ONLINE CLASS MONITORING SYSTEM

A MINI PROJECT REPORT

Submitted By

BOOBESHWARAN C R	**AC17UIT007**
SUJITH G	**AC17UIT065**
TAMIZHSELVAN G	**AC17UIT067**

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

ADHIYAMAAN COLLEGE OF ENGINEERING

(AUTONOMOUS)

Dr. M. G. R NAGAR, HOSUR 635 - 109

ANNA UNIVERSITY :: CHENNAI 600 - 025

JANUARY 2021

ANNA UNIVERSITY :: CHENNAI 600 - 025

BONAFIDE CERTIFICATE

Certified that this mini project report **"ONLINE CLASS MONITORING SYSTEM**" is the bonafide work of "**BOOBESHWARAN C R (AC17UIT007), SUJITH G (AC17UIT065), TAMIZHSELVAN G (AC17UIT075)**" who carried out the project work under my supervision.

SIGNATURE

Dr.D.THILAGAVATHY, M.E., Ph.D.,
HEAD OF THE DEPARTMENT
Professor,
Dept. of Information Technology,
Adhiyamaan College of
Engineering, (Autonomous)
Hosur-635 109.

SIGNATURE

Mrs.A.SAJITHABEGAM, M.Tech.,
SUPERVISOR
Assistant Professor,
Dept. of Information Technology,
Adhiyamaan College of
Engineering, (Autonomous)
Hosur-635 109.

Submitted for the VIVA-VOCE held on .....
at Adhiyamaan College of Engineering (Autonomous), Hosur.

INTERNAL EXAMINAR **EXTERNAL EXAMINAR**

ACKNOWLEDGEMENT

It is one of the most difficult tasks in life to choose the appropriate words to express one's gratitude towards the beneficiaries.

We are very much grateful to God who helped us all the way throughout the project and molded us into what we are today.

We show our sincere thanks to **Dr.G.RANGANATH, M.E., Ph.D.,** Principal, Adhiyamaan College of Engineering, Hosur, for this valuable opportunity and environment given to us.

We extend our hearty gratitude to **Dr.D.THILAGAVATHY, M.E., Ph.D.,** Professor and Head of the Department, Department of Information Technology, Adhiyamaan College of Engineering, Hosur, for her guidance and valuable suggestion and encouragement throughout this project.

We are highly indebted to **Mrs.A.SAJITHABEGAM M.Tech.,** Supervisor, Assistant Professor, Department of Information Technology, Adhiyamaan College of Engineering, Hosur, whose immense support encouragement and valuable guidance were responsible to complete the project successfully.

We extend our hearty thankfulness to our Project Coordinators and all our department faculties for their support in helping us complete this project successfully.

Finally, we would like to thank our parents and friends, without their motivation and support it would not have been possible for us to complete this successfully.

ABSTRACT

This Online Class Monitoring System is an android application which mainly helps to improve communication between faculty and student. This application actually consists of two modules, those are admin module and user module, where they can access and operate the application from their respective ends during online classes faculty can monitoring student's attendance, student login, and logout activity within this application.

In general, user who created the class in the application will act as admin of the class. The admin will maintain the entire administration like authorization, authentication, permissions, access history, and troubleshooting, etc. Admin will access the class participants information. user will receive any kind of information by the notification tab. Admin can delete or edit the created classes. The user who joined the class with the class id are called as class participants. The class participants will access all information like subject name as class name, faculty name, class scheduling, attendance details, shared notes, messaging, share study materials through class chatting system and a good relationship with the faculty and classmates.

This application maintains the daily attendance of each class participants with the class scheduling time. It will help us to improve attendance system and communication with less time consumption.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACRONYMS	ABBREVIATION
API	Application Programming Interface
HTTP	Hypertext Transfer Protocol
IDE	An Integrated Development Environment
JDK	Java Development Kit
QR	Quick Response
RAM	Random Access Memory
ROM	Read Only Memory
SDK	Software Development Kit
SVG	Scalable Vector Graphics
UI	User Interface
XML	Extensible Markup Language

CHAPTER - 1
INTRODUCTION

1.1OBJECTIVES

This application will be done on android studio which is costless platform to develop android application. To establish an online class monitoring system aims to provide an interface for the faculty member to monitor their student's attendance within the mobile during online classes. It provides effective communication between faculty and students. In this application both faculty and student can access and operate the application from their respective ends.

1.2 SCOPE AND NOVELITY

The scope of the system is to helps to communicate between faculty and student i.e., Application users on their online classes. Users can manage their attendance, send or receive messages, and share materials through this application. This application uses chat activity with good user interface for better communication.

The novelty of this system is that it gives instant attendance to them by monitoring login activity within this application about where the user is prompting and it helps the user to communicate with the system using android compatible interacting with video conference.

1.3 PROJECT OVERVIEW

Online Class Monitoring System is an android application which helps to communication between faculty and student during online classes. This application maintains student attendance, chat between faculty and students, shared study materials using database system. Student attendance are generated by class start time, and class end time interacting with video conference.

CHAPTER - 2
LITERATURE SURVEY

Sham B. Mehar “College Staff uploads attendance, results and college notifications”, 2017[1]. The design and implementation of the system is to provide service in institute and colleges. The system is to provide comprehensive student information system and user interface is to replace the current paper records. College Staff uploads attendance, results and college notifications through a secure, online interface using android devices. All data is thoroughly reviewed and validated on the server before actual record alteration occurs. The system plans for student user interface, allowing students to access tips and tricks as provided by their seniors. All data is stored securely on SQL servers managed by the college Administrator.

Mr. Yash Mittal “Biometric Based Attendance”, 2018[2]. In this paper Biometric scaled up for real time deployment, it provides solution of late coming.

Prof. G. N. Dhoot “Finger Based Attendance Management with SMS Alert to Parents”, 2018[3]. This paper introduced system include terminal fingerprint module and attendance module and SMS system for alerting parents for updating about their child.

Puja K. Chavhan “Student Attendance Management”, 2018[4] is defined which is traditionally taken manually by faculty. One alternative to make student attendance system automatic is provided by Computer Vision. In this paper we review the various computerized system which is being developed by using different techniques. Based on this review a new approach for student attendance recording and management is proposed to be used for various colleges or academic institutes.

Prof. H.B. Sale “Smart attendance Management and Learning System”, 2019[5]. This paper facility of notes dictation, defaulter list, notes view, notification, details view for students, staff, teachers and Admin.

Chinmay Kulkarni “Key Authentication Based Door Lock Monitoring System”, 2018[6]. This project is concentrated more on automation of institute security provides lesser security than actual physical security.

S. Mohan kumar “Employee Attendance Monitoring System Using radio frequency Identity Card”, 2019[7]. Facilitates automatic wireless identification using ID tags and reader method.

Sham B. Mehar This mobile application will require connecting to the internet through Wi-Fi (Wireless Fidelity) technology or through GPRS (General Packet Radio Service) [8]. Lecturers will first have to sign up for this and then they can take attendance any time they wish by first logging in with the help of a smartphone to the server. After attendance has been taken lecturer will send it over to sever via GPRS. The lecturers can also enrol new students, delete information about a particular student, modify some information etc.

Mekshyam Z. Lanjewar “Attendance Management System”, 2018[9] is to computerized the tradition way of taking attendance. Another purpose for developing this software is to generate the report automatically at the end of the session or in the between of the session.

Puja K. “mobile learning”, 2016[10]. Mobile learning is the next generation of e-learning that leads attractive way of knowledge delivery especially used in teaching and learning process. With development of this Android application the student preferred to use mobile devices as technology supported educational tool. This system is designed because notes dictation in the class is difficult considering semester duration, student might miss the exam and important notice displayed due to unawareness.

CHAPTER - 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In existing system there is no application to give communication between faculty and students by the mobile. In the previous system, this process will take on online websites for sharing information between faculty and students. Instead of sharing the information through online websites there is no any option for developing communication by giving attendance in predefined methods.

3.1.1 DRAWBACKS

- In previous attendance system data stored locally to the particular user.
- There is no particular application to communicate between student and teacher with attendance features.
- Attendance system available only in only website for private application.

3.2 PROPOSED SYSTEM

In this online class monitoring system application, proposing a new form of technology to interacting with video conference to get daily attendance. Students will get their daily attendance, study materials, class scheduling with in the application. Faculty will get the student attendance with active time in their respective classes it will monitored through video conference. Here the students receive notifications if there is any important information available for them.

3.2.1 ADVANTAGES

- This system stores the data into firebase database.
- It will access by the application user through created classes.
- User will join class simply type the class id and join the class.
- Materials will be shared between teacher and student.
- Class scheduling time by faculty member.

CHAPTER - 4

SYSTEM SPECIPICATIONS

4.1 SOFTWARE REQUIREMENTS

Operating System : Windows XP/7/8/10

Coding Language : JAVA

Front End : XML

Authentication : Firebase Authentication

Database : Firebase Database

Storage : Firebase Storage

Tool Kit : Android SDK

IDE : Android Studio

4.2 HARDWARE COMPONENTS

Processor : Intel I3 3rd Generation

ROM : 40GB

RAM : 1GB

Mobile : Android Version 7

CHAPTER - 5
SOFTWARE DESCRIPTION

5.1 JAVA

Java is a class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client-server web applications, with a reported 9 million developers.

Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle) and released in 1995 as a core component of Sun Microsystems' Java platform. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GNU General Public License. Oracle offers its own HotSpot Java Virtual Machine, however the official reference implementation is the OpenJDK JVM which is free open source software and used by most developers including the Eclipse IDE and is the default JVM for almost all Linux distributions.

The latest version is Java 15, released in September 2020, with Java 11, a currently supported long-term support (LTS) version, released on September 25, 2018; Oracle released for the legacy Java 8 LTS the last zero-cost public update in January 2019 for commercial use, although it will otherwise still support Java 8 with public updates for personal use indefinitely. Other vendors have begun to offer zero-cost builds of OpenJDK 8 and 11 that are still receiving security and other upgrades. Oracle (and others) highly recommend uninstalling older versions of Java because of serious risks due to unresolved security issues. Since Java 9, 10, 12 and 13 are no longer supported, Oracle advises its users to immediately transition to the latest version (currently Java 15) or an LTS release.

5.2 XML

Extensible Mark-up Language (XML) is a mark-up language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The World Wide Web Consortium's XML 1.0 Specification of 1998 and several other related specifications all of them free open standards define XML.

The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services.

Hundreds of document formats using XML syntax have been developed, including RSS, Atom, SOAP, SVG, and XHTML. XML-based formats have become the default for many office-productivity tools, including Microsoft Office (Office Open XML), OpenOffice.org and LibreOffice (OpenDocument), and Apple's iWork. XML has also provided the base language for communication protocols such

as XMPP. Applications for the Microsoft .NET Framework use XML files for configuration, and property lists are an implementation of configuration storage built on XML.

Many industry data standards, such as Health Level 7, Open Travel Alliance, FpML, MISMO, and National Information Exchange Model are based on XML and the rich features of the XML schema specification. Many of these standards are quite complex and it is not uncommon for a specification to comprise several thousand pages. In publishing, Darwin Information Typing Architecture is an XML industry data standard. XML is used extensively to underpin various publishing formats.

XML is widely used in a Service-oriented architecture (SOA). Disparate systems communicate with each other by exchanging XML messages. The message exchange format is standardised as an XML schema (XSD). This is also referred to as the canonical schema. XML has come into common use for the interchange of data over the Internet. IETF RFC:3023, now superseded by RFC:7303, gave rules for the construction of Internet Media Types for use when sending XML. It also defines the media types `application/xml` and `text/xml`, which say only that the data is in XML, and nothing about its semantics.

5.3 FIREBASE AUTHENTICATION

Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more. Firebase Authentication integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend. You can sign in users to your Firebase app either by using

FirebaseUI as a complete drop-in auth solution or by using the Firebase Authentication SDK to manually integrate one or several sign-in methods into your app.

