

# CHAPTER 1

## 1. INTRODUCTION

### 1.1 MOTIVATION

Technological advancements in this era of digitization along with being a boon to the world have been advantageous to the educational sector too. As we discussed earlier that manual maintenance of a Time Table Management System is a tedious job. So, to enhance the ease of working we go for this package.

Manual maintenance of databases of items, time table processing is a time taking process and somehow erroneous. To give more accuracy to the system i.e., rather going manual modification we involve computer for accuracy. The least but most important it saves time.

### 1.2 PROPOSED SYSTEM

The proposed System is completely computer-based application. In the proposed system administrator should not to worry about their late and improper management of sales details. All the information will be available by just clicking on a single button. Thousands of records can search and displayed without taking any significant time.

Some of the advantages are:

- Saves Time and Effort.
- Reduces Error
- Easy customization
- Instant Notifications for changes

### 1.3 EXISTING SYSTEM

The preparation of the timetable for the university is done by a timetable committee set-up by various schools (faculties) including the e-exam centre which serve as a unit on its own. Members of the timetable committee are mostly exam officers from different departments. The timetable is automatically generated after the timetable committee successfully allocate time slots, space, venue, lecturers, etc. considering the number of the students for each event (lecture/test/exam) and the volume capacity of the venues. Then it is manually compiled and distributed in paper format. Each department then extract courses that are relevant from the school (faculty) timetable. After the extraction and production of the department's version of the timetable, the exam officers then endorse the timetable and seek the Head of Department's approval for publishing the timetable.

## **CHAPTER 2**

### **2. REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION**

#### **2.1 SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT**

- OS: Android 8 or High
- Memory: 8GB RAM
- Free Storage: 2GB
- Android Studio
- Android Software Development Kit
- Java Development Kit
- The SDK and AVD Manager

#### **2.2 VALIDATION**

In this application, when admin tries to login with username and password, it must be valid username and password. When student or teacher tries to login, they should use the username and password given by the admin.

When faculties add the timetable for the subject first time then the timetable will uploaded successfully, but if the timetable is already exist in the database and that is uploaded by the same faculty then the timetable get updated in the database and displayed to both faculties and students.

## CHAPTER 3

### 3. SYSTEM DESIGN

#### 3.1 DESIGN APPROACH

##### Use-Case Diagram

A use case diagram is used to represent the dynamic behaviour of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

