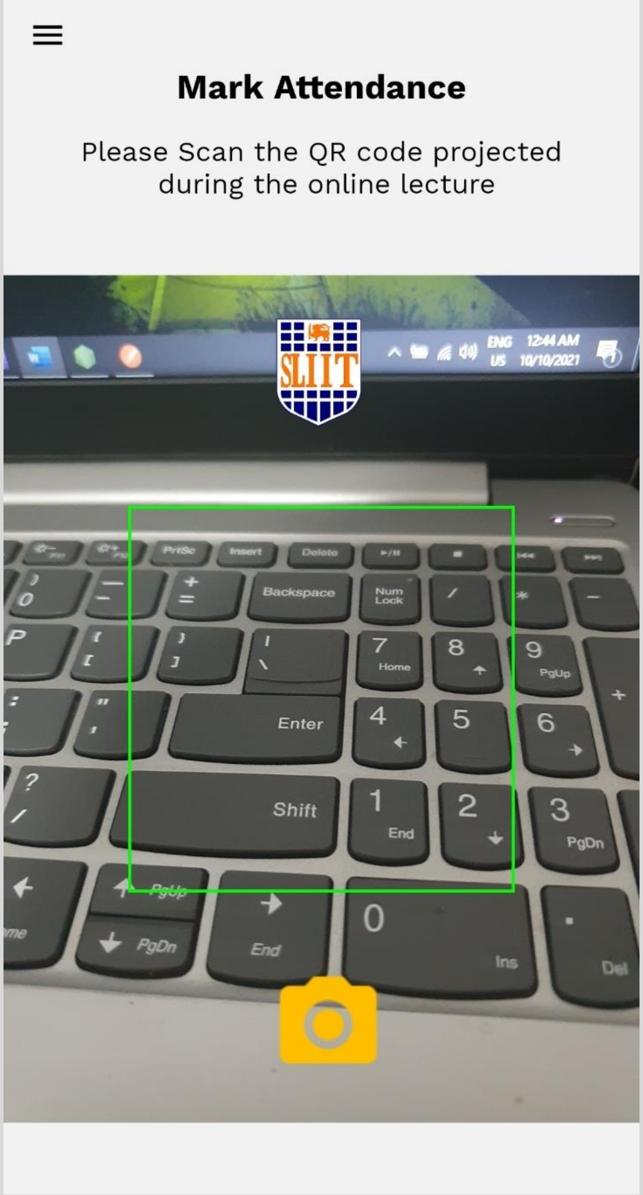


Mark Attendance

Please Scan the QR code projected during the online lecture

Please move your camera over the QR Code

Figure 7 – Mark Attendance UI

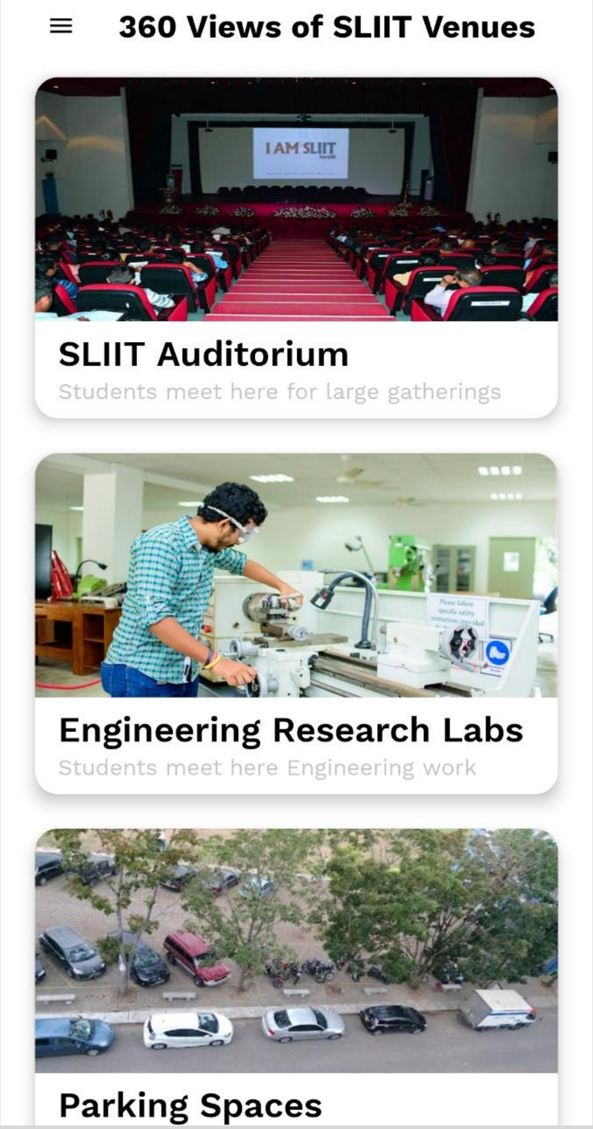


Mark Attendance

Please Scan the QR code projected during the online lecture

Figure 8 – QR Scanner UI

When Guests login to the system, guests can refer SLIIT Portal, Courseweb, general information about master's degrees offered at SLIIT and view 360 classroom images. These menus will be set to visible true, whilst the student menus will be invisible for guests. 360 Views of SLIIT Venues displays an embossed list of images, that the guest can click on to open a 360-degree view of the vicinity.



≡
360 Views of SLIIT Venues
SLIIT Auditorium
Students meet here for large gatherings
Engineering Research Labs
Students meet here Engineering work
Parking Spaces

Figure 9 – 360 View of SLIIT Venues

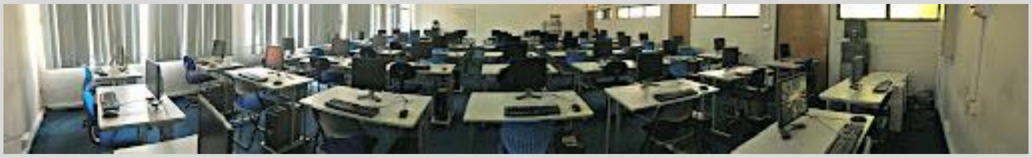


Figure 10 – 360-degree view of SLIIT Computer Lab [1]

Send feedback option is available for students to directly send message to the Lecturer, which will be stored in the backend with the respective student's message. Students are allowed to view student portal to view their results/academic information or pending payments.