Authenticate users with their email addresses and passwords. The Firebase Authentication SDK provides methods to create and manage users that use their email addresses and passwords to sign in. Firebase Authentication also handles sending password reset emails. To sign a user into your app, you first get authentication credentials from the user. These credentials can be the user's email address and password, or an OAuth token from a federated identity provider. Then, you pass these credentials to the Firebase Authentication SDK. Our backend services will then verify those credentials and return a response to the client.

After a successful sign in, you can access the user's basic profile information, and you can control the user's access to data stored in other Firebase products. You can also use the provided authentication token to verify the identity of users in your own backend services.

5.4 FIREBASE DATABASE

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in Realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

Instead of typical HTTP requests, the Firebase Realtime Database uses data synchronization—every time data changes, any connected device receives that update within milliseconds. Provide collaborative and immersive experiences without thinking about networking code. The Firebase Realtime Database lets you build rich, collaborative applications by allowing secure access to the

database directly from client-side code. Data is persisted locally, and even while offline, Realtime events continue to fire, giving the end user a responsive experience. When the device regains connection, the Realtime Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically.

The Realtime Database provides a flexible, expression-based rules language, called Firebase Realtime Database Security Rules, to define how your data should be structured and when data can be read from or written to. When integrated with Firebase Authentication, developers can define who has access to what data, and how they can access it. The Realtime Database is a NoSQL database and as such has different optimizations and functionality compared to a relational database. The Realtime Database API is designed to only allow operations that can be executed quickly. This enables you to build a great Realtime experience that can serve millions of users without compromising on responsiveness. Because of this, it is important to think about how users need to access your data and then structure it accordingly.

5.5 FIREBASE STORAGE

Cloud Storage for Firebase is a powerful, simple, and cost-effective object storage service built for Google scale. The Firebase SDKs for Cloud Storage add Google security to file uploads and downloads for your Firebase apps, regardless of network quality. You can use our SDKs to store images, audio, video, or other user-generated content. On the server, you can use Google Cloud Storage, to access the same files.

Firebase SDKs for Cloud Storage integrate with Firebase Authentication to provide simple and intuitive authentication for developers. You can use our declarative security model to allow access based on filename, size, content type, and other metadata.

Cloud Storage stores your files in a Google Cloud Storage bucket, making them accessible through both Firebase and Google Cloud. This allows you the flexibility to upload and download files from mobile clients via the Firebase SDKs, and do server-side processing such as image filtering or video transcoding using Google Cloud Platform. Cloud Storage scales automatically, meaning that there's no need to migrate to any other provider. Learn more about all the benefits of our integration with Google Cloud Platform. The Firebase SDKs for Cloud Storage integrate seamlessly with Firebase Authentication to identify users, and we provide a declarative security language that lets you set access controls on individual files or groups of files, so you can make files as public or private as you want.

5.6 ANDROID STUDIO

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development.

5.6.1 FEATURES OF ANDROID STUDIO

- Gradle-based build support.
- Android-specific refactoring and quick fixes.
- Lint tools to catch performance, usability, version compatibility and other problems.
- ProGuard integration and app-signing capabilities.
- Template-based wizards to create common Android designs and components.

- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations.
- Support for building Android Wear apps.
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine.
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.

CHAPTER - 6

SYSTEM DESIGN

6.1 ARCHITECTURE DIAGRAM

An architecture diagram is a graphical representation of a set of concepts, that are part of an architecture, user will login with details. It direct to dashboard then user will create or join class to the class group using class id. User data and class data will be stored in database. In class group user will chat between them, share materials, and attendance activity.

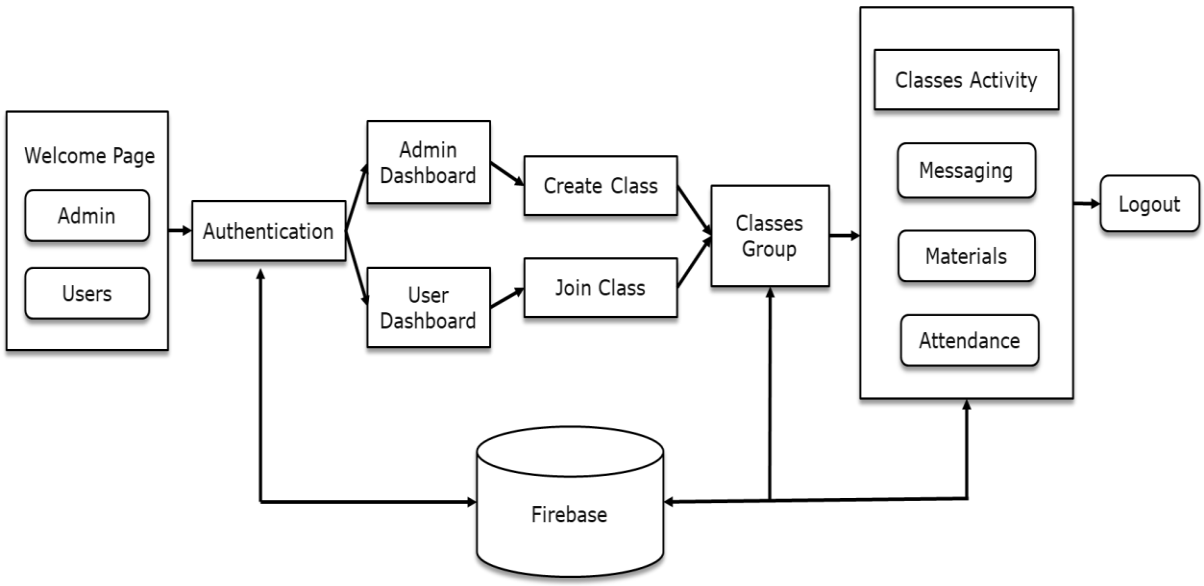


Fig 6.1 Architecture Diagram

6.2 USECASE DIAGRAM

A use case diagram is a group of actors, a set of use cases enclosed by a system boundary communication, association between the actors and the use cases and also generalization among the use cases. A use case diagram shows the relationship among the actors and the use cases within the system. Fig 6.2 shows the actors involved in the Online Class Monitoring System which is the Admin and the users. The admin monitors the records of the users and the user is able to perform all the features.

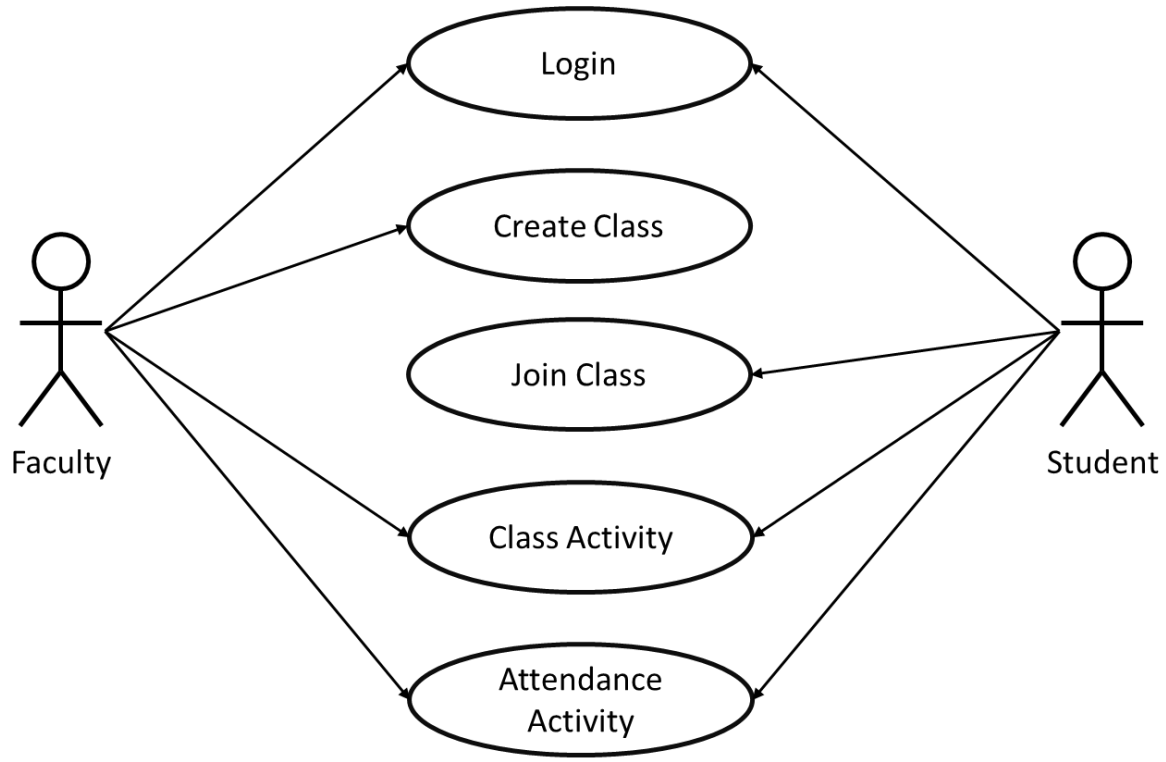


Fig 6.2 Usecase Diagram

6.3 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

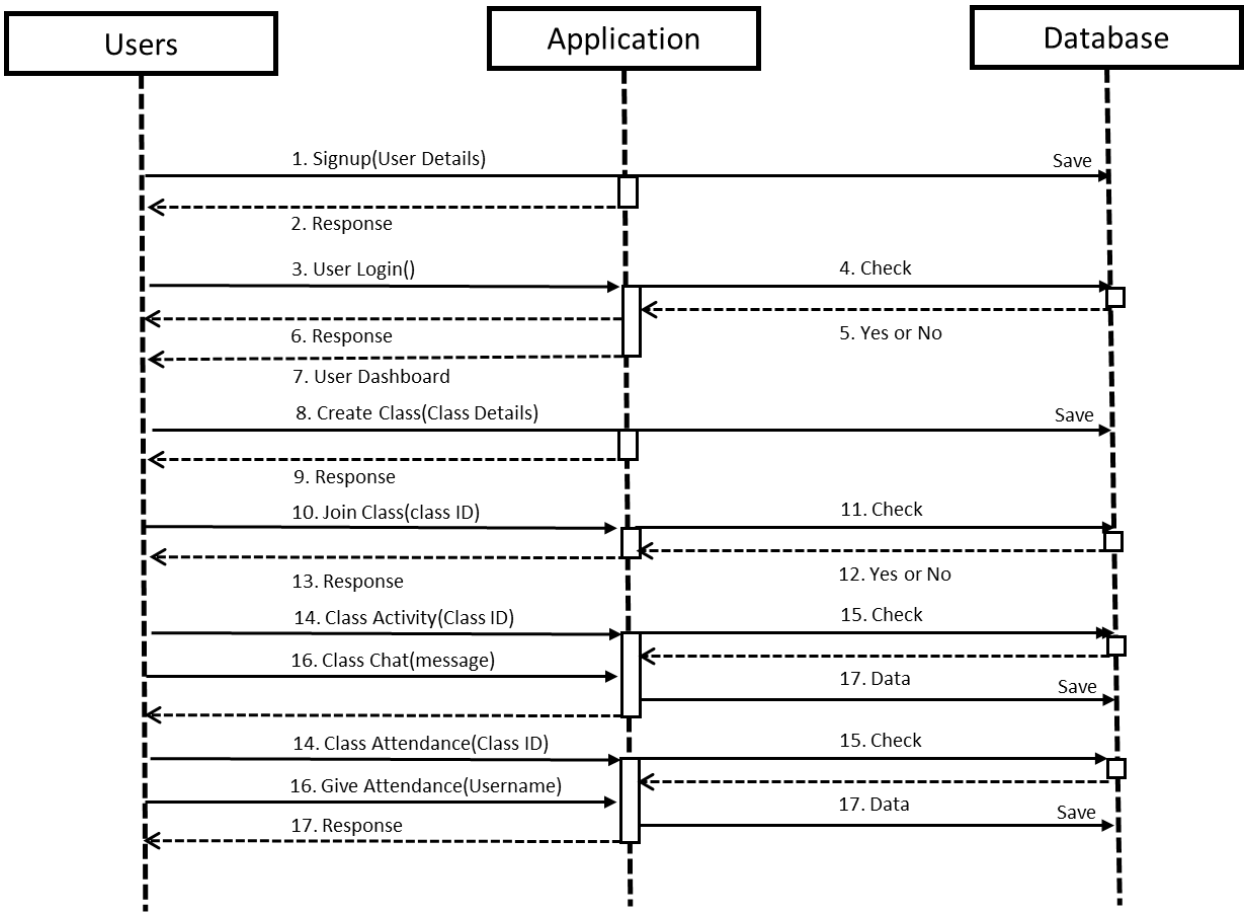


Fig 6.3 Sequence Diagram

6.4 CLASS DIAGRAM

Class diagram provides an overview of the target system by describing the objects and classes inside the system and the relationships between them. It provides a wide variety of usages; from modelling the domain specific data structure to detailed design of the target system. Fig 6.4 shows all the methods and operations done by the admin and the student.

