PSG COLLEGE OF TECHNOLOGY

(Autonomous Institution)

COIMBATORE – 641 004

CLAPP

The Classroom App

SUKLESH S RAO (16Z355)

HAREESHWAR K (17Z310)

MAJELLA PHILO KISHORE S (17Z316)

PAVITHRAN P (17Z327)

TEJAS H BADANI (17Z334)

MOHAMEDSHUAIB S (18Z470)

Dissertation submitted in partial fulfilment of the requirements for the degree of

BACHELOR OF ENGINEERING

Branch: COMPUTER SCIENCE ENGINEERING

of Anna University

May 2019

.....

Dr. K. Sathyapriya

Faculty guide

Certified that the candidate was examined in the viva-voce examination held on

…...……

……..
……..
(Internal Examiner)
(External Examiner)

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

CLAPP is a classroom management application which has different sections to cater to the different needs that arise in the classroom environment. One of its functionalities is to enable the users (the student community) to upload their notes and other academic materials to a cloud based database to serve during the times of need facilitating quick access.

It also includes a dynamic timetable section which notifies the users whenever there are any changes made. This application also lets the users to post articles regarding any upcoming events in a content sharing section.

Another feature of the application is that it contains the contact info that includes the mobile numbers and e-mail addresses of the students and also the faculty members to facilitate further communication.

And additionally as a future enhancement, it also has provisions to provide a chat section for student discussions. It pacifies the need for an environment that is capable of giving a small area underneath a user name to exchange data.

The primary motive of this application is to eliminate the dependence on social media platforms for communication and management related activities in the classroom environment and provide an exclusive platform for information sharing functionalities required for the classroom system.

TABLE OF CONTENTS

1. INTRODUCTION

The institutions around the world need an application to be developed for the classroom. Every user [who is authorised] should be able to create or join a classroom community online using a username and a password assigned to the classroom. A classroom must also have admins to perform the exclusive administrative functions.

Every classroom is also expected to have the following features :

- Subject wise document sharing capabilities
- Real-time timetable
- List of members in the classroom
- Faculty information
- Sharing option to share the username and password of the classroom so other users can join.

Only an admin should be able to make changes to the timetable. The admins must also be able to remove themselves as admins and appoint other members as admins. The admin alone should be allowed change the password of the classroom. The admin alone should be able to remove members from the classroom.
The faculty must be able to add details of themselves to display in the faculty info tab.

All these functions must be implemented in a smooth application with a user-friendly user interface.

With a view to cater to all these needs, the Clapp application is developed for the mobile platform.

2. SOFTWARE REQUIREMENTS SPECIFICATION

2.1. Introduction

2.1.1. Purpose

In colleges, there exists a dependency on several social media apps for several classroom related tasks. This software application is designed with the purpose of eliminating the need for several such applications by providing all the necessary features required, as a single package.

2.1.2. Document Conventions

The following conventions have been followed in this document for representative purposes.

DB = Database
ER = Entity Relationship
FRU = Functional Requirement: User Registration
FRC = Functional Requirement: Class
FRP = Functional Requirement: Profile Settings
FRD = Functional Requirement: Document Uploads
FRO = Functional Requirement: Others
GUI = Graphical User Interface

2.1.3. Intended Audience and Reading Suggestions

This Software Requirements Document is intended for:

- Developers who can review project's capabilities and more easily understand where their efforts should be targeted to improve or add more features to it (design and code the application - guidelines for future development).
- Project testers can use this document as a base for their testing strategy as some bugs are easier to find using a requirements document. This way testing becomes more methodically organized.

- End users of this application who wish to read about what this project can do.

2.1.4. Product Scope

This classroom application is hugely beneficial for faculty, administrators and students of the classroom. It provides improved visibility and content distribution with analytics and better communication channels. Smart content, enhanced communication, consolidation of data and improved efficiency are all inclusive in the scope of the implementation of this software.

2.1.5. References

Documentations from

- https://www.postgresql.org/
- https://docs.flutter.io/

Courses

- https://www.udemy.com/learn-flutter-dart-to-build-ios-android-apps/

2.2. Overall Description

2.2.1. Product Perspective

The classroom application provides an easy interface for classmates for uploading documents, content sharing and keeps a chat interface separate from the sharing interface making it easy to search and find documents. There is an in-built timetable feature which can be viewed by anyone and can be edited by the faculty and the class admin. The application provides separate classrooms making sure that each classroom has an exclusive interface for its own content.

2.2.2. Product Functions

In the classroom application the admin can create a classroom, add classmates and remove existing participants if needed. The faculty on the other hand has the access privileges to create timetables, edit timetables and send materials to the classes she/he is a part of. First time users will be able link their google account with the app and afterwards can get into classes created by the admins. There are two separate interfaces after joining a classroom namely, the content sharing interface and the documents interface both of which can be accessed by a swipe to the right or the left respectively. Sharing will be the most integral part of the app which will work seamlessly and will have a clear and simple interface making it easy for students to find the files they need easily.

2.2.3. User Classes and Characteristics

The app is aimed mostly at college students and so in accordance 16 years of age would be recommended. The user does not need any specific knowledge of computer science although basic knowledge about app navigation would be helpful. The app is going to be primitively in English language. So basic English knowledge is enough for the user to understand the usage of the app.

2.2.4. Operating Environment

Hardware Requirements:

- 512 MB RAM (Minimum)
- 1 MB Cache Memory
- 100 MB of free space

Software Requirements:

- Operating System: Android, iOS

Programming languages used: Dart
Back-End setup using: PostgreSql

2.2.5. Design and Implementation Constraints

- The software must be supported on all devices that satisfy the hardware and software requirements.
- The application must be adaptive to various dimensional and processing power constraints of the devices.
- The application's execution must be fast and must support a large user base as well.
- The application's functioning must be efficient and must not increasingly affect the battery life of the end user's device.

2.2.6. User Documentation

The classroom application does not provide any manual documentation. A tutorial for new users will be included in the application itself. Any further queries will be clarified via emails through the support email-id for the application.

2.2.7. Assumptions and Dependencies

- The system set up as the backend server will be active at all times.
- Processing power of the server will be able to accept many multiple data access requests.
- So the system will not crash even with multiple users uploading and downloading files at the same time.
- User has a valid google account
- Availability of internet to the users
- User has knowledge of English language and mobile application usage.

- The user is morally responsible and understands the ethics of the software usage.

2.3. External Interface Requirements

2.3.1. User Interfaces

The application will provide a user friendly and navigation based interface.
The following screens will be provided in the app's UI.

- A login screen for user authentication using a sign in feature.
- A registration screen to join a classroom by entering the unique id and password.
- A file sharing section to enable sharing of resources among students. Here the documents can be uploaded to organised sections
- A content sharing area for sharing of articles, news and other announcements.

2.3.2. Hardware Interfaces

N/A

2.3.3. Software Interfaces

- Flutter and Dart API – To export the project as android and iOS packages.
- PostgreSql - Backend
- Camera Interface – An interface to access the device's camera inside the application.
- Google Sign In Interface – Google's App Signing Interface to authenticate the users.

2.3.4. Communications Interfaces

- TCP/IP – For content sharing

2.4. System Features

This application will incorporate the following modules having the specified many features.

2.4.1. Module 1

Id : User Sign In (FRU1)
Dependency : None
Description :
The user has to log in through google sign in order to obtain access to the app's features.
A unique id is created for the user on his/her sign in. A terms and conditions document for the app usage will also be provided in this screen and the user has to comply with these terms in order to gain access.
Access Privileges : All users

2.4.2. Module 2

Id : Profile Selection (FRP1)
Dependency : FRU1
Description :
The user must be able to choose between the profile choices of a student or a faculty in the classroom. The users can also provide their profile details such as name, contact info and mail id.
Access Privileges : All users

2.4.3. Module 3

Id : Creating a new class (FRC1)
Dependency : FRU1
Description :
A user can opt to create a new class after logging into the app. The class must be having a unique id and a password. The input fields for these values must be validated before entry into the database. The user who creates the class will serve as the admin by default.
Access Privileges : All users

2.4.4. Module 4

Id : : Joining a Class (FRU2)
Dependency : FRU1,FRC1
Description :
Every class will have a unique id and a specific password. The user has to enter the appropriate credentials to join the classroom. The input fields must be validated for the correct format and values.
Access Privileges : All users

2.4.5. Module 5

Id : Viewing Time Table (GUI1)
Dependency : FRU2
Description:
Every classroom will have a separate timetable section which the members of the class can view anytime.
Access Privileges: All users

2.4.6. Module 6

Id : Documents Download (FRD1)
Dependency : FRU2
Description:
The users can upload documents such as notes to different sections in the sharing area. These uploaded documents must be

ordered in an organized fashion and must be visible and downloadable to all the members of the class.
Access Privileges: All users

2.4.7. Module 7

Id: Contact Info Section (GUI3)
Dependency : FRU2
Description :
To display the contact info that includes mobile numbers and e-mail addresses of the students and the faculty members to facilitate further communication.
Access Privileges : All users

2.4.8. Module 8

Id : Documents Upload (FRD2)
Dependency : FRU2
Description:
The users can upload documents such as notes to different sections in the sharing area. These uploaded documents must be ordered in an organized fashion and must be visible and downloadable to all the members of the class.
Access Privileges: All users

2.4.9. Module 9

Id : Remove Member (FRC2)
Dependency : FRU2
Description:
The Administrator of the class must be allowed to remove any member of the class.
Access Privileges: Admins

2.4.10. Module 10

Id : Change Password (FRC3)
Dependency : FRU2

Description:

The Administrator of the class must be also be allowed to change the password of the class.