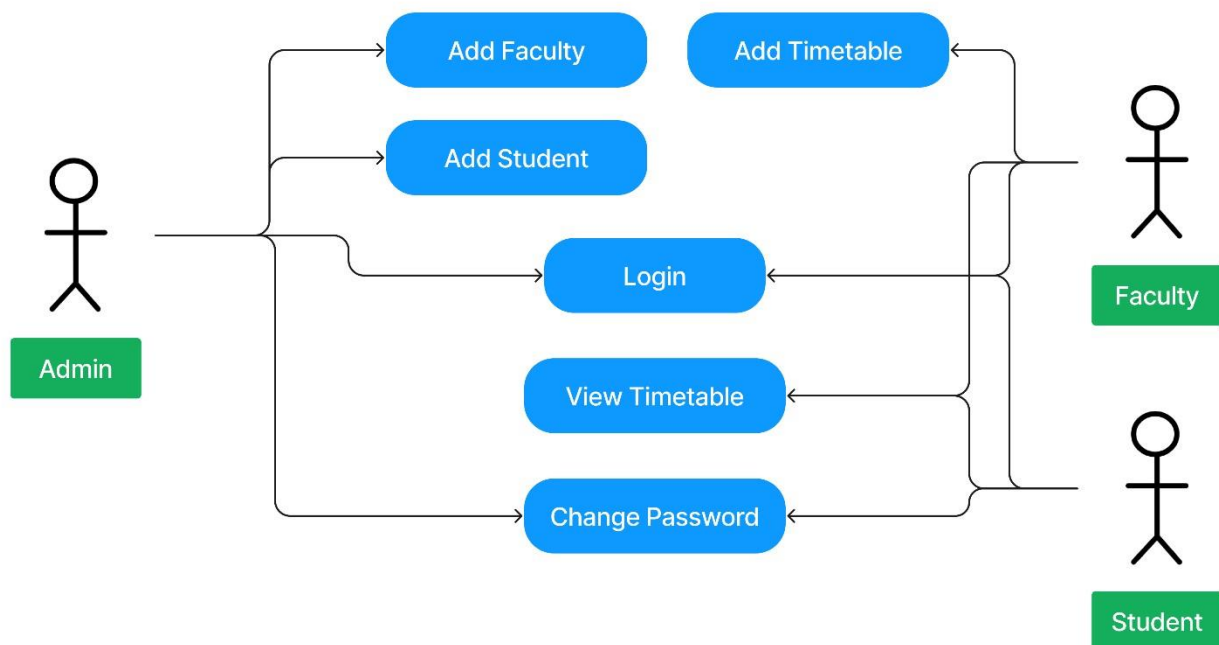


Fig 3.1.1 Use-Case Diagram

Above figure represents Use-Case diagram of the Timetable management system. In this there is three modules Admin, Faculty and Student where admin login to the and he/she can add the faculties and students and he can also change the password of their account. For Faculties they can login to the system, they can view their account and they can add and view the timetable which they have added. Student's module is used by the admin registered student where they can login to the system to view the timetable of the subjects and if they want, they can al so change the password of their account.

## 3.2 LAYOUT DESIGN

### Home Screen

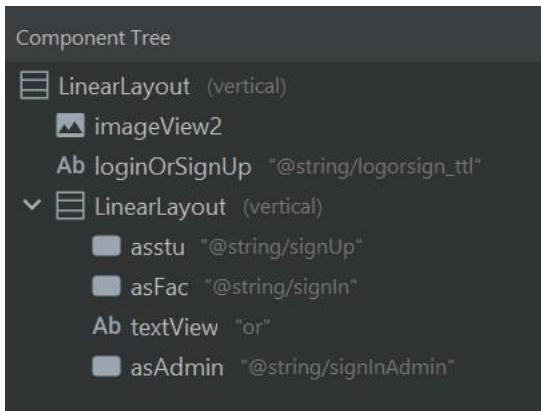


Fig 3.2.1: In Home screen there is a 3 Button where user can login as a student, faculty or as admin

### Student Registration Screen

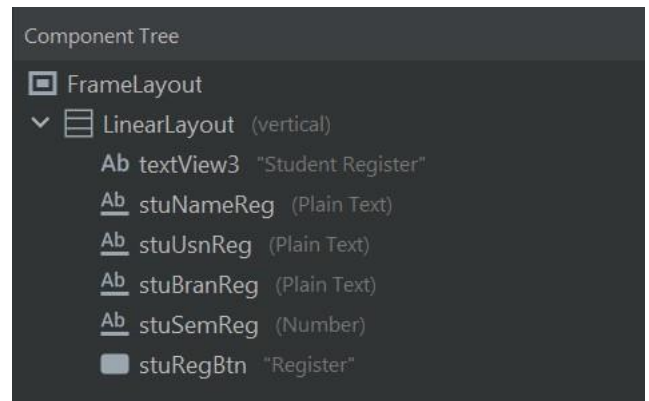


Fig 3.2.2: In this screen the admin can add the student by entering student's usn, name, sem etc.

### Faculty Registration Screen

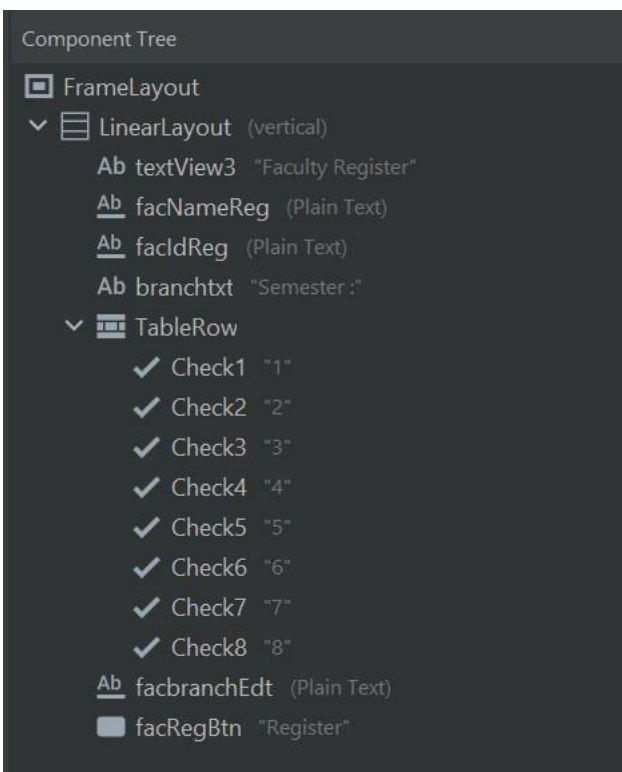


Fig 3.2.3: In this screen the admin will add the faculty by entering their information