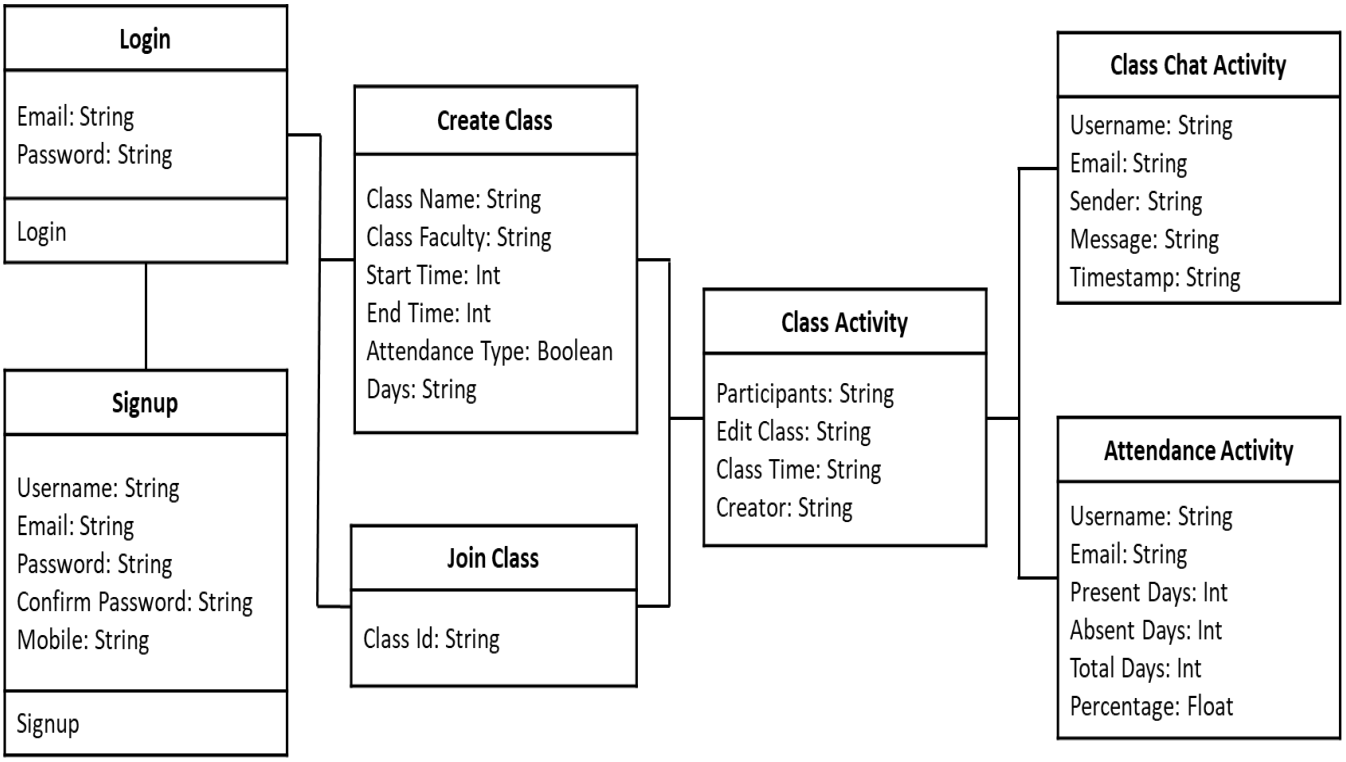


Fig 6.4 Class Diagram

6.5 DATA FLOW DIAGRAM

A data-flow diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

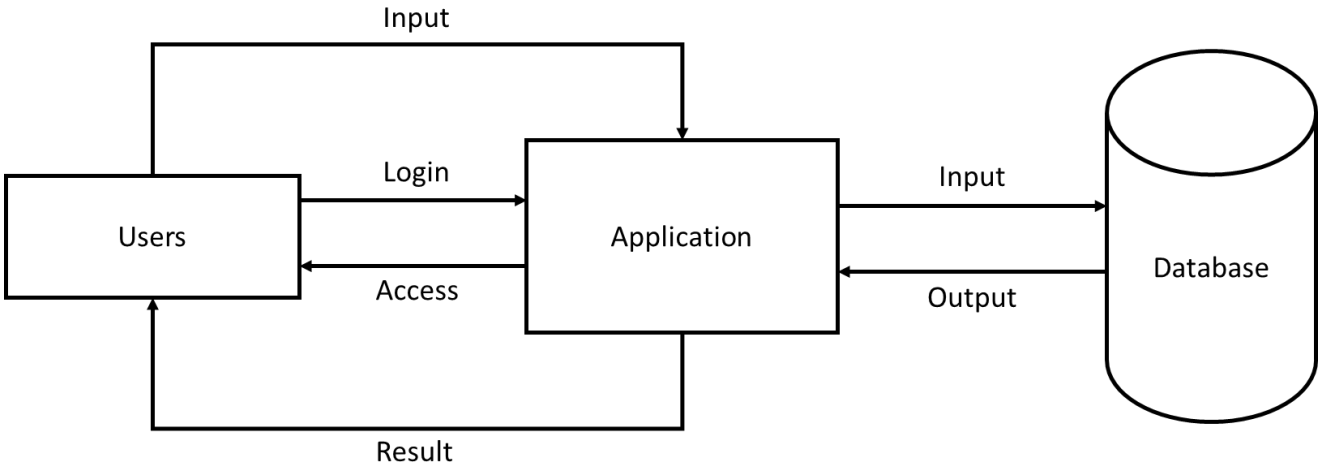


Fig 6.5 Data-Flow Diagram

6.6 DEPLOYMENT DIAGRAM

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them. Deployment diagrams are typically used to visualize the physical hardware and software of a system.

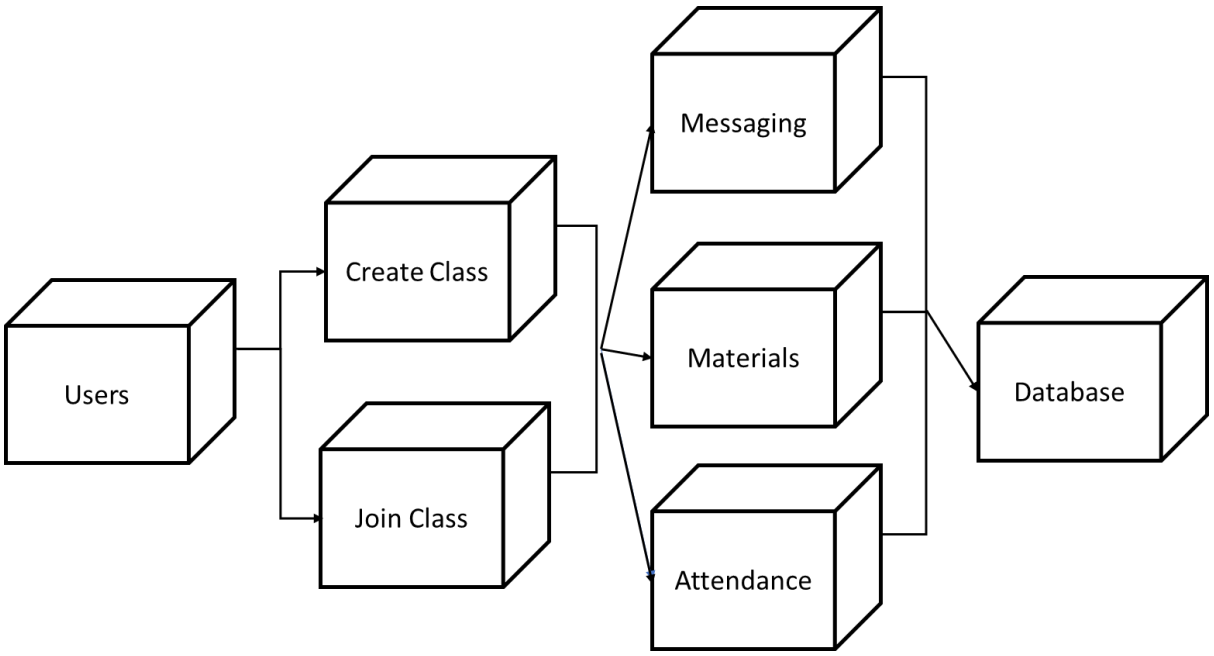


Fig 6.5 Deployment Diagram

CHAPTER - 7
PROJECT DESCRIPTION

7.1 OVERVIEW OF THE PROJECT

Online Class Monitoring System is an android application which helps to communication between faculty and student during online classes. This application maintains student attendance, chat between faculty and students, shared study materials using database system. Student attendance are generated by class start time, and class end time interacting with video conference.

7.2 MODULE DESCRIPTION

This system after careful analysis has been identified to present itself with the following modules.

7.2.1 AUTHENTICATION MODULE

Firebase authentication is used to authenticate the user of in a very easy manner. Not only for the users but for the developers also, it provides every easy flow for the authentication and login process that is present in the almost every application. Firebase authentication using email and password. Firebase authentication system will create account with user id with email and password for firebase account reference.

Signup includes username and mobile number for user details and it will be stored in the firebase database with the user id reference. It allows user will login with email id and password credentials to firebase SDK.

It also includes change password option, if any of the user forgot password application will allow to change password using registered email id.

7.2.2 DASHBOARD MODULE

Application dashboard contains user activity modules. It allows user to locate which activity will perform based on user selection. Once user will login

it gets details about user from firebase database using user unique id of firebase authentication.

It allows user to change application settings. Dashboard contains navigation buttons for navigate separate activities or fragment through navigation option.

7.2.3 CREATE CLASS MODULE

In create class module user can create class group using class name and faculty Name, attendance type, and class icon if the user selects attendance type as ON (automatic) application will show the additional option like class start time, class end time, select days and select monitoring apps. if once user will create class application send class details to firebase database system with user id and store the data. Once class create application will generate class id and QR code for created class.

It will save class group information in the firebase database system with user id and class id with group role and creator name. class creator will act as admin for created classes.

If user select automatic attendance application will show installed application in list user will select the application for monitoring that application in class given time i.e., class start time, class end time. Application automatically monitoring on class scheduled time will class participants are active or not according to that active time application will generate attendance to the class participants.

7.2.4 JOIN CLASS MODULE

In join class module user can join class group using class Id or QR code. If class id matches with the firebase database class id application allows the user to add to the class group it will save the additionally participants uid, role,

and timestamp with the class group. If the class id did not match with the firebase database child class id it will shows the error message to the user.

7.2.5 CLASS CHAT MODULE

In class chat module user can view their created classes or joined classes in the list form. If the user selects the class it shows the class chat activity. User can chat with the classmates using chat system classmates can ask queries and doubts to their faculty member in the classes then message will send as notification with in application.

If the class active time starts it will send alert notification to all the class participants. Faculty member and class participants will chat, share notes, images, materials through the class chat modules. It stores the data in the firebase storage system and other copies of the data shown in the class chat activity.

7.2.6 CLASS MANAGE MODULE

In class manage module faculty member and class participants will see their classmate's information. Class creator and class admin only have permission to add or remove participants, make admin or remove admin to the class participants. Class creator only have the access to change class scheduling, change class name, class icon, class faculty name, and delete class option.