Access Privileges: Admins

2.4.11. Module 11

Id : Appointing Admins for the class (FRC4)

Dependency : FRC1,FRU2

An admin can be selected for the classroom with a maximum of two admins per class.

Description:

Access Privileges: Existing Admins

2.4.12. Module 12

Id : Drop out from being admin (FRP2)

Dependency : FRC2,FRP1

Existing admins can select another participant as the admin of the class and then drop out from being the admin.

Description:

Access Privileges: Existing Admins

Access Privileges: All users

2.4.13. Module 13

Id : Modifying Time Table (GUI2)

Dependency : FRU2,GUI1,FRC2

Description:

The timetable of the classroom can be modified to incorporate changes in the schedule.

Access Privileges: Admins of the class.

2.4.14. Module 14

Id : Document Deletion (FRD3)
Dependency : FRD2
Description:
The uploaded documents can be deleted when not required anymore.
Access Privileges: Admins

2.5. Other Nonfunctional Requirements

2.5.1. Performance Requirements

- **Scalability** – The system should be scalable to support a large user base.
- **Speed** - The application usage experience should be smooth and fast. Increase in the number of users should not affect the speed of the system. Search and loading functionalities should be fast facilitating a better end-user experience. The system's response time should also be very short for the user's data access requests.

2.5.2. Safety Requirements

N/A

2.5.3. Security Requirements

- User Authentication - A sign in feature to authenticate the user to allow access to the software.
- Unique Identification - A unique domain identifier for each classroom

- Password - A password stored as hash value in the database for each classroom domain.
- Admins - Each domain or classroom needs to have two users with administrator access. The admins must be able to modify the data of the classroom and must also be able to remove the existing users of the domain.
- User Categories – The user should be differentiated as a student or a faculty for a particular class.

2.5.4. Software Quality Attributes

- Usability - An easy to understand and simple user interface for a smooth usage experience.
- Availability - Availability of the system at any time to use the software.
- Maintainability - Extensibility of the software for features in the future and efficient maintenance

2.5.5. Business Rules

This project is a non-profitable undertaking and will continue to be free for public usage until any business models are proposed for this project. In case of the future versions of the application being modelled with the motive of monetary gains, the application will be available as a freemium model with certain features being offered exclusively for the premium users.

2.6. Other Requirements

Other additional dependencies
- Database Plans
- Privacy Policies for legal release

Appendix A: Glossary

- App – Application
- API – Application Programming Interface
- UI – User Interface

Appendix B: Analysis Models

N/A

Appendix C: To Be Determined List

- Chat functionality features and constraints

3. SYSTEM DESIGN

CLASS DIAGRAM
DATA FLOW DIAGRAM
TABLES
DB SCHEMA

4. IMPLEMENTATION AND RESULTS

IMPLEMENTATION (PSEUDO-CODE):

MODULE 1 : USER SIGN IN

```
import 'firebase_core.pkg'
import 'googlesignin.pkg'
import 'homescreen'
import 'profileselection'
main()
{
        // SIGN IN MODULE
        FirebaseAuth auth = FirebaseAuth.getinstance();
        FirebaseUser user = auth.user;
        if(user.isloggedin == true)
        {
                if (user.profile == null || user.name==null || user.contact == null)
                        Navigate_to(profileselection);
                else
                        Navigate_to(homecreen);
        }
        else
                googlesignin();
        return;
}
```

MODULE 2 : PROFILE SELECTION

```
import 'firebase_core'.pkg
import 'postgres.pkg'
main()
{
        // PROFILE MODULE
        FirebaseAuth auth = FirebaseAuth.getinstance();
        FirebaseUser user = auth.user;
        if(user.profile==null)
        {
                string profile;
                int invalid = 1;
                while(invalid == 1)
                {
                        display('Select Profile choice: Student or Faculty ');
                        read(profile);
                        if(profile != teacher && profile!= student)
```

```
                        display('Invalid choice');
                else
                        invalid=0;
        }
        user.profile = this.profile;
    }
    if(user.name == null)
    {
        String name;
        display('Enter name');
        read(name);
        user.name = this.name;
    }
    if(user.contact == null)
    {
        Stirng contact;
        display('Enter contact');
        read(contact);
        user.contact = this.contact;
    }
    PostgresConnection postgresConnection = new PostgresConnection();
    postgresConnection.write(user);
    return;
}
```

MODULE 3 : CREATING A NEW CLASS
MODULE 4 : JOINING A NEW CLASS

```
import 'postgres.pkg'
import 'firebase_core.pkg'
import 'classroom'
main()
{
    FirebaseAuth auth = FirebaseAuth.getinstance();
    FirebaseUser user = auth.user;
    PostgresConnection postgresConnection = new PostgresConnection();
    var results[][]  = postgresConnection.getdata(user);
    ListView list = ListView.build(results.classes);
    display(list);
    Menu menu = options(Create New Class,Join New Class,Logout);
    display(menu);
    int choice;
```

```
	onselect(Menu)
	{
		read(choice);
		if(choice==1)
		{
			// CREATING NEW CLASS MODULE
			display('Enter new Class Id , Password and Confirm Password');
			read(classid);
			read(password);
			read(confirmpassword);
			if(password!=confirmpassword){
				display('Passwords dont match');
			}
			else
			{
				postgresConnection.write(classid,password);
				postgresConnection.write(classid.user.isAdmin:1);
			}

		}
		if(choice==2)
		{
			// JOINING A CLASS MODULE
			display('Enter Class id and Password');
			read(classid);
			read(password);
			if(!postgresConnection.data(classid).exists)
				display('Invalid classid');
			else if(postgresConnection.data(classid).password!
=this.password)
				display('Invalid password');
			else{
				list.add(classid);
				postgresConnection.write(classid.user.isAdmin:0);

			}
		}
		if(choice==3)
		{
			logout();
		}
	}
	onselect(list)
```

```
        {
                Navigate_to(classroom,index:list.selectedindex);  //selected class id
        }
        return;
}
```

MODULE 5 : VIEWING TIME TABLE
MODULE 6 : DOCUMENTS DOWNLOAD

```
import 'postgres.pkg'
import 'firebase_core.pkg'
import 'options'
main()
{

        PostgresConnection postgresConnection = new PostgresConnection();
        String classid = classes[index];
        var results[][]  = postgresConnection.getdata(classid);
        Menu menu = options(classoptions);
        display(menu);
        Tabs tab = new Tabs(TimeTable,Documents);
        display(tab);
        int choice;
        onselect(tab)
        {
                // VIEWING TIMETABLE MODULE
                read(choice);
                if(choice == 1)
                {
                        date _date;
                        display('Select Date');
                        read(date);
                        ListView timetable = ListView.build(results.timetable[date]);
                        display(timetable);
                }
                // DOCUMENTS DOWNLOAD MODULE
                else if(choice == 2)
```

```
		{
			ListView subjects = ListView.build(results.subjects)
			display(subjects);
			onselect(subjects)
			{
				ListView documents =
ListView.build(data.subjects[selectedindex].documents);
				display(documents);
				onselect(documents)
				{
					download(documents[selectedindex].link);
				}
			}
		}

	}
	onselect(menu)
	{
		Navigate_to(options);
	}
	return;
}
```