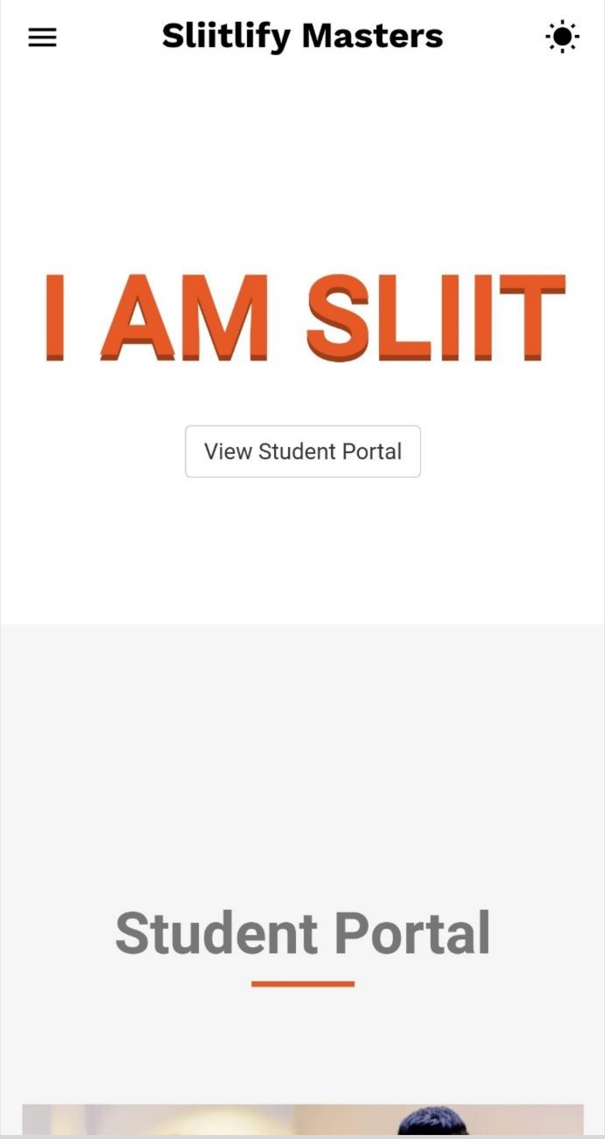


Figure 11 – Student Portal UI