If the creator selects the delete class option application remove the class data from the firebase database. Admin and class participants have leave class option, if the participants select the leave class option application will remove class participants reference data will be removed in the class data in the firebase database.

7.2.7 CLASS ATTENDANCE MODULE

In the class attendance module, there are two different type of attendance system type manual mode and automatic mode. In manual mode

faculty member or class admin will give daily attendance to their class participants by viewing their class active status.

In automatic mode application will generate the daily attendance to the class participants by monitoring the selected application active time compared with class start time and class end time.

If view attendance info option will be selected application will show the attendance details of each class participants it will show the alert dialog box with “Present Days”, “Absent Days”, “Total Days”, and “Attendance Percentage” as text view option class participants does have permission to change attendance details, only admin (faculty) and class creator will have access to modify attendance data.

7.2.8 APPLICATION MANAGE MODULE

In application manage module user have the options to change application interface and settings have options to change password, forgot password, help, about application. It helps to the users, if the user selects the option change password option application will ask for current password, new password, confirm password, if the input is done application will check the current password matches with the firebase authentication data with the user unique id, if it matches and checks new password and confirm password matches the string if its equals application will update the new password in the firebase authentication data, if its not equals application shows the error message to the user.

If the user selects the forgot password options, the application allows user to recover user password using registered email id, if the email id matches with the firebase authentication data it allows user to change password option through email id.

CHAPTER - 8
TESTING

8.1 UNIT TESTING

Unit testing is a level of software testing where individual units / components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

S. NO	TEST CASE	INPUT	RESULT
1	Signup Module	abc@gmail.com	Pass
		ABC	Fail
2	Login Module	abc@123.com	Pass
		abc@123	Fail
3	Create Class Module	Correct class details	Pass
		Empty or Incorrect	Fail
4	Join Class Module	0123456789012	Pass
		class93937	Fail
		QR Image	Pass
		Other Image	Fail

8.2 SYSTEM TESTING

System Testing (ST) is a black box testing technique performed to evaluate the complete system the system's compliance against specified requirements. In System testing, the functionalities of the system are tested from an end-to-end perspective.

System Testing is usually carried out by a team that is independent of the development team in order to measure the quality of the system unbiased. It includes both functional and Non-Functional testing.

CHAPTER – 9

CONCLUSION AND FUTURE ENHANCEMENT

9.1 CONCLUSION

This online class monitoring system is the android application for easily communicate between faculty and student. The application provided facilities like chatting system, sharing study materials between classmates, and attendance system. This system can facilitate monitored through mobile. This system can help in overcoming online classes, tuition centres, institutions through mobile only. This makes the job easier and the time consumption will be less.

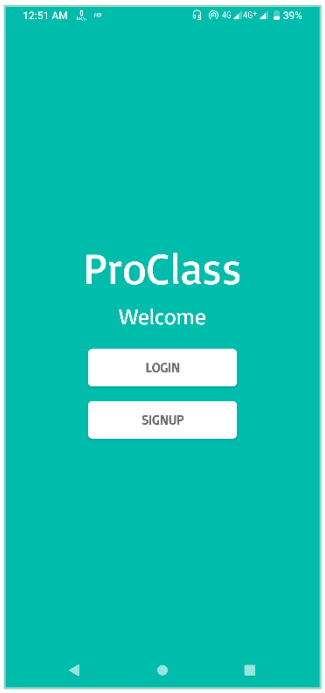
9.2 FUTURE ENHANCEMENT

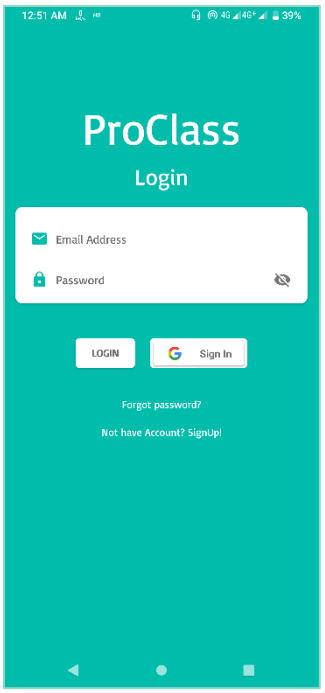
In future this application can be enhanced with video conference, cloud storage facilities, Augmented Reality classes, online exams etc., and will not only can be implemented for communication, and attendance.