MODULE 7 : VIEW CONTACT INFO
MODULE 8 : DOCUMENTS UPLOAD

```
import 'postgres.pkg'
import 'firebase_core.pkg'
import 'adminfunctions'
import 'filepicker'

main()
{
	PostgresConnection postgresConnection = new PostgresConnection();
	var data = postgresConnection.getdata(classid);
	if(user.isadmin == true)
	{
		display(settings);
		onselect(settings)
		{
			Navigate_to(adminfunctions);
		}
```

```
        }

        var subjects = data.subjects;
        int i = subjects.length;


        ListView optionslist = ListView.build(View Member Info, View Faculty Info,
Add Subject, Upload Documents, Invite to  class);
        onselect(optionslist)
        {
                // VIEW CONTACT INFO MODULE
                if(selectedindex == 1)
                {
                        var membersdata = data.members;
                        ListView mlist = ListView.build(membersdata);
                        onselect(mlist)
                        {

display(membersdata[selectedindex].name,membersdata[selectedindex].contact,
membersdata[selectedindex].mail);
                        }}
                else if(selectedindex == 2)
                {
                        var facultydata = data.faculties;
                        ListView flist = ListView.build(facultydata);
                        onselect(flist)
                        {

display(facultydata[selectedindex].name,facultydata[selectedindex].contact,faculty
data[selectedindex].mail);
                        }
                }

                // UPLOAD DOCUMENTS MODULE
                else if(selectedindex == 3)
                {
                        read(subjects[i]);
                        i++;
                        postgresConnection.write(classid.subjects:subjects);
                }
                else if(selectedindex == 4)
                {
```

```
                        if(subjects==null)
                        {
                                read(subjects[i]);
                                i++;
                                postgresConnection.write(classid.subjects:subjects);
                        }
                        Menu submenu = options(subjects);
                        var selectedsub;
                        onselect(submenu)
                        {
                                selectedsub = subjects[selectedindex];
                                file url = filepicker();
                                FirebaseStorage storage =
FirebaseStorage.getinstance;
                                String downloadurl =
storage.Upload(file).getdownloadurl();;

postgresConnection.write(classid.subjects[selectedindex].documents[i]:download
url);}

                }
                else if(selectedindex == 5)
                {
                        share(classid,password);
                }
        }
        return; }
```

MODULE 9 : REMOVE MEMBER
MODULE 10 : CHANGE PASSWORD
MODULE 11 : APPOINTING ADMINS
MODULE 12 : DROPOUT FROM BEING ADMIN
MODULE 13 : MODIFYING TIME TABLE
MODULE 14 : DOCUMENTS DELETION

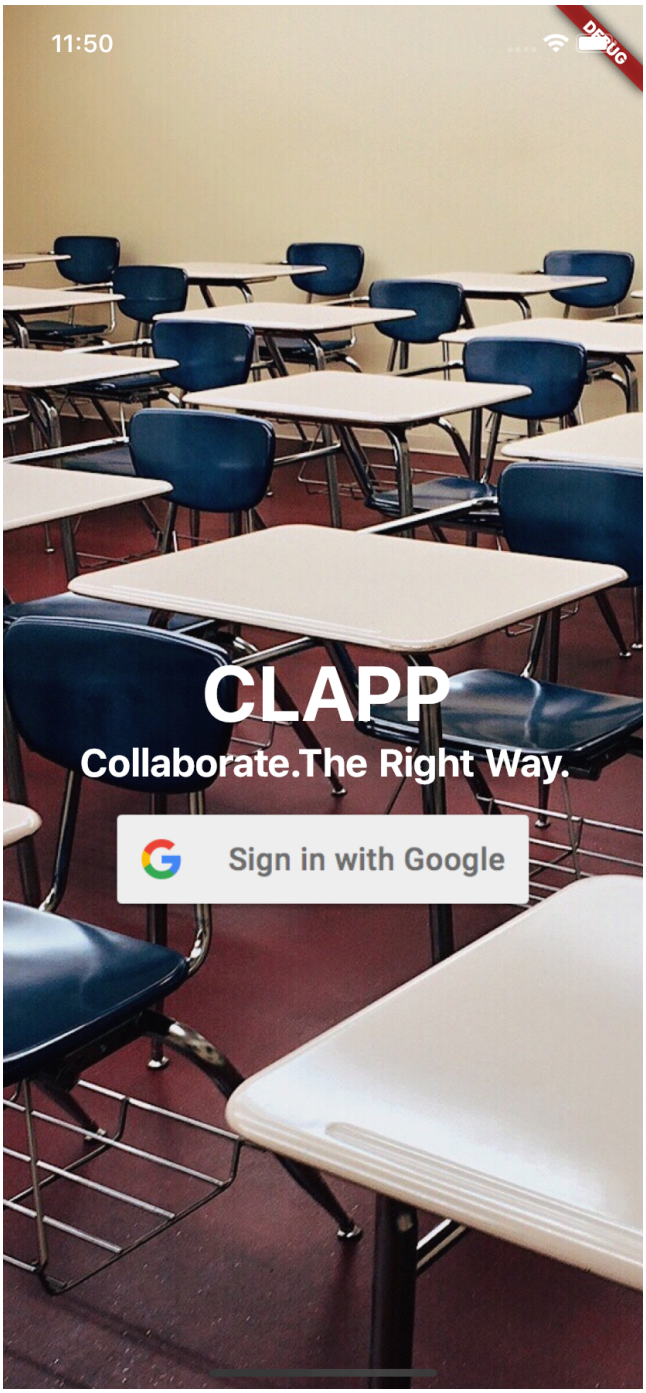
```
import 'postgres.pkg'
import 'firebase_core.pkg'
import 'adminfunctions'
main()
{
        PostgresConnection postgresConnection = new PostgresConnection();
        var data = postgresConnection.getdata(classid);
        Menu menu = options(Remove Member,Change Password,Add
Admin,Dropout As Admin,Modify Timetable,Delete Documents);
        display(menu);
```

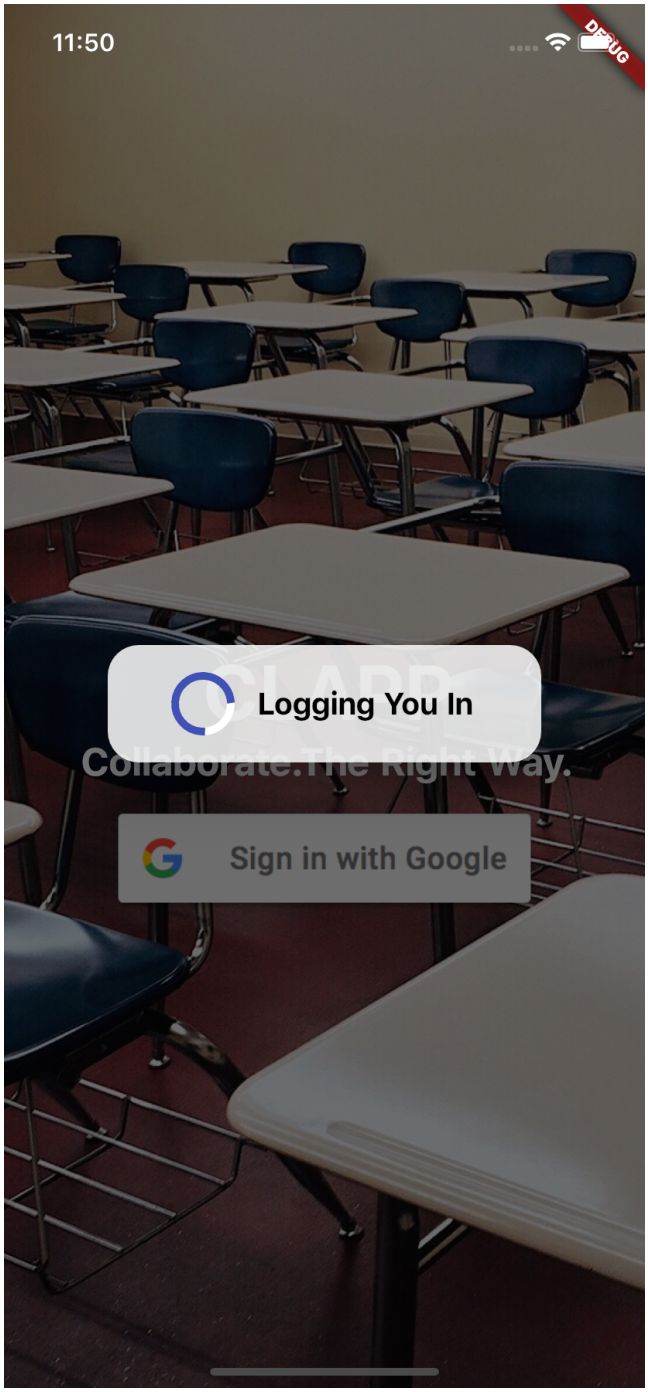
```
onselect(menu)
{
        // REMOVE MEMBER MODULE
        if(selectedindex == 1)
        {
                var members = data.members;
                Listview mlist = ListView.build(members);
                display(mlist);
                var selected[];
                onselect(mlist)
                {

postgresConnection.delete(classid.members[selectedindex]);
                }
        }
        // CHANGE PASSWORD MODULE
        else if(selectedindex == 2)
        {
                String newpassword;
                read(newpassword);
                postgresConnection.write(classid,newpassword);

        }
        // ADD ADMIN MODULE
        else if(selectedindex == 3)
        {
                var members = data.members;
                Listview mlist = ListView.build(members);
                display(mlist);
                var selected[];
                int admincount =
postgresConnection.getdata(count(classid.members[].isAdmin==1));
                onselect(mlist){
                        if(admincount<2)

postgresConnection.write(classid.members[selectedindex].isAdmin:1);
                }}
        // DROPOUT AS ADMIN MODULE
        else if(selectedindex == 4){
                var members = data.members;
                Listview mlist = ListView.build(members);
                display(mlist);
                var selected[];
```

```
                int admincount =
postgresConnection.getdata(count(classid.members[].isAdmin==1));
                onselect(mlist){
                        if(admincount<2)

postgresConnection.write(classid.members[selectedindex].isAdmin:1);
                }
                postgresConnection.write(classid.user.isAdmin:0);

        // MODIFY TIMETABLE MODULE
        else if(selectedindex == 5){
                date _date;
                display('Select Date');
                read(date);
                var tt = data.timetable[date];
                ListView timetable = ListView.build(tt);
                display(timetable);
                onselect(timetable){
                        read(tt[selectedindex]);
                        postgresConnection.write(classid.timetable[date]:tt);
                }
        }
        // DELETE DOCUMENTS MODULE
        else if(selectedindex == 6){
                var subs = data.subjects;
                ListView subjects = ListView.build(subs);
                display(subjects);
                onselect(subjects){
                        ListView documents =
ListView.build(data.subjects[selectedindex].documents);
                        int selectedsub = selectedindex;
                        display(documents);
                        onselect(documents)
                        {

postgresConnection.delete(classid.subjects[selectedindex].documents[selectedin
dex]);
}}}}}
```