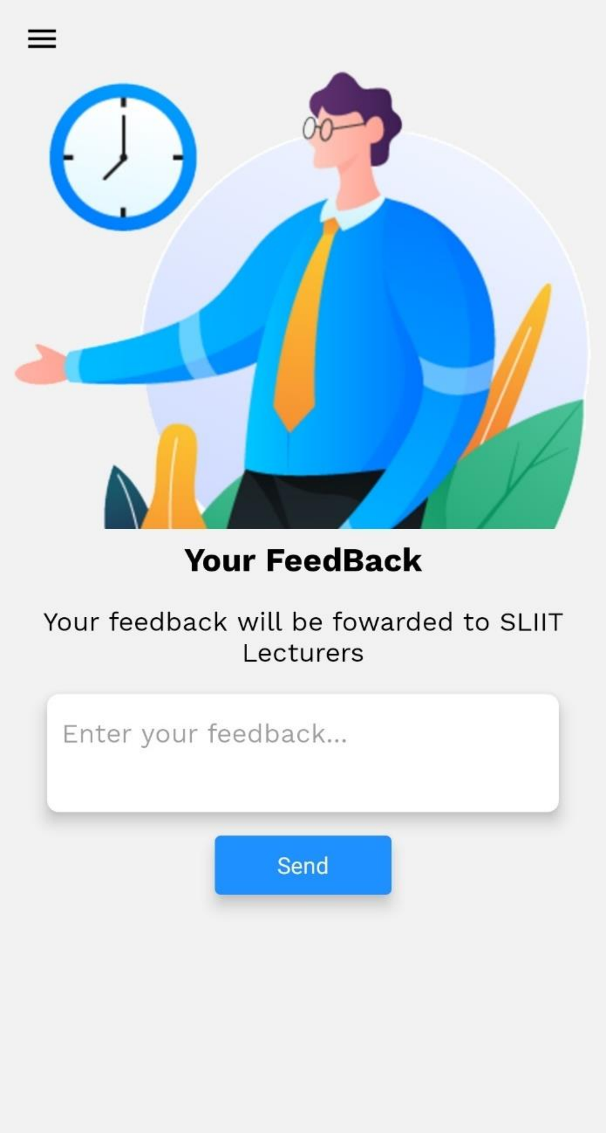
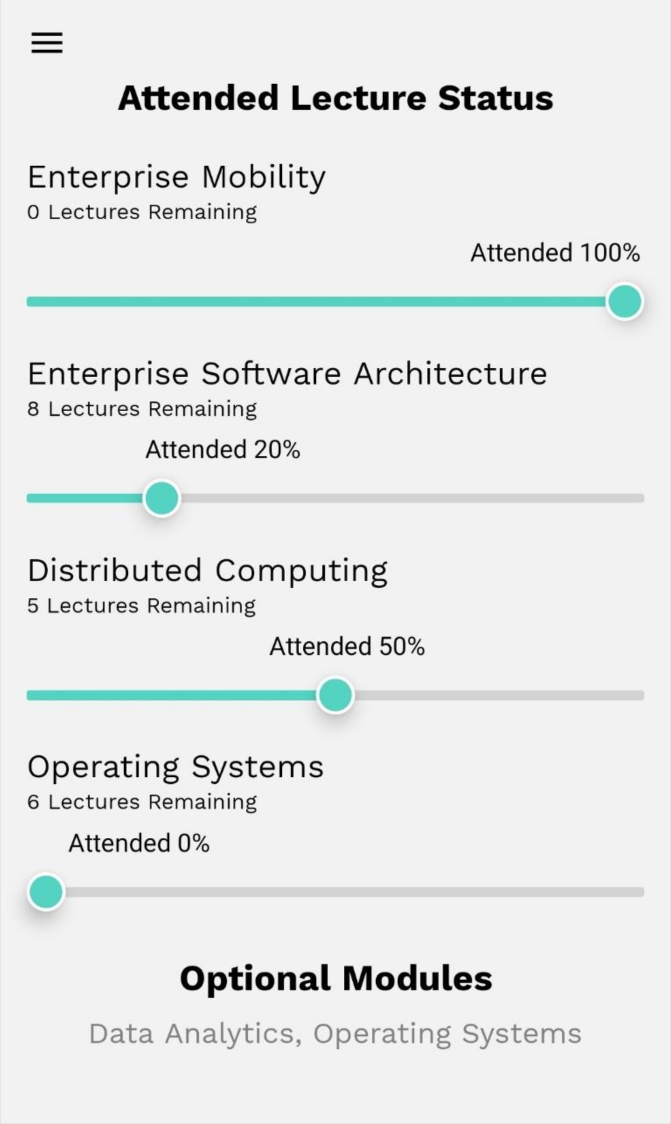