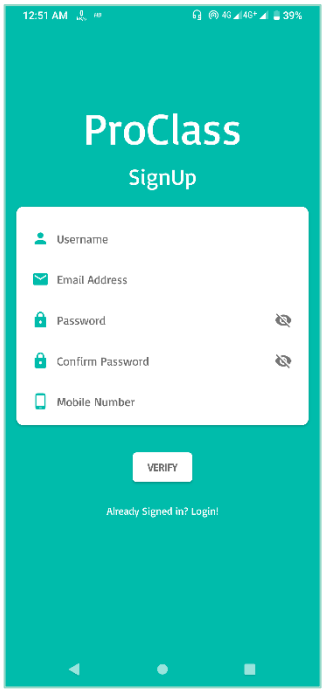
APPENDICES

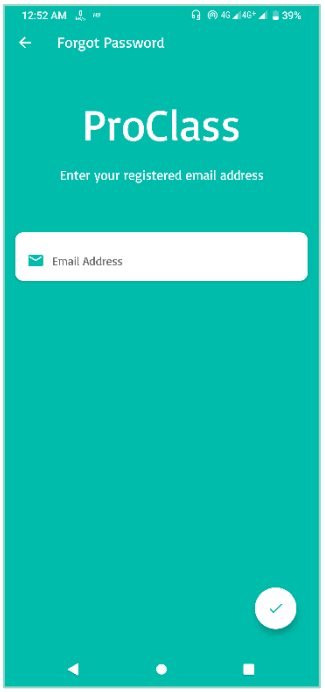
SCREENSHOTS

AUTHENTICATION MODULE

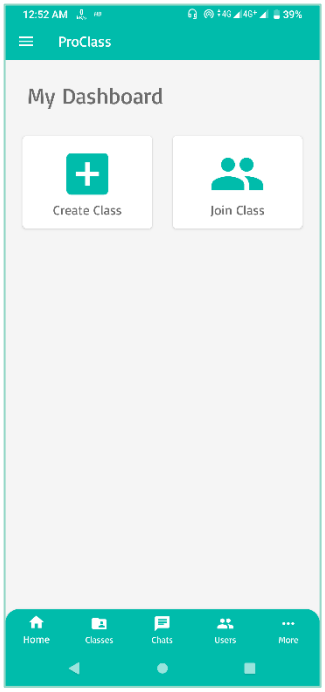


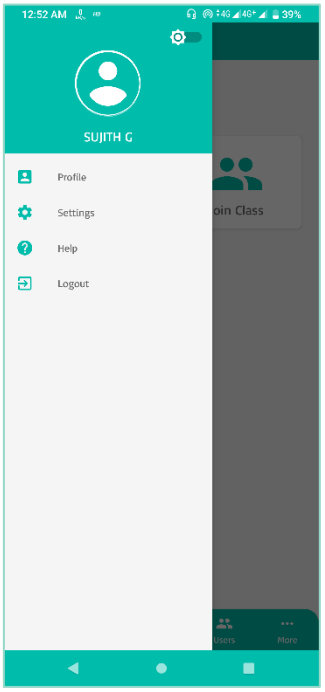




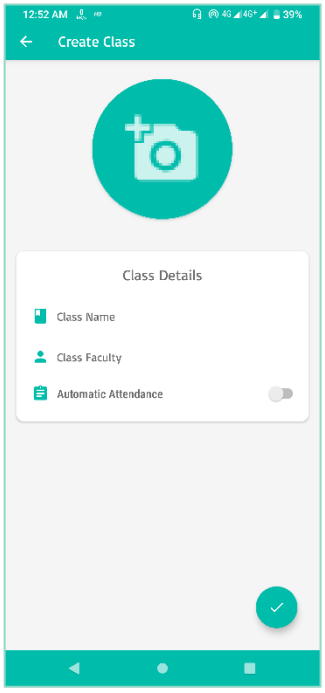


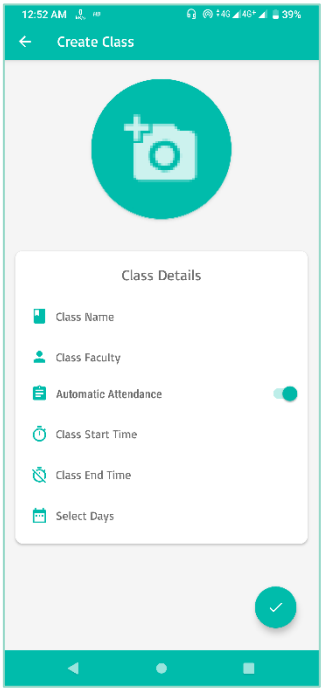
DASHBOARD MODULE



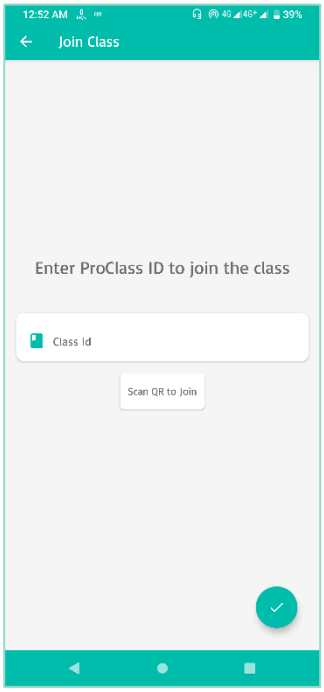


CREATE CLASS MODULE

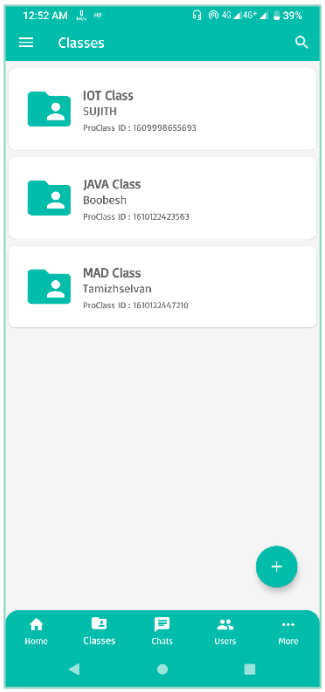


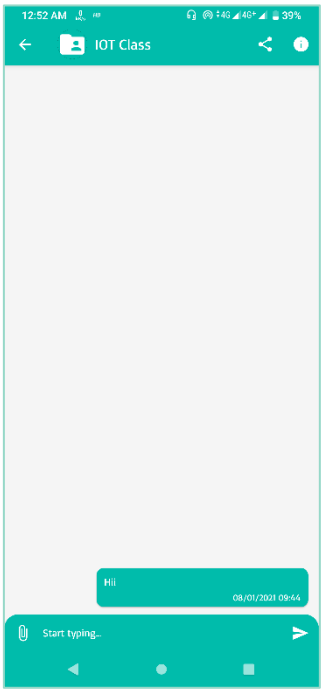


JOIN CLASS MODULE

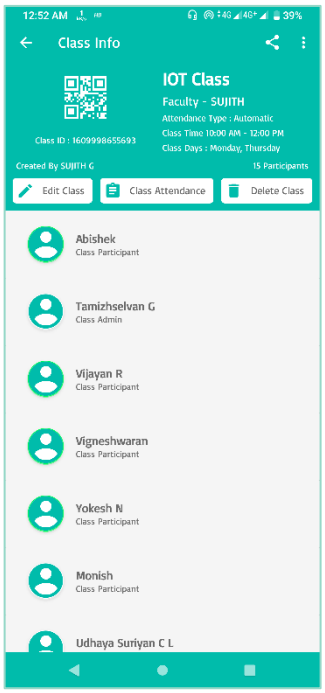


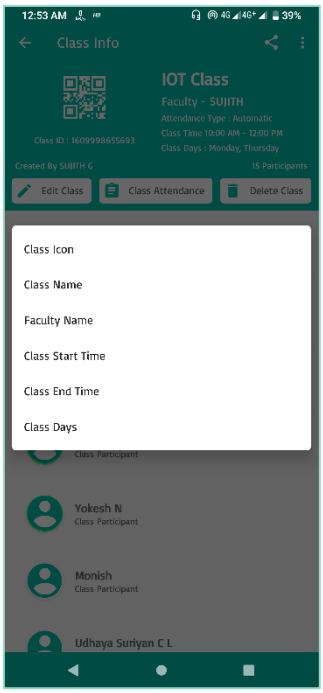
CLASS CHAT MODULE

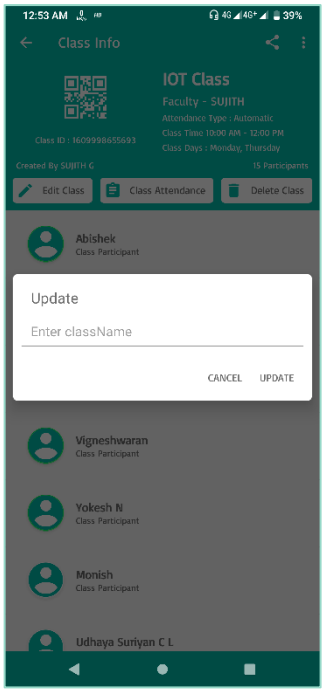


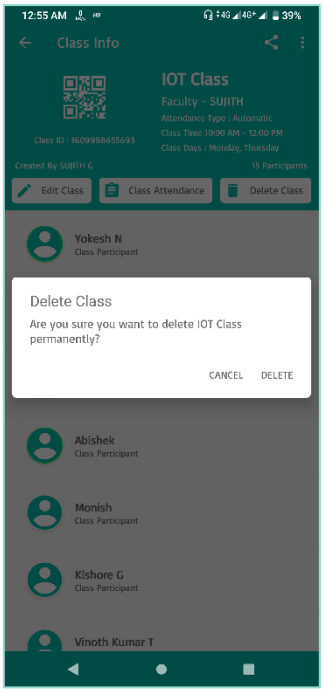


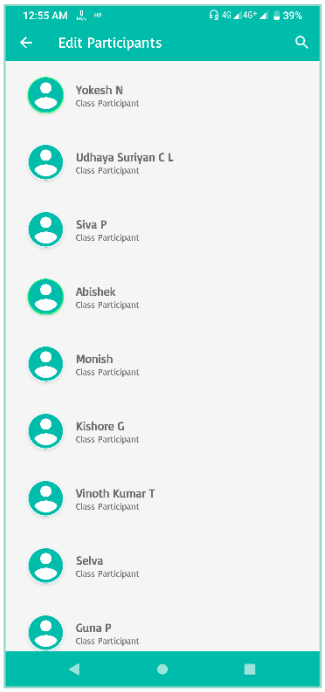
CLASS MANAGE MODULE

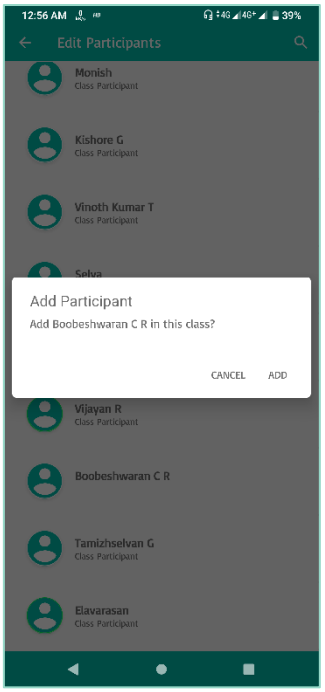


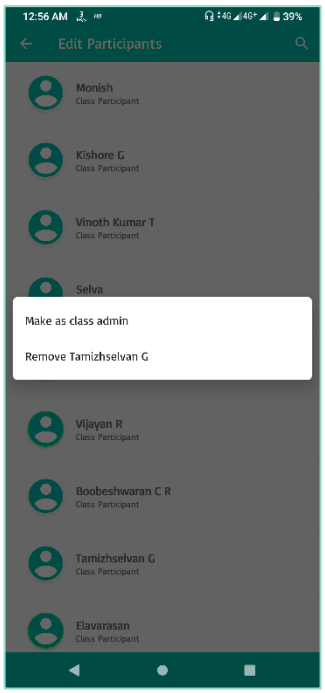


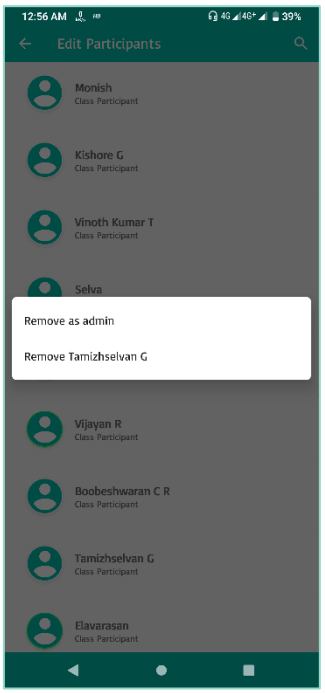


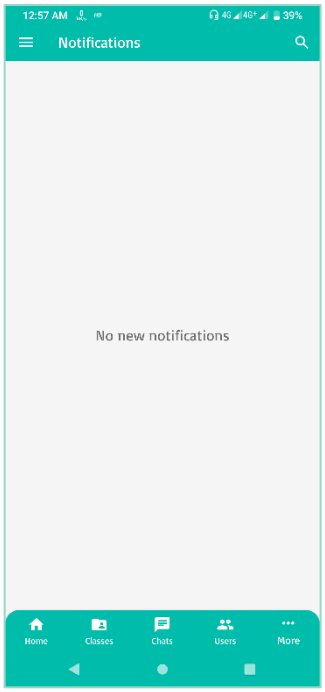


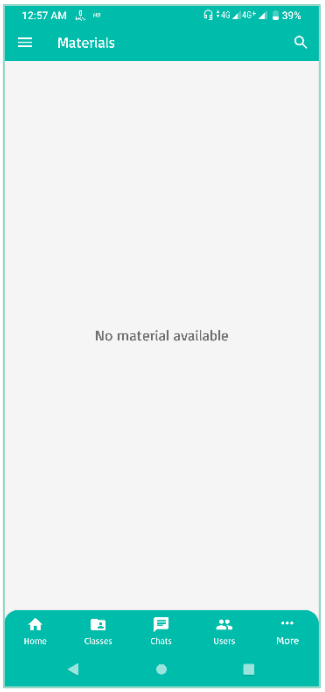

12:55 AM
39%
Edit Participants
Yokesh N
Class Participant
Udhaya Suriyan C L
Class Participant
Siva P
Class Participant
Abishek
Class Participant
Monish
Class Participant
Kishore G
Class Participant
Vinoth Kumar T
Class Participant
Selva
Class Participant
Guna P
Class Participant


12:56 AM
39%
Edit Participants
Monish
Class Participant
Kishore G
Class Participant
Vinoth Kumar T
Class Participant
Selva
Add Participant
Add Boobeshwaran C R in this class?
CANCEL
ADD
Vijayan R
Class Participant
Boobeshwaran C R
Tamizhselvan G
Class Participant
Elavarasan
Class Participant

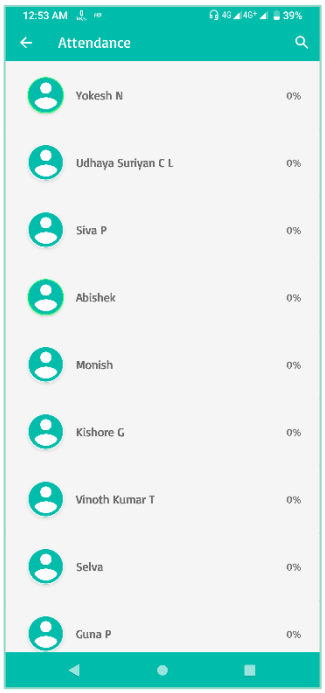

12:56 AM
39%
Edit Participants
Monish
Class Participant
Kishore G
Class Participant
Vinoth Kumar T
Class Participant
Selva
Make as class admin
Remove Tamizhselvan G
Vijayan R
Class Participant
Boobeshwaran C R
Class Participant
Tamizhselvan G
Class Participant
Elavarasan
Class Participant

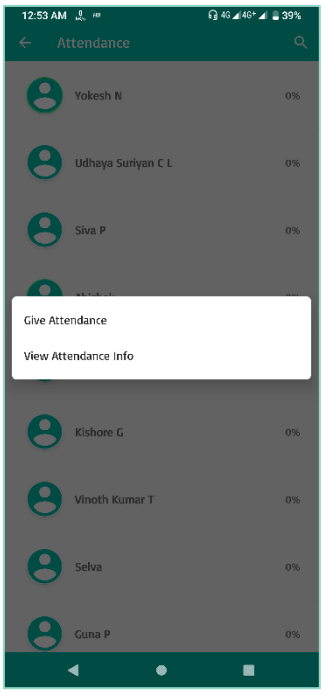

12:56 AM
39%
Edit Participants
Monish
Class Participant
Kishore G
Class Participant
Vinoth Kumar T
Class Participant
Selva
Remove as admin
Remove Tamizhselvan G
Vijayan R
Class Participant
Boobeshwaran C R
Class Participant
Tamizhselvan G
Class Participant
Elavarasan
Class Participant