RESULTS:

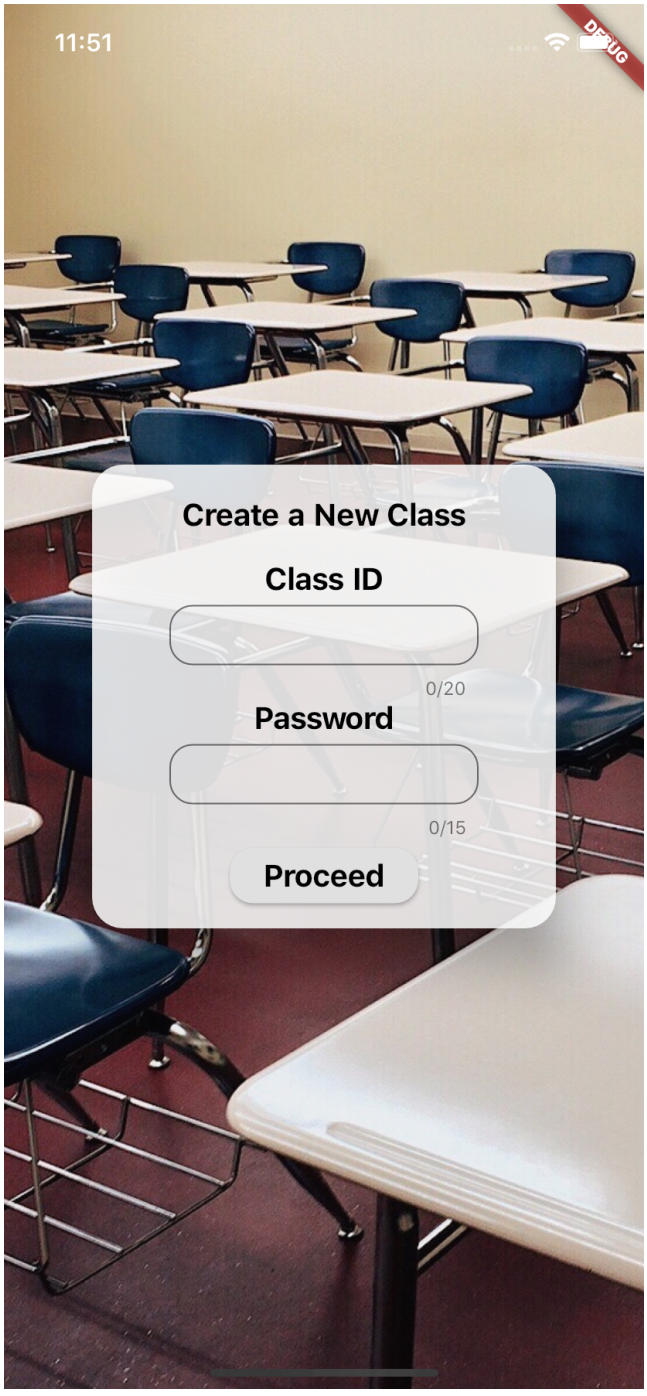
MODULE 1 : USER SIGN IN

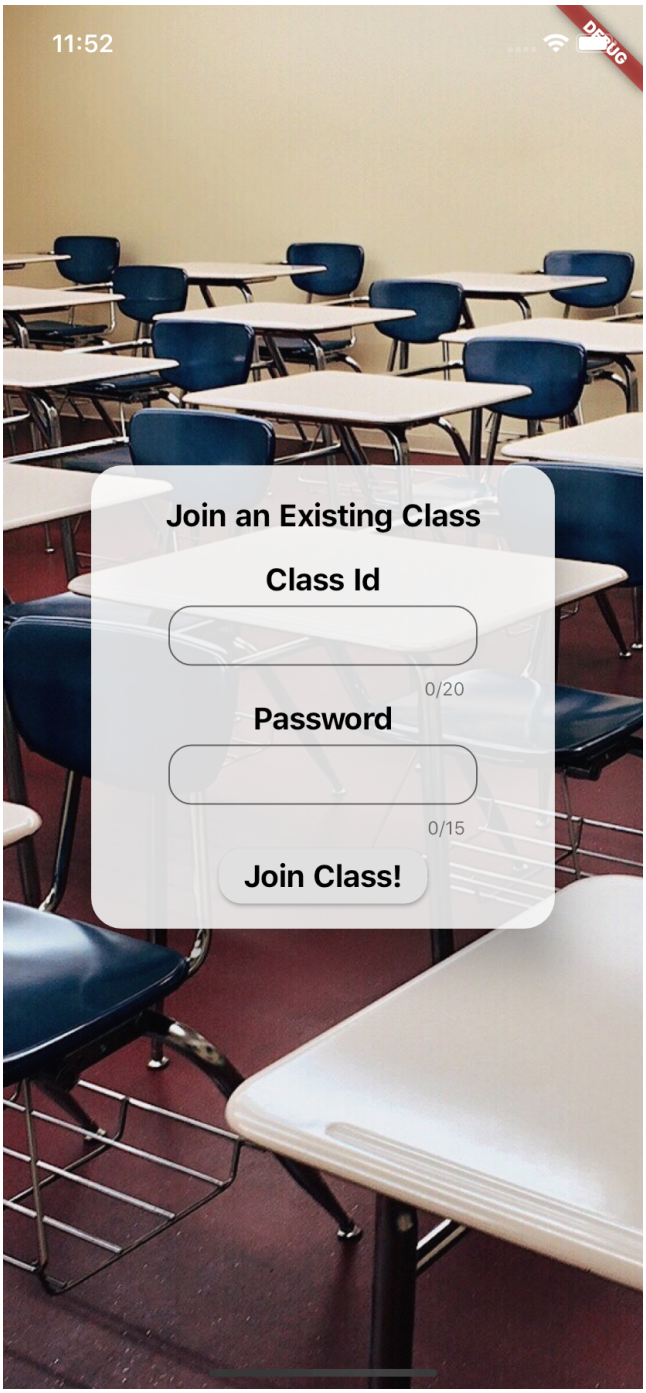

11:50
DEBUG
CLAPP
Collaborate.The Right Way.
Sign in with Google


11:50
DEBUG
Logging You In
Collaborate.The Right Way.
Sign in with Google

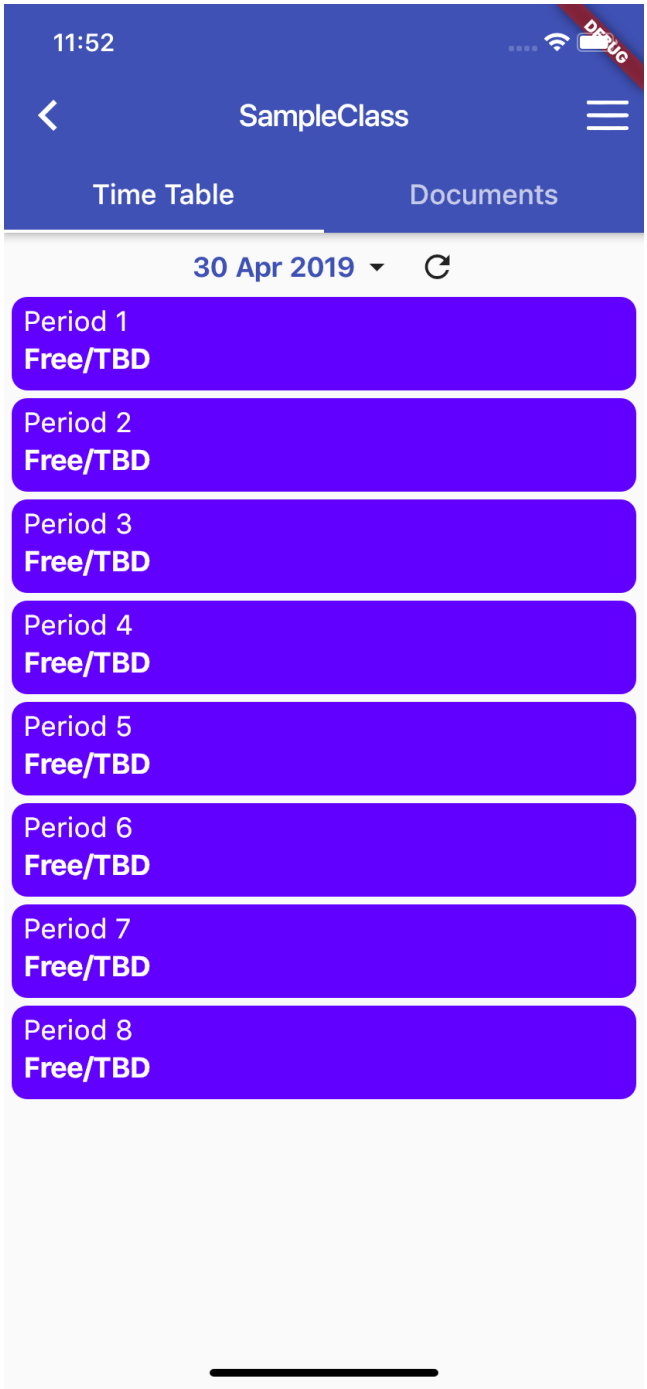