Figure 12 – Student Feedback UI

Student can also navigate to the “View Attendance” Menu to view all the current semester modules. This UI displays the student’s total attendance for the respective module as a percentage mark. To make things interesting, this is graphically aided with bar lines with the actual % retried from the database. Furthermore, the student can check the remaining attendance for each corresponding module and view the selected Optional modules for the enrolled semester.



Attended Lecture Status

Enterprise Mobility
0 Lectures Remaining
Attended 100%

Enterprise Software Architecture
8 Lectures Remaining
Attended 20%

Distributed Computing
5 Lectures Remaining
Attended 50%

Operating Systems
6 Lectures Remaining
Attended 0%

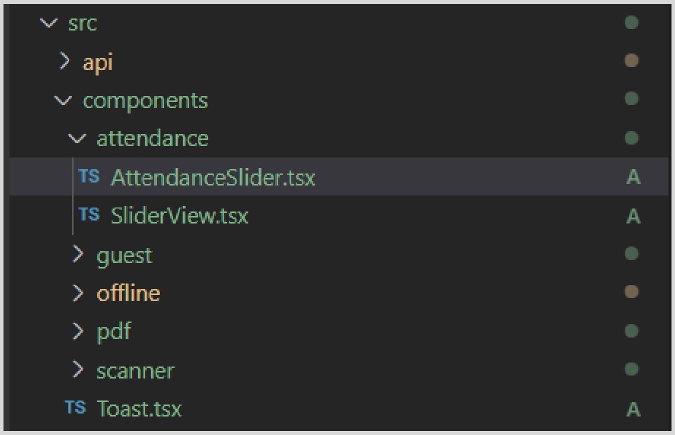
Optional Modules

Data Analytics, Operating Systems

Figure 13 – View Attendance UI

Usage of custom component(s)

Among several custom components, a custom component built from scratch is attendance.



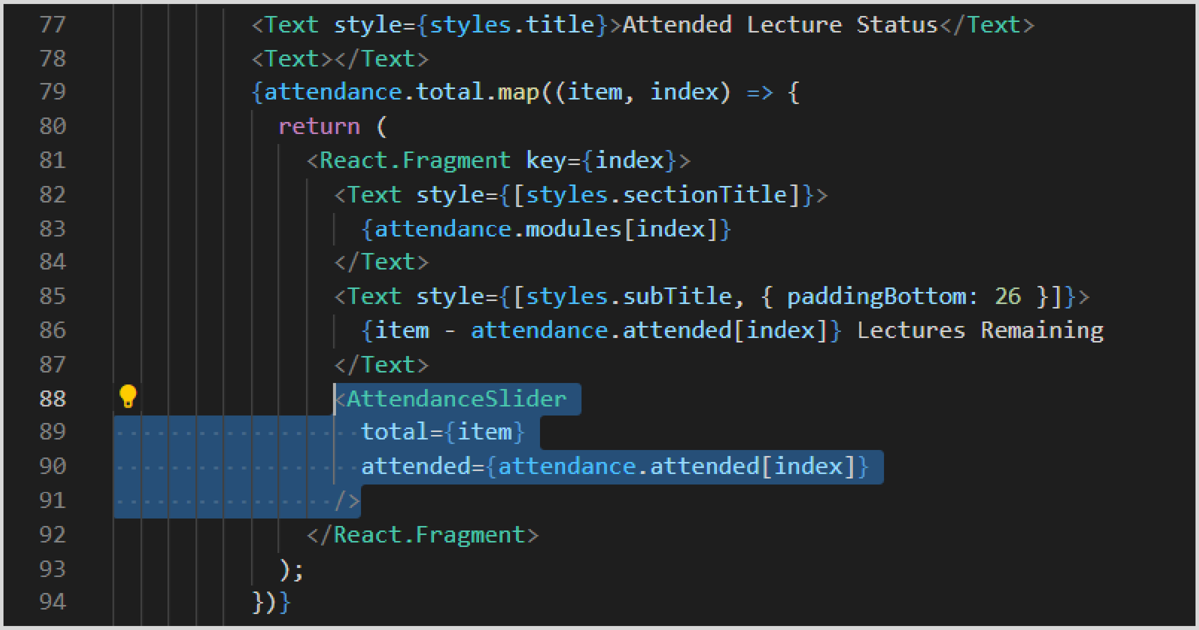
```
src
  api
  components
    attendance
      TS AttendanceSlider.tsx        A
      TS SliderView.tsx              A
    guest
    offline
    pdf
    scanner
    TS Toast.tsx                     A
```