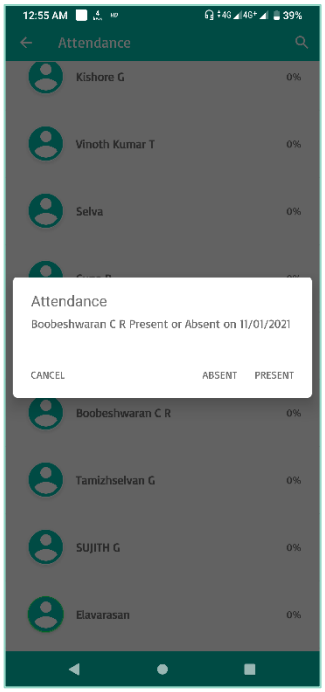


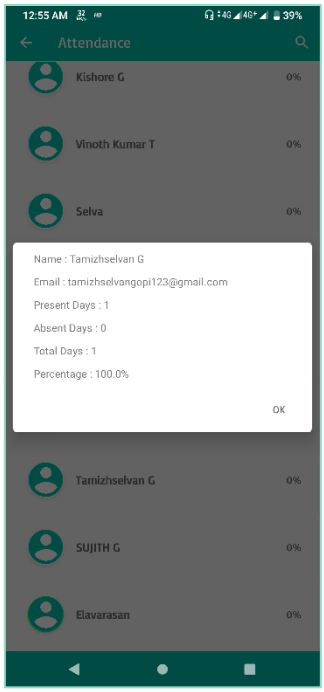


CLASS ATTENDANCE MODULE

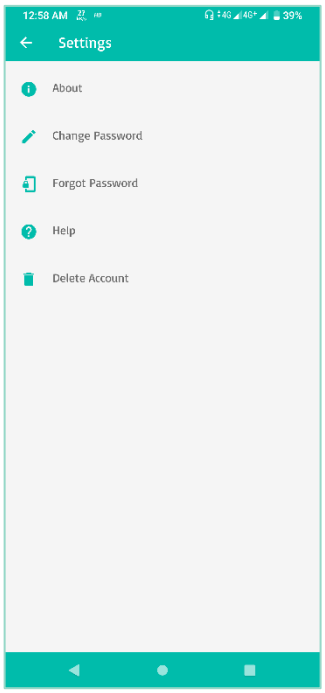


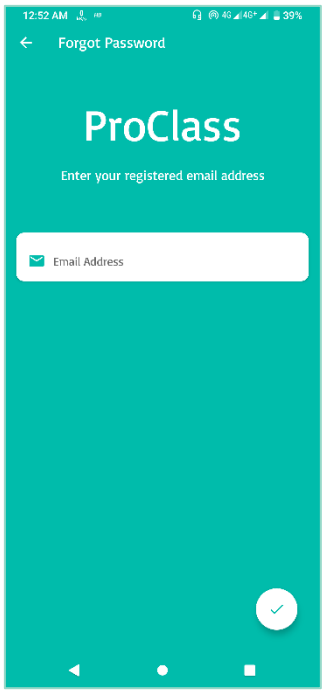


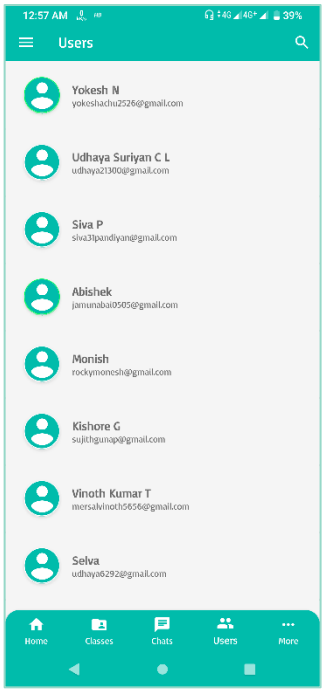


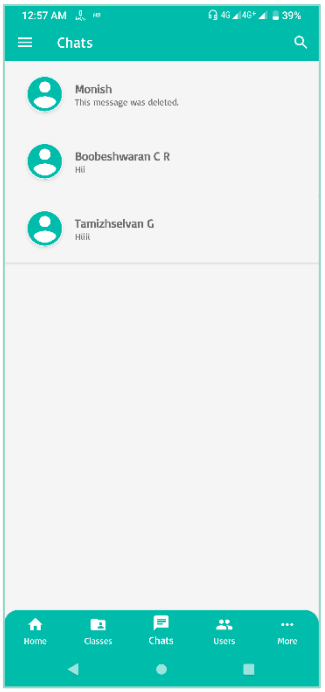


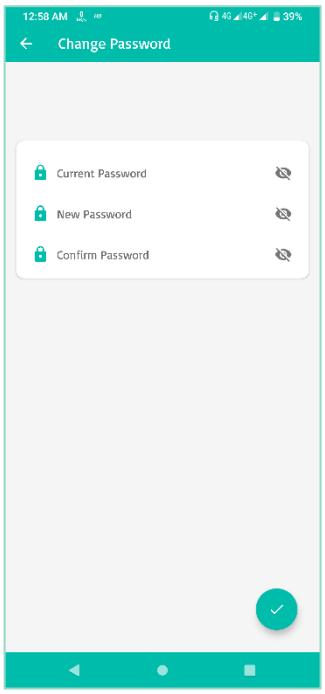
APPLICATION MANAGE MODULE

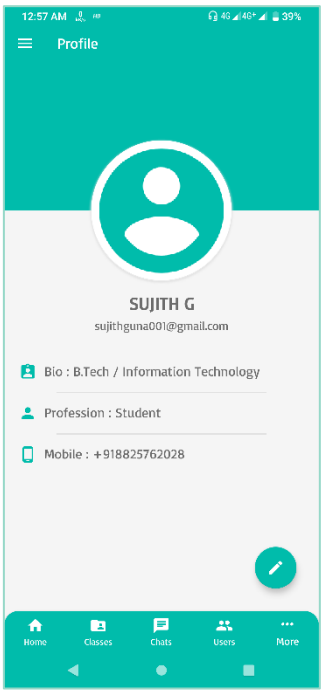


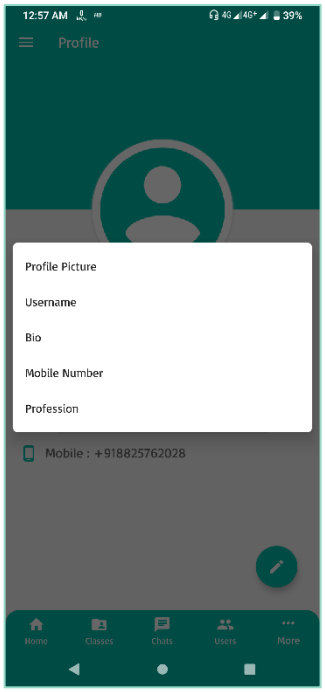


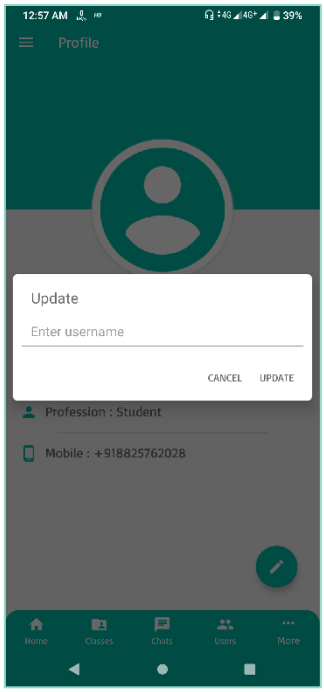

12:57 AM
39%
Users
Yokesh N
yokeshachu2526@gmail.com
Udhaya Suriyan C L
udhaya21300@gmail.com
Siva P
siva3lpandiyan@gmail.com
Abishek
jamunabai0505@gmail.com
Monish
rockymonesh@gmail.com
Kishore G
sujithgunap@gmail.com
Vinoth Kumar T
mersalvinoth5656@gmail.com
Selva
udhaya6292@gmail.com
Home
Classes
Chats
Users
More


12:57 AM
39%
Chats
Monish
This message was deleted.
Boobeshwaran C R
Hii
Tamizhselvan G
Hiiii
Home
Classes
Chats
Users
More


12:58 AM
39%
Change Password
Current Password
New Password
Confirm Password


12:57 AM
39%
Profile
SUJITH G
sujithguna001@gmail.com
Bio : B.Tech / Information Technology
Profession : Student
Mobile : +918825762028
Home
Classes
Chats
Users
More


12:57 AM
39%
Profile
Profile Picture
Username
Bio
Mobile Number
Profession
Mobile : +918825762028
Home
Classes
Chats
Users
More


12:57 AM
39%
Profile
Update
Enter username
CANCEL
UPDATE
Profession : Student
Mobile : +918825762028
Home
Classes
Chats
Users
More

SOURCE CODE

User_Login.java

```
package com.example.proclass.UserActivity;

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Patterns;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.example.proclass.R;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;

public class User_Login extends AppCompatActivity {

    private Toolbar toolbar;
    private EditText userEmailEt, userPassEt;
    private Button loginBtn, signUpBtn;
    private TextView forgotPasswordTv;
    private ProgressDialog progress Dialog;

    private FirebaseAuth mAuth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_user_login);
        setSupportActionBar(toolbar);
        setUI();

        toolbar.setNavigationOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                onBackPressed();
            }
        });

        mAuth = FirebaseAuth.getInstance();
```

```
        progressDialog = new ProgressDialog(this);
        progressDialog.setCancelable(false);

        loginBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String email = userEmailEt.getText().toString();
                String password = userPassEt.getText().toString();

                if(TextUtils.isEmpty(email) || TextUtils.isEmpty(password)) {
                    Toast.makeText(User_Login.this, "Please fill all fields",
Toast.LENGTH_SHORT).show();
                }
                else if(!Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
                    userEmailEt.setError("Invalid Email Address");
                    userPassEt.setFocusable(true);
                }
                else { loginUser(email, password); }
            }
        });

        signUpBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(User_Login.this, User_SignUp.class));
            }
        });

        forgotPasswordTv.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(User_Login.this, User_ForgotPassword.class));
            }
        });
    }

    private void loginUser(String email, String password) {

        progressDialog.setMessage("Logging In...");
        progressDialog.show();

        mAuth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task) {
                        if (task.isSuccessful()) {
                            progressDialog.dismiss();
                            FirebaseUser user = mAuth.getCurrentUser();
                            startActivity(new Intent(User_Login.this, User_Dashboard.class));
                            Toast.makeText(User_Login.this, "Login Successfully.",
Toast.LENGTH_SHORT).show();
                            finish();
                        } else {
                            progressDialog.dismiss();
```

```
                    Toast.makeText(User_Login.this, "Login failed.",
Toast.LENGTH_SHORT).show();
                }
            }
        }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            progressDialog.dismiss();
            Toast.makeText(User_Login.this, ""+e.getMessage(), Toast.LENGTH_SHORT).show();
        }
    });
  }

  private void setUI() {
    toolbar = (Toolbar) findViewById (R.id.toolbar_UL);
    userEmailEt = (EditText) findViewById (R.id.userEmailEt1);
    userPassEt = (EditText) findViewById (R.id.userPassEt1);
    forgotPasswordTv = (TextView) findViewById (R.id.forgotPassTv);
    loginBtn = (Button) findViewById (R.id.loginBtn);
    signUpBtn = (Button) findViewById (R.id.signUpBtn);
  }
}
```