MODULE 2 : PROFILE SELECTION

MODULE 3 : CREATING A NEW CLASS
MODULE 4 : JOINING A NEW CLASS

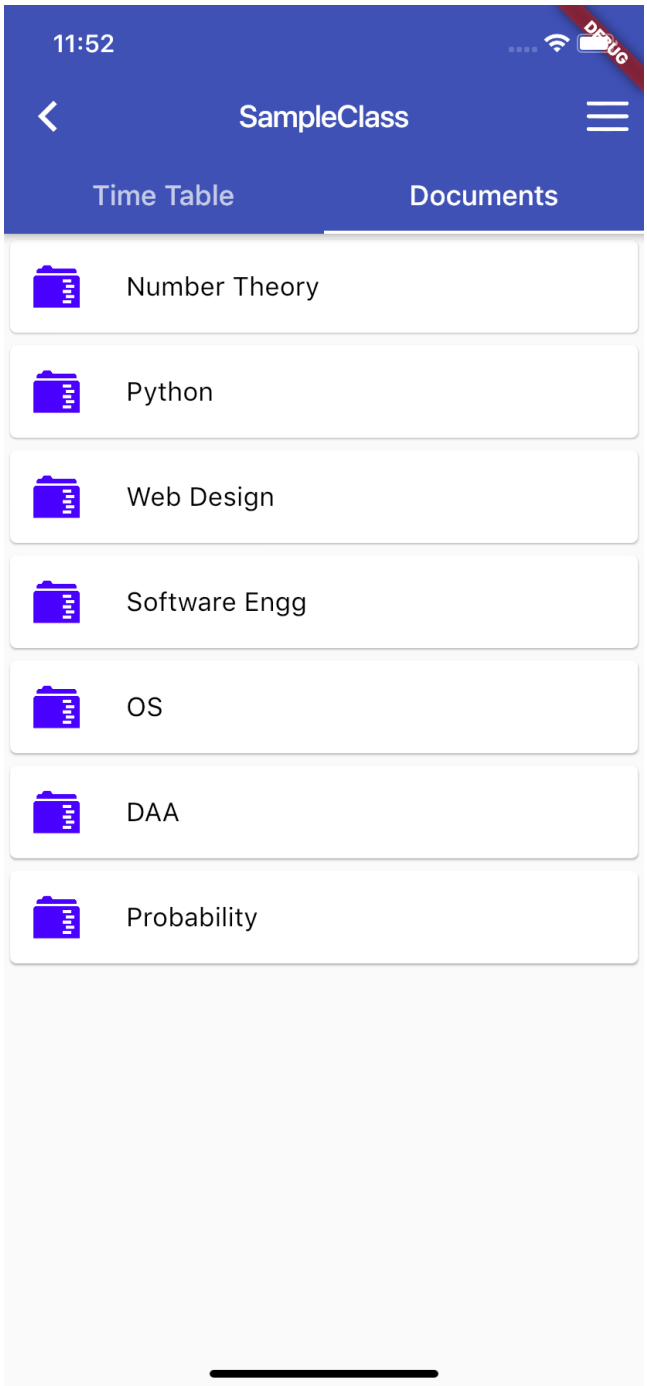


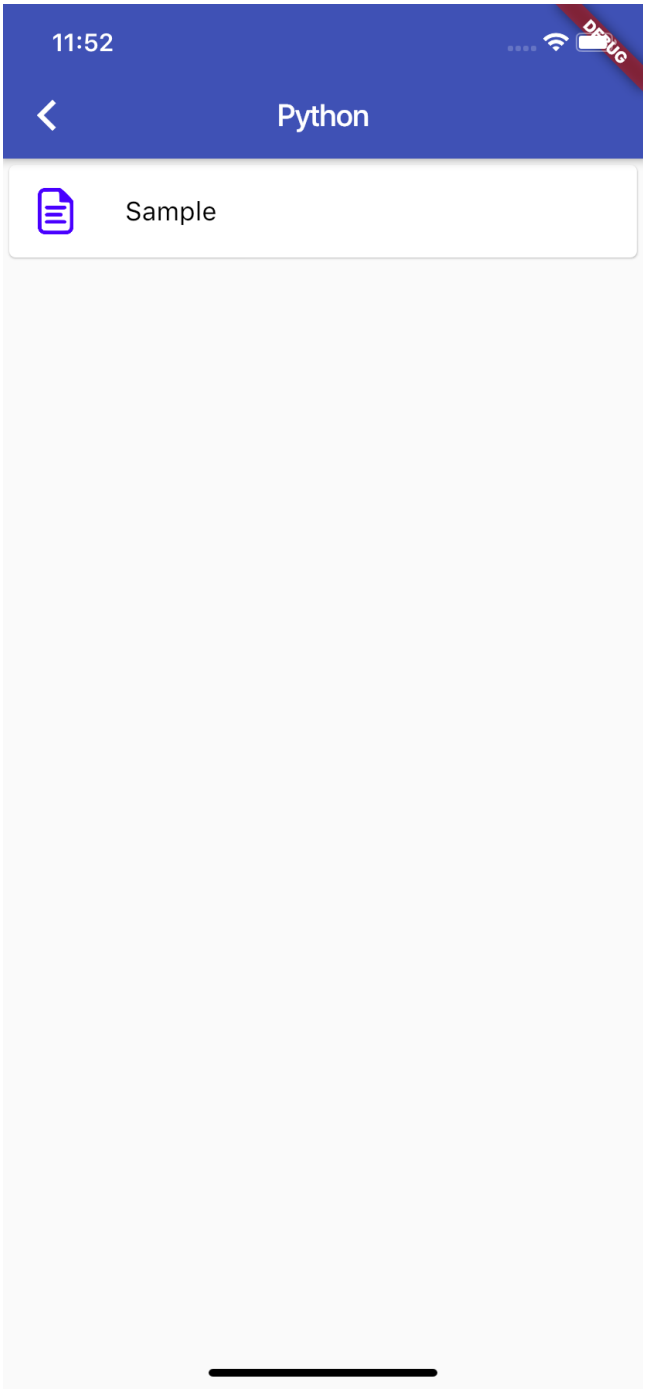


MODULE 5 : VIEWING TIME TABLE

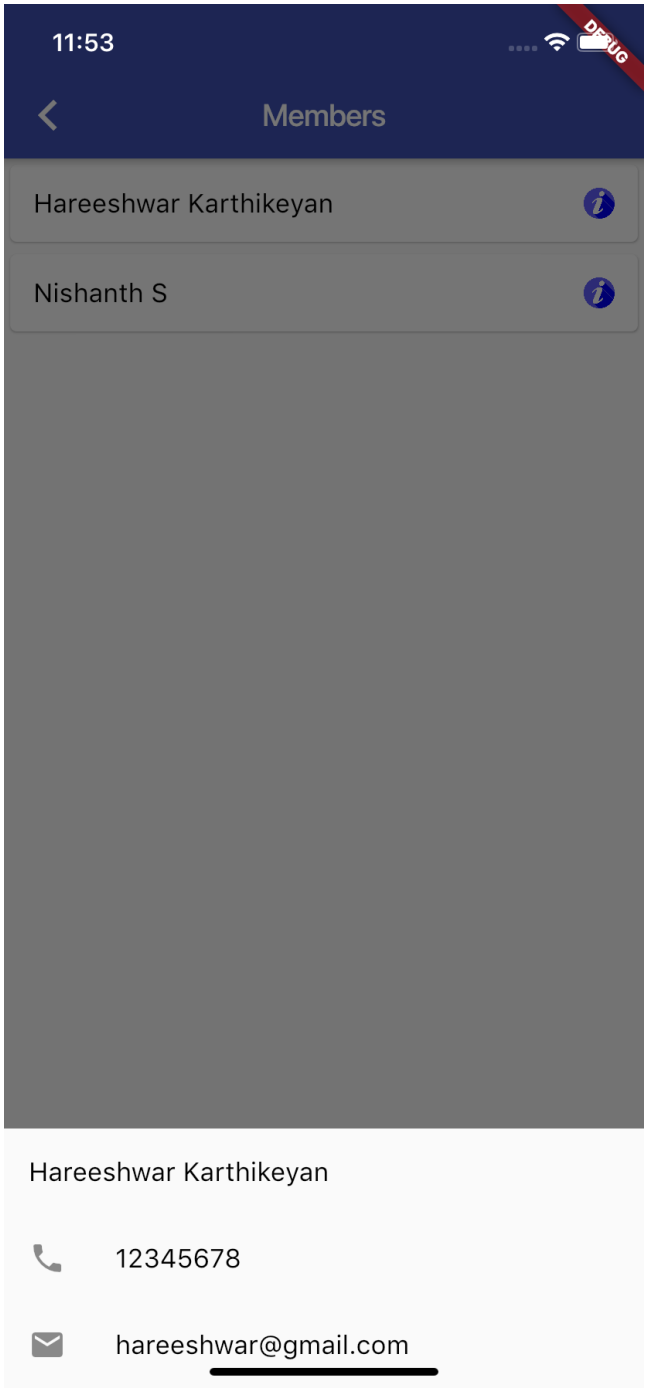