### Add Timetable Screen

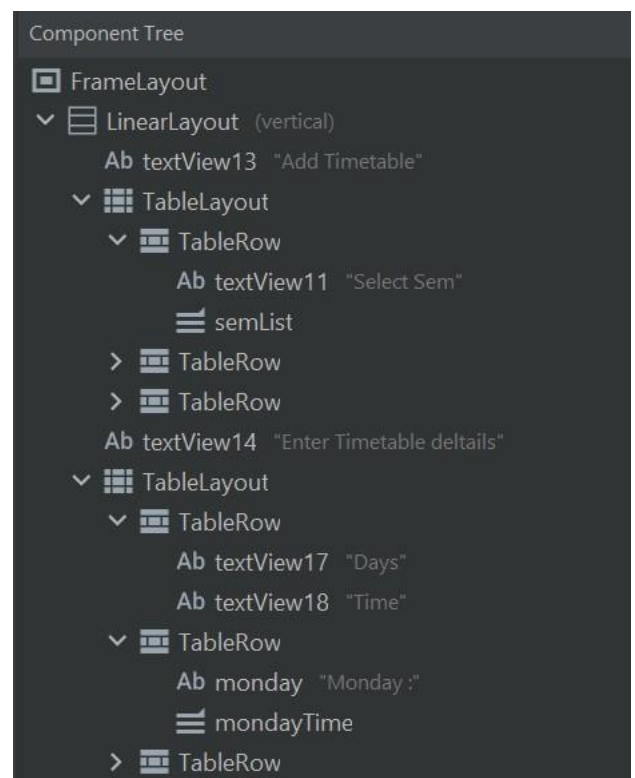


Fig 3.2.4: In this screen the faculty will add the time table of their subject with weekly timetable.

## Student Dashboard Screen

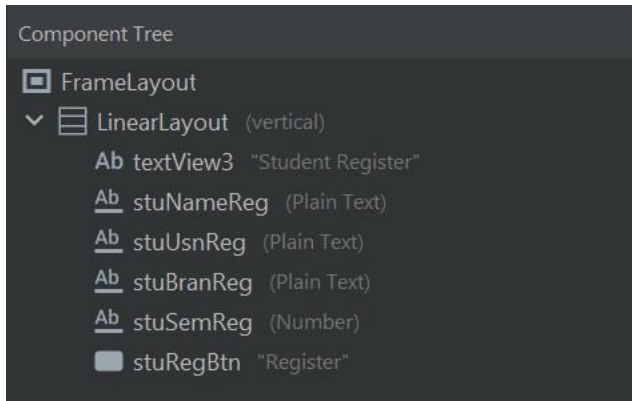


Fig 3.2.5: In student dashboard screen the student will login and they can see the timetable of the subjects.

## Faculty View

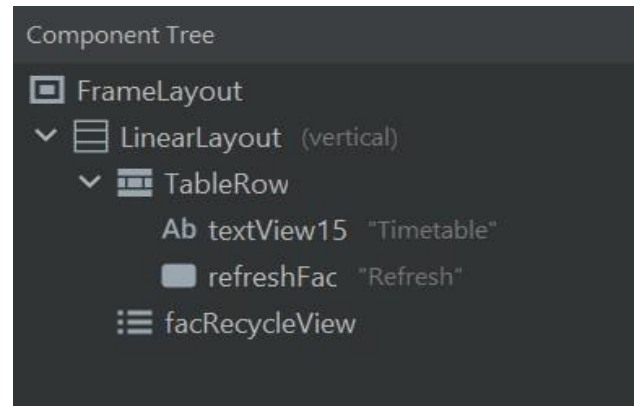


Fig 3.2.6: Faculty will login to the system and they can view the timetable which they have added.