PC_ClassCreate.java

```
package com.example.proclass.UserActivity.ProClassActivity;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.app.ProgressDialog;
import android.app.TimePickerDialog;
import android.content.ContentValues;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.TimePicker;
```

```
import android.widget.Toast;

import com.example.proclass.R;
import com.example.proclass.UserActivity.User_Dashboard;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;

import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;

public class PC_ClassCreate extends AppCompatActivity {

    private Toolbar toolbar;

    private TextView titleTv;
    private ImageView classIconIv;
    private EditText classNameEt, classFacultyEt, classStartTimeEt, classEndTimeEt;
    private Button dayBtn, appSelectBtn;
    private Switch attendanceTypeBtn;
    private FloatingActionButton CCDoneBtn;
    private TimePickerDialog timePickerDialog;
    private ProgressDialog progressDialog;

    private FirebaseAuth firebaseAuth;

    private static final int CAMERA_REQUEST_CODE=100;
    private static final int STORAGE_REQUEST_CODE=200;
    private static final int IMAGE_PICK_CAMERA_CODE=300;
    private static final int IMAGE_PICK_GALLERY_CODE=400;

    private String[] cameraPermissions;
    private String[] storagePermissions;
    private String[] listDays;
    private boolean[] checkedDays;
    private ArrayList<Integer> selectedDays = new ArrayList<>();

    private Uri image_uri=null;

    @Override
    protected void onCreate(final Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pc_class_create);
        setSupportActionBar(toolbar);
        setUI();

        titleTv.setText("Create Class");
```

```
        toolbar.setNavigationIcon(R.drawable.ic_back);
        toolbar.setNavigationOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                onBackPressed();
            }
        });

        firebaseAuth = FirebaseAuth.getInstance();

        progressDialog = new ProgressDialog(this);
        progressDialog.setCancelable(false);

        cameraPermissions = new String[]{Manifest.permission.CAMERA,
Manifest.permission.WRITE_EXTERNAL_STORAGE};
        storagePermissions = new String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE};

        classStartTimeEt.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                timePickerDialog = new TimePickerDialog(PC_ClassCreate.this, new
TimePickerDialog.OnTimeSetListener() {
                    @Override
                    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
                        String timeSet = "";
                        if (hourOfDay > 12) {
                            hourOfDay -= 12;
                            timeSet = "PM";
                        } else if (hourOfDay == 0) {
                            hourOfDay += 12;
                            timeSet = "AM";
                        } else if (hourOfDay == 12) {
                            timeSet = "PM";
                        } else {
                            timeSet = "AM";
                        }
                        String min = "";
                        if (minute < 10)
                            min = "0" + minute;
                        else
                            min = String.valueOf(minute);
                        String aTime = new StringBuilder().append(hourOfDay).append(':')
                                .append(min).append(" ").append(timeSet).toString();
                        classStartTimeEt.setText(aTime);
                    }
                },0 ,0,false);
                timePickerDialog.show();
            }
        });

        classEndTimeEt.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                timePickerDialog = new TimePickerDialog(PC_ClassCreate.this, new
TimePickerDialog.OnTimeSetListener() {
```

```
            @Override
            public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
                String timeSet = "";
                if (hourOfDay > 12) {
                   hourOfDay -= 12;
                   timeSet = "PM";
                } else if (hourOfDay == 0) {
                   hourOfDay += 12;
                   timeSet = "AM";
                } else if (hourOfDay == 12) {
                   timeSet = "PM";
                } else {
                   timeSet = "AM";
                }
                String min = "";
                if (minute < 10)
                   min = "0" + minute;
                else
                   min = String.valueOf(minute);
                String aTime = new StringBuilder().append(hourOfDay).append(':')
                      .append(min).append(" ").append(timeSet).toString();
                classEndTimeEt.setText(aTime);
              }
            },0 ,0,false);
          timePickerDialog.show();
        }
      });

     attendanceTypeBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
          if(attendanceTypeBtn.isChecked()){
            dayBtn.setVisibility(View.VISIBLE);
            appSelectBtn.setVisibility(View.VISIBLE);
          }
          else {
            dayBtn.setVisibility(View.GONE);
            appSelectBtn.setVisibility(View.GONE);
          }
        }
     });

      listDays = getResources().getStringArray(R.array.list_days);
      checkedDays = new boolean[listDays.length];

      dayBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
          AlertDialog.Builder builder = new AlertDialog.Builder(PC_ClassCreate.this);
          builder.setTitle("Select Days");
          builder.setMultiChoiceItems(listDays, checkedDays, new
DialogInterface.OnMultiChoiceClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int position, boolean isChecked) {
              if(isChecked){
```

```
                if(!selectedDays.contains(position)){
                    selectedDays.add(position);
                }
                else {
                    selectedDays.remove(position);
                }
            }
        }
    });
    builder.setCancelable(false);
    builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            String item = "";
            for (int i = 0; i < selectedDays.size(); i++ ){
                item = item +listDays[selectedDays.get(i)];
            }
        }
    });
    builder.setNegativeButton("cancel", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
        }
    });
    AlertDialog dialog = builder.create();
    dialog.show();
  }
});

classIconIv.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    showImagePickDialog();
  }
});

CCDoneBtn.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    startCreatingClass();
  }
});
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
  switch (requestCode){
    case CAMERA_REQUEST_CODE:{
      if(grantResults.length > 0){
        boolean cameraAccepted = grantResults[0] ==
PackageManager.PERMISSION_GRANTED;
        boolean storageAccepted = grantResults[0] ==
PackageManager.PERMISSION_GRANTED;
```

```
                if(cameraAccepted && storageAccepted){
                    pickFromCamera();
                }
                else {
                    Toast.makeText(this, "Camera & Storage permissions are required",
Toast.LENGTH_SHORT).show();
                }
            }
        }
        break;
        case STORAGE_REQUEST_CODE:{
            if(grantResults.length > 0){
                boolean storageAccepted = grantResults[0] ==
PackageManager.PERMISSION_GRANTED;
                if(storageAccepted){
                    pickFromGallery();
                }
                else {
                    Toast.makeText(this, "Storage permissions required",
Toast.LENGTH_SHORT).show();
                }
            }
        }
    }
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
  }

  @Override
  protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    if(requestCode == RESULT_OK){
        if(requestCode == IMAGE_PICK_GALLERY_CODE){
            image_uri = data.getData();
            try {
                Bitmap result Code =
MediaStore.Images.Media.getBitmap(getContentResolver(),image_uri);
                classIconIv.setImageBitmap(bitmap);
            }catch (IOException e){
                e.printStackTrace();
            }
        }
        else if(requestCode == IMAGE_PICK_CAMERA_CODE){
            classIconIv.setImageURI(image_uri);
        }
    }
    super.onActivityResult(requestCode, resultCode, data);
  }

  private void startCreatingClass() {
    progressDialog.setMessage("Creating Class");

    final String className = classNameEt.getText().toString().trim();
    final String classFaculty = classFacultyEt.getText().toString().trim();
    final String classStartTime = classStartTimeEt.getText().toString().trim();
    final String classEndTime = classEndTimeEt.getText().toString().trim();
    final String c_timestamp = ""+System.currentTimeMillis();
```

```
    if(TextUtils.isEmpty(className) || TextUtils.isEmpty(classFaculty) ||
TextUtils.isEmpty(classStartTime) || TextUtils.isEmpty(classEndTime)){
      Toast.makeText(this, "please fill all fields..", Toast.LENGTH_SHORT).show();
      return;
    }
    progressDialog.show();
    if(image_uri == null){
      createGroup(""+c_timestamp,""+className,""+classFaculty,
""+classStartTime,""+classEndTime, "");
    }
    else {
      String fileNameAndPath = "ClassData/ClassIcon" + "image" + c_timestamp;
      StorageReference storageReference =
FirebaseStorage.getInstance().getReference(fileNameAndPath);
      storageReference.putFile(image_uri)
          .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
            @Override
            public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
              Task<Uri> p_uriTask = taskSnapshot.getStorage().getDownloadUrl();
              while (!p_uriTask.isSuccessful());
              Uri p_downloadUri = p_uriTask.getResult();
              if(p_uriTask.isSuccessful()){
                createGroup(""+c_timestamp,""+className,""+classFaculty,
""+classStartTime,""+classEndTime, ""+p_downloadUri);
              }
            }
          })
          .addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
              progressDialog.dismiss();
              Toast.makeText(PC_ClassCreate.this, ""+e.getMessage(),
Toast.LENGTH_SHORT).show();
            }
          });
    }
  }
  private void createGroup(final String c_timestamp, String className, String classFaculty,
String classStartTime, String classEndTime, String classIcon) {
    final HashMap<String, String> hashMap = new HashMap<>();
    hashMap.put("classId", ""+c_timestamp);
    hashMap.put("className", ""+className);
    hashMap.put("classFaculty", ""+classFaculty);
    hashMap.put("classStartTime", ""+classStartTime);
    hashMap.put("classEndTime", ""+classEndTime);
    hashMap.put("classIcon", ""+classIcon);
    hashMap.put("timestamp", ""+c_timestamp);
    hashMap.put("createdBy", ""+firebaseAuth.getUid());

    DatabaseReference ref = FirebaseDatabase.getInstance().getReference("Classes");
    ref.child(c_timestamp).setValue(hashMap)
        .addOnSuccessListener(new OnSuccessListener<Void>() {
          @Override
          public void onSuccess(Void aVoid) {
            HashMap<String, String> hashMap1 = new HashMap<>();
```

```
                hashMap1.put("uid",firebaseAuth.getUid());
                hashMap1.put("role", "creator");
                hashMap1.put("timestamp", c_timestamp);

                DatabaseReference ref1 =
FirebaseDatabase.getInstance().getReference("Classes");
                ref1.child(c_timestamp).child("Participants").child(firebaseAuth.getUid())
                        .setValue(hashMap1)
                        .addOnSuccessListener(new OnSuccessListener<Void>() {
                            @Override
                            public void onSuccess(Void aVoid) {
                                progressDialog.dismiss();
                                Toast.makeText(PC_ClassCreate.this, "Class Created Successfully",
Toast.LENGTH_SHORT).show();
                                finish();
                                startActivity(new Intent(PC_ClassCreate.this, User_Dashboard.class));
                            }
                        })
                        .addOnFailureListener(new OnFailureListener() {
                            @Override
                            public void onFailure(@NonNull Exception e) {
                                progressDialog.dismiss();
                                Toast.makeText(PC_ClassCreate.this, ""+e.getMessage(),
Toast.LENGTH_SHORT).show();
                            }
                        });
                }
            })
            .addOnFailureListener(new OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception e) {
                    progressDialog.dismiss();
                    Toast.makeText(PC_ClassCreate.this, ""+e.getMessage(),
Toast.LENGTH_SHORT).show();
                }
            });
    }

    private void showImagePickDialog() {
        String[] options = {"Camera", "Gallery"};
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Pick Image From")
                .setItems(options, new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        if(which==0){
                            if(!checkCameraPermission()){
                                requestCameraPermissions();
                            }
                            else{
                                pickFromCamera();
                            }
                        }else {
                            if(!checkStoragePermission()){
                                requestStoragePermissions();
```

```
                }
                else {
                    pickFromGallery();
                }
            }
        }
    }).show();
}

private void pickFromGallery() {
    Intent set Items = new Intent(Intent.ACTION_PICK);
    intent.setType("image/*");
    startActivityForResult(intent, IMAGE_PICK_GALLERY_CODE);
}

private void pickFromCamera() {
    ContentValues cv = new ContentValues();
    cv.put(MediaStore.Images.Media.TITLE, "Class Icon");
    cv.put(MediaStore.Images.Media.DESCRIPTION, "Class Description");
    image_uri =
getContentResolver().insert(MediaStore.Images.Media.EXTERNAL_CONTENT_URI, cv);

    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    intent.putExtra(MediaStore.EXTRA_OUTPUT, image_uri);
    startActivityForResult(intent, IMAGE_PICK_CAMERA_CODE);
}

private boolean checkStoragePermission(){
    boolean result= ContextCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) ==
(PackageManager.PERMISSION_GRANTED);
    return result;
}

private boolean checkCameraPermission(){
    boolean result= ContextCompat.checkSelfPermission(this,
Manifest.permission.CAMERA) == (PackageManager.PERMISSION_GRANTED);
    boolean result1= ContextCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) ==
(PackageManager.PERMISSION_GRANTED);
    return result && result1;
}

private void requestStoragePermissions(){
    ActivityCompat.requestPermissions(this, storagePermissions,
STORAGE_REQUEST_CODE);
}

private void requestCameraPermissions(){
    ActivityCompat.requestPermissions(this, cameraPermissions,
CAMERA_REQUEST_CODE);
}

private void setUI() {
    classIconIv = (ImageView) findViewById (R.id.classIcon);
```

```
        classNameEt = (EditText) findViewById (R.id.classNameEt);
        classFacultyEt = (EditText) findViewById (R.id.classFacultyEt);
        toolbar = (Toolbar) findViewById (R.id.toolbar_CC);
        CCDoneBtn = (FloatingActionButton) findViewById(R.id.CCDone);
        dayBtn = (Button) findViewById (R.id.dayBtn);
        attendanceTypeBtn = (Switch) findViewById(R.id.attendanceTypeBtn);
        appSelectBtn = (Button) findViewById (R.id.appSelectBtn);
        titleTv = (TextView) findViewById (R.id.actionBarTitle);
        classStartTimeEt = (EditText) findViewById (R.id.classStartTimeEt);
        classEndTimeEt = (EditText) findViewById (R.id.classEndTimeEt);
    }
}
```