MODULE 6 : DOCUMENTS DOWNLOAD

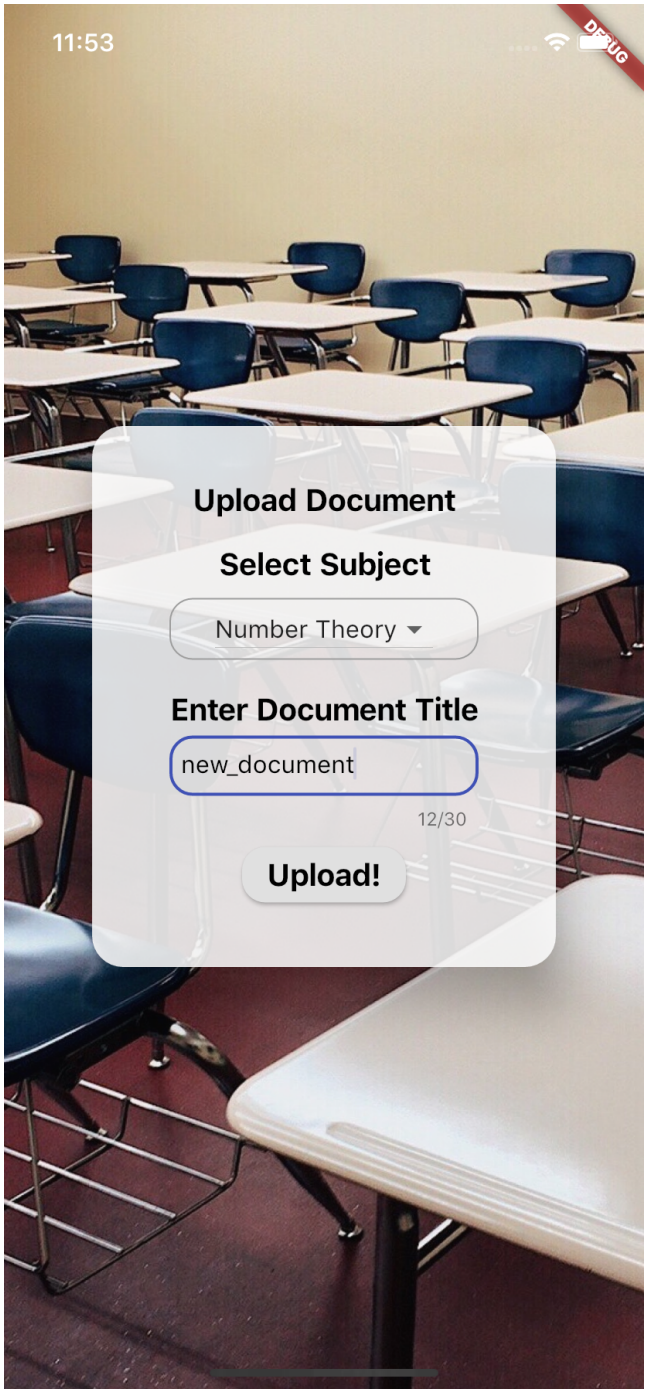




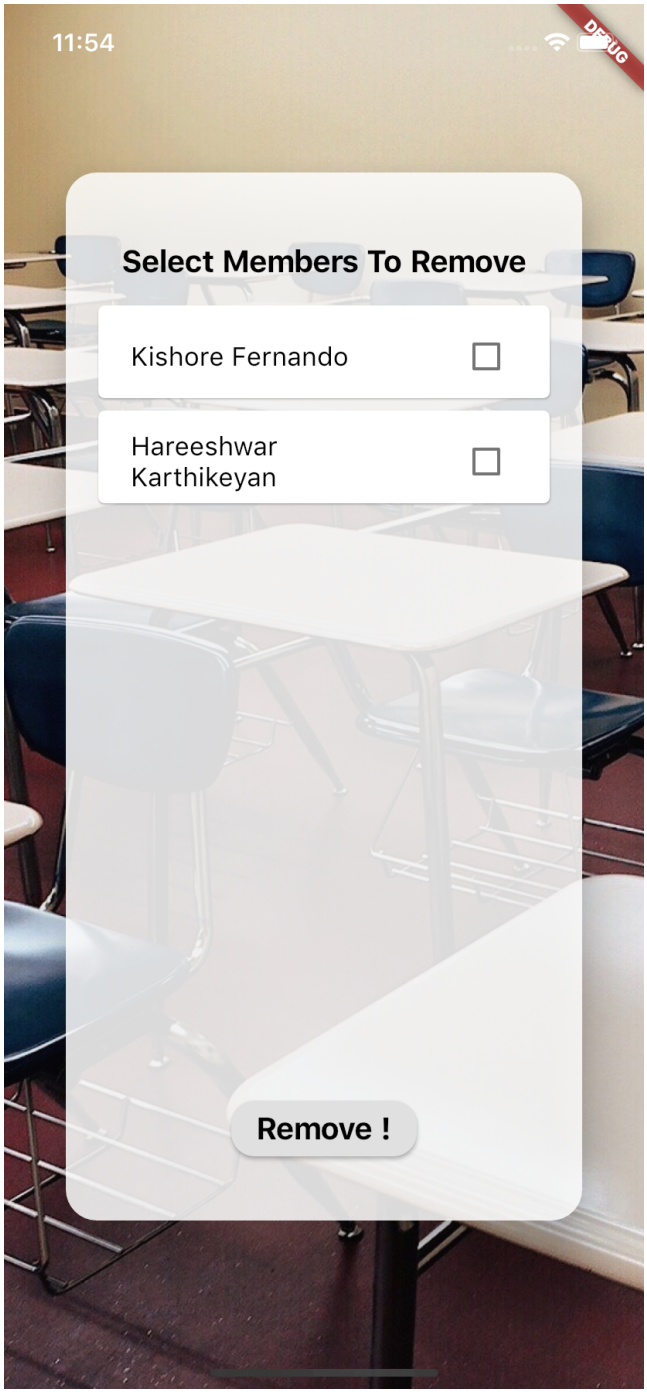
MODULE 7 : VIEW CONTACT INFO

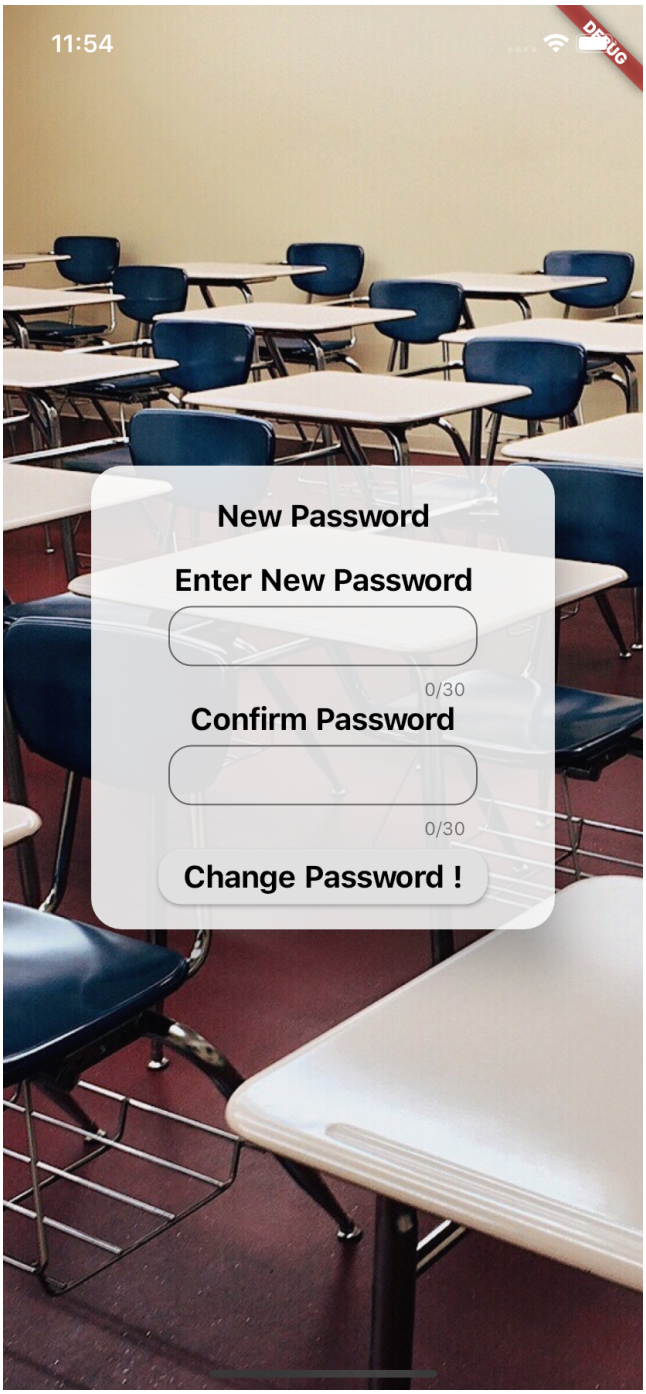

11:53
DEBUG
Members
Hareeshwar Karthikeyan
Nishanth S
Hareeshwar Karthikeyan
12345678
hareeshwar@gmail.com