<- Another completely designed from scratch custom component is Toast.tsx, which styles the given input text and displays a Toast message

The Attendance slider custom component takes 2 input props, which are

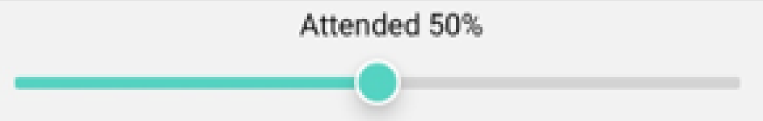
- Number of attended Lectures
- Number of total lectures

These two values are used to calculate the student's current attendance % and returns a slider bar with the corresponding values. Below shows the usage of this component



```
          <Text style={styles.title}>Attended Lecture Status</Text>
          <Text></Text>
          {attendance.total.map((item, index) => {
            return (
              <React.Fragment key={index}>
                <Text style={[styles.sectionTitle]}>
                  {attendance.modules[index]}
                </Text>
                <Text style={[styles.subTitle, { paddingBottom: 26 }]}>
                  {item - attendance.attended[index]} Lectures Remaining
                </Text>
                <AttendanceSlider
                  total={item}
                  attended={attendance.attended[index]}
                />
              </React.Fragment>
            );
          })}
```

Final output of **<AttendanceSlider/>** only displayed when **<OfflineApp/>** is validated



Further Prototype Enhancements

- Implementation of Guest Login using Google Authentication, to display Guest specific drawer components.
- Implementation of user sign out was not included in the current prototype. Currently, user state is maintained with react context and hooks. This need to be replaced with react redux in a future version update.
- Currently Firebase Functions requires a firebase blaze package, which is a paid service on usage. When scaling above 2 million requests per day, we need to consider other unpaid backend services, such as Prisma.
- Future major business logics that consume a lot of power and resources to be implemented directly as cloud functions on the backend server.
- Get official approval from SLIIT authorities for a production build release.

Challenges Faced

Some redundant features were implemented and removed later in Design prototyping. One good example is the usage of react-native-torch. This feature was available during the initial release, but accessing this hardware seemed redundant as the student would not be scanning QR codes in the dark. They would scan it off a laptop/computer screen which would cause a glare affect. Since this can cause some frustration in UX, it was **removed** during the latter phases of development.

For knowledge sharing purposes and assignment criteria retrospective regarding the learning curve was discussed among 2 students, Dinuka Kodituwakku (MS21911644) and Randika Peiris (MS21911712)

References

[1] http://www.cse.mrt.ac.lk/life/resources_and_facilities

[2] Images used for application purposes are purely from https://www.sliit.lk/ and courseweb. Image courtesy credits goes to original owners.

[3] https://romannurik.github.io/AndroidAssetStudio/icons-launcher.html

[4] https://medium.com/@savinihemachandra/creating-rest-api-using-express-on-cloud-functions-for-firebase-53f33f22979c