PC_ClassJoin.java

```
package com.example.proclass.UserActivity.ProClassActivity;

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.example.proclass.R;
import com.example.proclass.UserActivity.User_Dashboard;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.Query;
import com.google.firebase.database.ValueEventListener;

import java.util.HashMap;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;

public class PC_ClassJoin extends AppCompatActivity {

    private Toolbar Ridden;

    private TextView title;
    private EditText classIdEt;
    private FloatingActionButton joinClassBtn;
    private ProgressDialog progressDialog;
```

```
    private FirebaseAuth firebaseAuth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pc_class_join);
        setSupportActionBar(toolbar);
        setUI();

        title.setText("Join Class");
        toolbar.setNavigationIcon(R.drawable.ic_back);
        toolbar.setNavigationOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                onBackPressed();
            }
        });

        progressDialog = new ProgressDialog(this);
        progressDialog.setCancelable(false);

        joinClassBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                String classId = classIdEt.getText().toString().trim();
                if(TextUtils.isEmpty(classId)){
                    Toast.makeText(PC_ClassJoin.this, "Please enter classId.",
Toast.LENGTH_SHORT).show();
                    classIdEt.setError("Please enter classId");
                    classIdEt.setFocusable(true);
                }
                else {
                    joinClass();
                    Toast.makeText(PC_ClassJoin.this, "Later you will added to this class",
Toast.LENGTH_SHORT).show();
                }
            }
        });

    }

    private void joinClass() {
        progressDialog.show();
        int presentDays = 0, absentDays = 0 , totalDays = 0;
        float percentage = 0;
        String classIdJoin = classIdEt.getText().toString();
        String timestamp = ""+System.currentTimeMillis();
        HashMap<String, String> hashMap1 = new HashMap<>();
        hashMap1.put("uid",firebaseAuth.getUid());
        hashMap1.put("role", "participant");
        hashMap1.put("timestamp", ""+timestamp);
        hashMap1.put("presentDays", ""+presentDays);
        hashMap1.put("absentDays", ""+absentDays);
        hashMap1.put("totalDays", ""+totalDays);
```

```
        hashMap1.put("percentage", ""+percentage);

        DatabaseReference ref1 = FirebaseDatabase.getInstance().getReference("Classes");
        ref1.child(classIdJoin).child("Participants")
                .setValue(hashMap1)
                .addOnSuccessListener(new OnSuccessListener<Void>() {
                    @Override
                    public void onSuccess(Void aVoid) {
                        progressDialog.dismiss();
                        Toast.makeText(PC_ClassJoin.this, "Class Created Successfully",
Toast.LENGTH_SHORT).show();
                        finish();
                        startActivity(new Intent(PC_ClassJoin.this, User_Dashboard.class));
                    }
                })
                .addOnFailureListener(new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception e) {
                        progressDialog.dismiss();
                        Toast.makeText(PC_ClassJoin.this, ""+e.getMessage(),
Toast.LENGTH_SHORT).show();
                    }
                });
    }

    private void setUI() {
        toolbar = (Toolbar) findViewById(R.id.toolbar_JC);
        classIdEt = (EditText) findViewById(R.id.classIdEt);
        joinClassBtn = (FloatingActionButton) findViewById(R.id.joinClassBtn);
        title = (TextView) findViewById(R.id.actionBarTitle);
    }
}
```

PC_ClassAttendanceList.java

```
package com.example.proclass.UserActivity.ProClassActivity;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

import com.example.proclass.R;
import com.example.proclass.UserActivity.Adapter.AdapterAttendanceList;
import com.example.proclass.UserActivity.Models.ModelUsersData;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
```

```
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.util.ArrayList;
import java.util.List;

public class PC_ClassAttendanceList extends AppCompatActivity {

    private AdapterAttendanceList adapterAttendanceList;
    private List<ModelUsersData> userList;

    private Toolbar toolbar;
    private RecyclerView attendanceListRv;

    private String classId, myClassRole;

    private TextView title;

    private FirebaseAuth Ridotto;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pc_class_attendance_list);
        setSupportActionBar(toolbar);
        setUI();

        title.setText("Attendance");
        toolbar.setNavigationOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                onBackPressed();
            }
        });

        Intent intent = getIntent();
        classId = intent.getStringExtra("classId");

        firebaseAuth = FirebaseAuth.getInstance();

        attendanceListRv = findViewById(R.id.attendanceListRv);
        attendanceListRv.setHasFixedSize(true);
        attendanceListRv.setLayoutManager(new
LinearLayoutManager(PC_ClassAttendanceList.this));
        userList = new ArrayList<>();

        loadClassInfo();
        loadMyClassRole();
        loadClassUsers();
    }

    private void loadClassInfo() {

        DatabaseReference ref = FirebaseDatabase.getInstance().getReference("Classes");
```

```
        ref.orderByChild("classId").equalTo(classId)
                .addValueEventListener(new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot snapshot) {
                        for (DataSnapshot ds: snapshot.getChildren()){
                            String classId = ""+ds.child("classId").getValue();
                            String className = ""+ds.child("className").getValue();
                            String classFaculty = ""+ds.child("classFaculty").getValue();
                            String timestamp = ""+ds.child("timestamp").getValue();
                            String classIcon = ""+ds.child("classIcon").getValue();
                            String createdBy = ""+ds.child("classFaculty").getValue();
                        }
                    }
                    @Override
                    public void onCancelled(@NonNull DatabaseError error) {}
                });
    }

    private void loadMyClassRole() {

        DatabaseReference ref = FirebaseDatabase.getInstance().getReference("Classes");
        ref.child(classId).child("Participants")
                .orderByChild("uid").equalTo(firebaseAuth.getUid())
                .addValueEventListener(new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot snapshot) {
                        for(DataSnapshot ds:snapshot.getChildren()){
                            myClassRole = ""+ds.child("role").getValue();
                        }
                    }
                    @Override
                    public void onCancelled(@NonNull DatabaseError error) {}
                });
    }

    private void loadClassUsers() {

        userList = new ArrayList<>();
        DatabaseReference ref = FirebaseDatabase.getInstance().getReference("Classes");
        ref.child(classId).child("Participants").addValueEventListener(new ValueEventListener()
{
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                userList.clear();
                for(DataSnapshot ds: snapshot.getChildren()){
                    String uid = ""+ds.child("uid").getValue();
                    final String hisPreviousRole = ""+ds.child(uid).child("role").getValue();
                    DatabaseReference ref = FirebaseDatabase.getInstance().getReference("Users");
                    ref.orderByChild("uid").equalTo(uid).addValueEventListener(new
ValueEventListener() {
                        @Override
                        public void onDataChange(@NonNull DataSnapshot snapshot) {
                            for(DataSnapshot ds: snapshot.getChildren()) {
                                ModelUsersData modelUsersData = ds.getValue(ModelUsersData.class);
                                userList.add(modelUsersData);
```

```
                    adapterAttendanceList = new
AdapterAttendanceList(PC_ClassAttendanceList.this, userList, classId, myClassRole);
                    attendanceListRv.setAdapter(adapterAttendanceList);
                }
            }
            @Override
            public void onCancelled(@NonNull DatabaseError error) {}
        });
      }
    }
    @Override
    public void onCancelled(@NonNull DatabaseError error) {}
  });
 }

 private void setUI() {
  toolbar = (Toolbar) findViewById (R.id.toolbar_AA);
  title = (TextView) findViewById (R.id.actionBarTitle);
  attendanceListRv = (RecyclerView) findViewById (R.id.attendanceListRv);
 }
}
```

ModelUsersData.java

```
package com.example.proclass.UserActivity.Models;

public class ModelUsersData {

 String email, image, username, uid, role, attendance;

 public ModelUsersData() {
 }

 public ModelUsersData(String email, String image, String username, String uid, String
role) {
  this.email = email;
  this.image = image;
  this.username = username;
  this.uid = uid;
  this.role = role;
 }

 public String getEmail() {
  return email;
 }

 public void setEmail(String email) {
  this.email = email;
 }

 public String getImage() {
  return image;
 }
```

```
    public void setImage(String image) {
        this.image = image;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }
    public String getUid() {
        return uid;
    }

    public void setUid(String uid) {
        this.uid = uid;
    }

    public String getRole() {
        return role;
    }

    public void setRole(String role) {
        this.role = role;
    }
}
```

ModelClassData.java

```
package com.example.proclass.UserActivity.Models;

public class ModelClassData {
    String classFaculty, classIcon, classId, className, classStartTime, classEndTime,
createdBy, timestamp;

    public ModelClassData() {
    }

    public ModelClassData(String classFaculty, String classIcon, String classId, String
className, String classStartTime, String classEndTime, String createdBy, String timestamp) {
        this.classFaculty = classFaculty;
        this.classIcon = classIcon;
        this.classId = classId;
        this.className = className;
        this.classStartTime = classStartTime;
        this.classEndTime = classEndTime;
        this.createdBy = createdBy;
        this.timestamp = timestamp;
    }

    public String getClassFaculty() {
        return classFaculty;
    }
```

```
public void setClassFaculty(String classFaculty) {
   this.classFaculty = classFaculty;
}

public String getClassIcon() {
   return classIcon;
}

public void setClassIcon(String classIcon) {
   this.classIcon = classIcon;
}
public String getClassId() {
   return classId;
}

public void setClassId(String classId) {
   this.classId = classId;
}

public String getClassName() {
   return className;
}

public void setClassName(String className) {
   this.className = className;
}

public String getClassStartTime() {
   return classStartTime;
}

public void setClassStartTime(String classStartTime) {
   this.classStartTime = classStartTime;
}

public String getClassEndTime() {
   return classEndTime;
}

public void setClassEndTime(String classEndTime) {
   this.classEndTime = classEndTime;
}

public String getCreatedBy() {
   return createdBy;
}

public void setCreatedBy(String createdBy) {
   this.createdBy = createdBy;
}

public String getTimestamp() {
   return timestamp;
}
```

```
    public void setTimestamp(String timestamp) {
        this.timestamp = timestamp;
    }
}
```

ModelAttendanceData.java

```
package com.example.proclass.UserActivity.Models;

public class ModelAttendanceData {

    String classId, presentDays, absentDays, totalDays, percentage;

    public ModelAttendanceData() {
    }

    public ModelAttendanceData(String classId, String presentDays, String absentDays, String
totalDays, String percentage) {
        this.classId = classId;
        this.presentDays = presentDays;
        this.absentDays = absentDays;
        this.totalDays = totalDays;
        this.percentage = percentage;
    }

    public String getClassId() {
        return classId;
    }

    public void setClassId(String classId) {
        this.classId = classId;
    }

    public String getPresentDays() {
        return presentDays;
    }

    public void setPresentDays(String presentDays) {
        this.presentDays = presentDays;
    }

    public String getAbsentDays() {
        return absentDays;
    }

    public void setAbsentDays(String absentDays) {
        this.absentDays = absentDays;
    }

    public String getTotalDays() {
        return totalDays;
    }
```

```
public void setTotalDays(String totalDays) {
   this.totalDays = totalDays;
}

public String getPercentage() {
   return percentage;
}
```

REFERENCES

[1] Aruna Bhat, "Face Recognition Using Eigen and Fisher Faces". (2017) International Journal of Soft Computing, Mathematics and Control (IJSCMC), Vol. 2, No. 3.

[2] Badal J. Deshmukh, Sudhir, M. Kharad, (2019) "Efficient Attendance Management", Volume 1 Issue 1.

[3] Deepak Ghimire and Joonwhoan Lee. (2017) "Face Detection Method Based on Skin Color and Edges", J Inf Process Syst, Vol.9, No.1.

[4] Faudzi, N. Yahya. (2018) “Evaluation of face recognition techniques”, 5th International Conference on Intelligent and Advanced Systems (ICCIA), Volume 1, Issue 2, pp.1-6.

[5] J. Chatrath, P. Gupta, P. Ahuja, A. Goel. (2018) “Real Time Human Face Detection”, International Conference on Signal Processing and Integrated Networks (SPIN) Volume 1, Issue 1, pp. 705-710.

[6] Jagadeesh H S, Suresh Babu K, and Raja K B. (2018) "SOM Based Face Recognition", Signal & Image Processing: An International Journal (SIPIJ) Vol.3, No.2.

[7] Jobin J., Jiji Joseph, Sandhya Y.A, Soni P. Saji, Deepa P.L... (2018) “Palm Biometrics Recognition and Verification System”. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering Vol. 1, Issue 2.

[8] Kandia Arora, "Real Time Application of Face Recognition Concept". (2018) International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231- 2307, Volume-2, Issue-5.

[9] Omaima N. A. AL-Allaf. (2018) “Review of Face Detection Systems Based Artificial Intelligence Algorithms” The International Journal of Multimedia & Its Applications (Ijma) Vol.6, No.1.

[10] Paul Viola, Michael J. Jones. (2019) “Robust Real-Time Face Detection” International Journal of Computer Vision Volume 57 Issue 2, pp. 137 – 154.

[11] R N Daschoudhary, Rajashree Tripathy. (2019) “Real-time Face Detection and Tracking Using Multimodal Density Model” International Journal of Electronics and Computer Science Engineering Volume 3, Issue 2, pp.175-184.

[12] Sapna Vishwakarma, Prof. Krishan Kant Pathak. (2018) “Face Recognition using DCT Coefficient Vectors”, International Journal of Engineering Trends and Technology (IJETT), Volume 9, Issue 2, pp.96-100.