MODULE 8 : DOCUMENTS UPLOAD

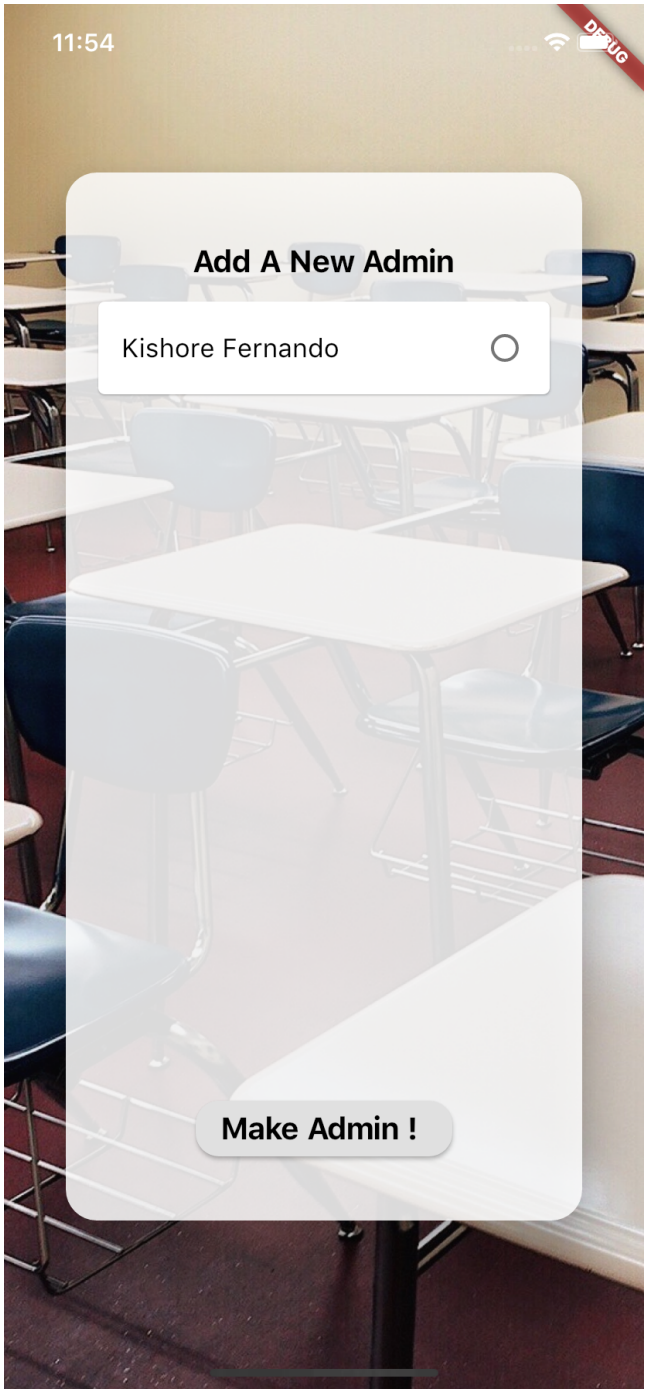


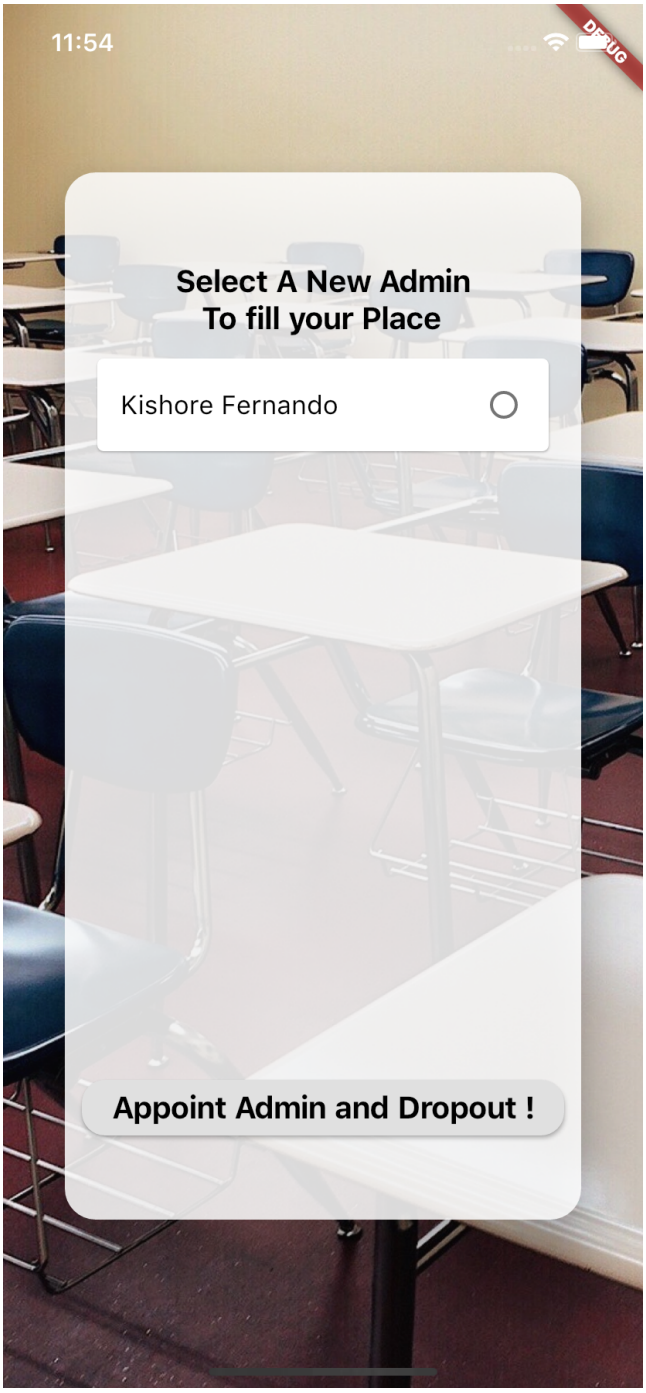
MODULE 9 : REMOVE MEMBER
MODULE 10 : CHANGE PASSWORD



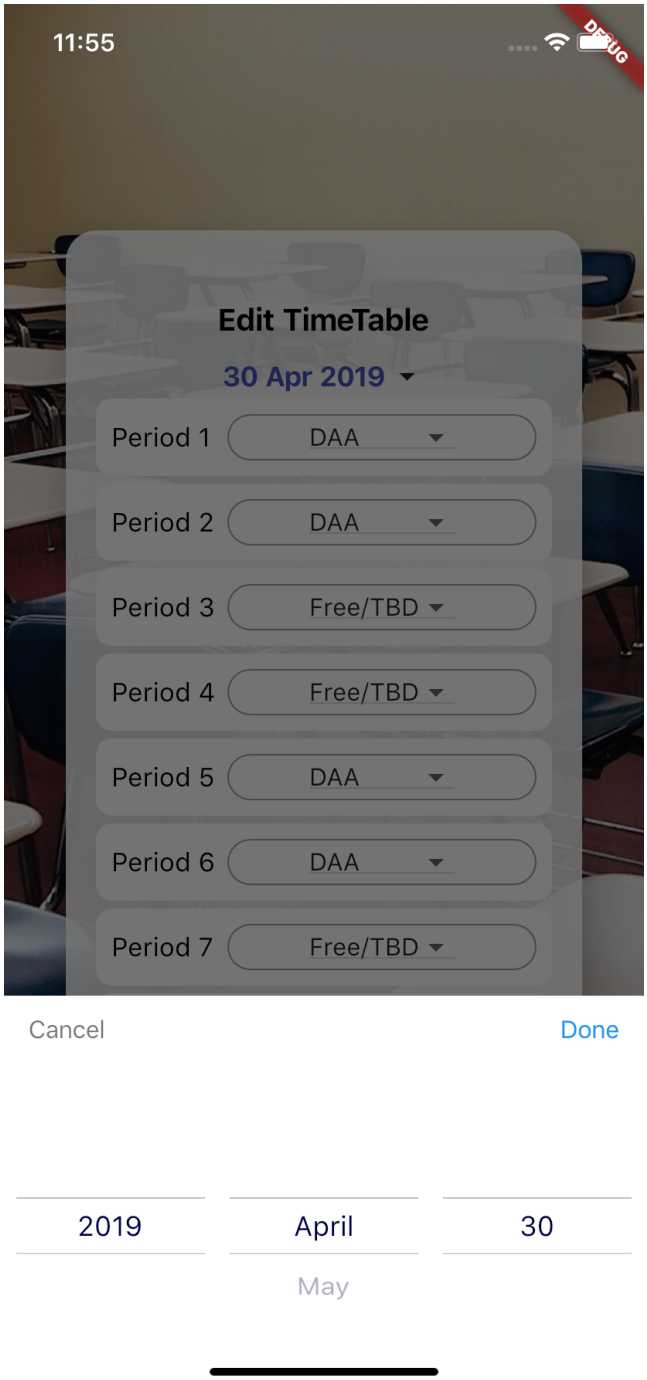


MODULE 11 : APPOINTING ADMINS
MODULE 12 : DROPOUT FROM BEING ADMIN





MODULE 13 : MODIFYING TIME TABLE
MODULE 14 : DOCUMENTS DELETION



5. TESTING

6. CONCLUSION AND FUTURE ENHANCEMENT

Clapp is an application that can be used in every possible classroom scenario ranging from schools to colleges. Today we have our documents for each class scattered around in our phones. Our application helps you solve the hassle of organising your documents together and promoting better collaboration with classmates.

This report has effectively given insight on the various functionalities of the software and how they have been effectively tested using the white box testing technique.

In conclusion all the desired features have been analysed, designed, coded, and tested.

As a

7. BIBLIOGRAPHY

The following are the resources utilised for the creation of the project.

1. Shang Huang , “How to integrate your iOS Flutter App with Firebase on MacOS” , Publication date March 4 2019 , https://medium.freecodecamp.org/how-to-integrate-your-ios-flutter-app-with-firebase-on-macos-6ad08e2714f0/
2. “Postgres flutter” , Publication date October 9 2018 , https://pub.dartlang.org/packages/postgres/
3. Akul Tomar , “How to get started with PostgreSQL” , Publication date July 2 2018 , https://medium.freecodecamp.org/how-to-get-started-with-postgresql-9d3bc1dd1b11/
4. Aseem Wangoo , “Firebase google sign in and Flutter” , Publication date June 10 2018 , https://medium.com/flutterpub/firebase-google-sign-in-and-flutter-6f7abde9ae5a/
5. “Flutter Documentation” , Publication date February 26 2019 , https://flutter.dev/docs/
6. Aseem Wangoo , “Table in flutter”, Publication date September 30 2018 , https://medium.com/flutterpub/table-in-flutter-faaf0b5f6a0b/
7. Nils Backe , “Flutter Alert dialogs” , Publication date June 28 2018 , https://medium.com/@nils.backe/flutter-alert-dialogs-9b0bb9b01d28/

Appendix A

Dart and Flutter

Flutter is Google's mobile UI framework for crafting high-quality native interfaces on iOS and Android in record time. Flutter works with existing code, is used by developers and organisations around the world, and is free and open source. More details can be found at https://fl utter.io

Dart is a general-purpose programming language originally developed by Google and later approved as a standard by Ecma (ECMA-408). It is used to build web, server, desktop, and mobile applications.
Dart is an object-oriented, class defined, garbage-collected language using a C-style syntax that transcompiles optionally into JavaScript. It supports interfaces, mixins, abstract classes, reified generics, static typing, and a sound type system.

Dart is the language used to build flutter apps.

Appendix B

PostgreSQL

PostgreSQL is a general purpose and object-relational database management system, the most advanced open source database system. PostgreSQL was developed based on POSTGRES 4.2 at Berkeley Computer Science Department, University of California.

PostgreSQL was designed to run on UNIX-like platforms. However, PostgreSQL was then also designed to be portable so that it could run on various platforms such as Mac OS X, Solaris, and Windows.

PostgreSQL is free and open source software. Its source code is available under PostgreSQL license, a liberal open source license. You are free to use, modify and distribute PostgreSQL in any form.

PostgreSQL requires very minimum maintained efforts because of its stability. Therefore, if you develop applications based on PostgreSQL, the total cost of ownership is low in comparison with other database management systems.

Appendix C

Firebase

Firebase is a mobile and web app development platform that provides developers with a plethora of tools and services to help them develop high-quality apps, grow their user base, and earn more profit.

A Brief History

Back in 2011, before Firebase was Firebase, it was a startup called Envolve. As Envolve, it provided developers with an API that enabled the integration of online chat functionality into their website.
What's interesting is that people used Envolve to pass application data that was more than just chat messages. Developers were using Envolve to sync application data such as a game state in real time across their users.
This led the founders of Envolve, James Tamplin and Andrew Lee, to separate the chat system and the real-time architecture. In April 2012, Firebase was created as a separate company that provided Backend-as-a-Service with *real-time functionality*.
After it was acquired by Google in 2014, Firebase rapidly evolved into the multifunctional behemoth of a mobile and web platform that it is today.

We use firebase in this project to implement google sign-in in the application.