## Student Profile Screen

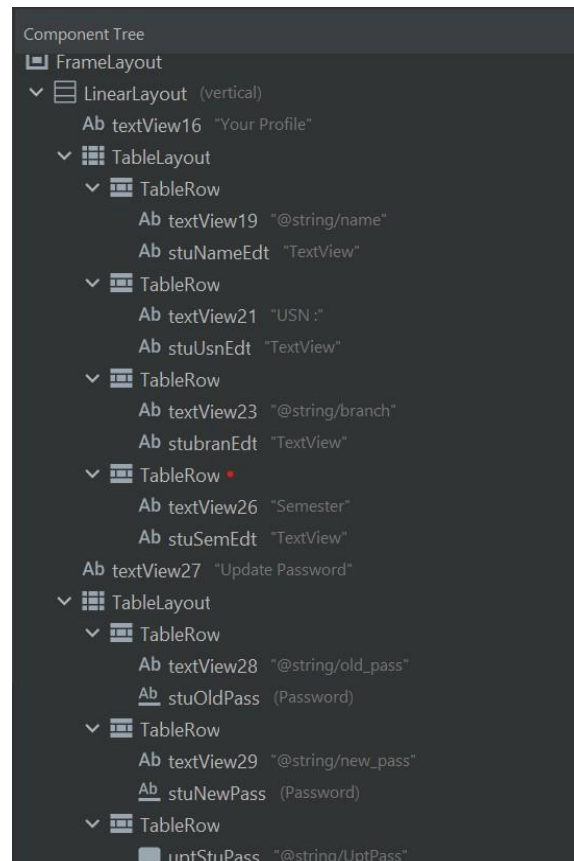


Fig 3.2.7: This screen is for students to see their details and to update their password.

## CHAPTER 4

### 4. IMPLEMENTATION

#### 4.1 INTRODUCTION TO PROGRAMMING LANGUAGES, IDE's, TOOLS AND TECHNOLOGIES

##### Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA. On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps.

##### Android Software Development Kit

The Android SDK (software development kit) is a set of development tools used to develop applications for the Android platform. It's a set of software tools and programs used by developers to create applications for specific platforms. The Android SDK includes the following: Required libraries, Debugger.

##### Java Development Kit

Java is one of the most popular coding languages out there, and Java Development Kit (or JDK) is its official development package. Currently, Java Development Kit is also one of the most popular development environments in which to code Java. The Java Development Kit offers a wide range of practical tools like javac. You also have java and jdb which works as the debugger of the system.

##### The SDK and AVD Manager

The sdk manager is a command line tool that allows you to view, install, update, and uninstall packages for the Android SDK. The avd manager is a command line tool that allows you to create and manage Android Virtual Devices (AVDs) from the command line. An AVD lets you define the characteristics of an Android handset, Wear OS watch, or Android TV device that you want to simulate in the Android Emulator.

##### NodeJS

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser, which was designed to build scalable network applications. Node.js lets developers use JavaScript to write command line tools and for server-side scripting running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web application for

development around a single programming language, rather than different languages for server side and client-side scripts.

## Express Web Framework

Express is a popular unopinionated web framework, written in JavaScript and hosted within the Node.js runtime environment. This module explains some of the key benefits of the framework, how to set up your development environment and how to perform common web development and deployment tasks.

## MongoDB

MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server-Side Public License which is deemed non-free by several distributions

## 4.2 IMPLEMENTATION

### Student Registration

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                        Bundle savedInstanceState) {
    Activity activityObj = this.getActivity();
    View myView =inflater.inflate(R.layout.fragment_student_reg, container, false);
    retrofit = new
Retrofit.Builder().baseUrl(BASE_URL).addConverterFactory(GsonConverterFactory.create()).b
uild();
    retrofitInterface = retrofit.create(RetrofitInterface.class);
    regStudent =(Button) myView.findViewById(R.id.stuRegBtn);
    stuNameEd =(EditText) myView.findViewById(R.id.stuNameReg);
    stuUsnEd =(EditText)myView.findViewById(R.id.stuUsnReg);
    stuBraEdt =(EditText)myView.findViewById(R.id.stuBranReg);
    stuSemEdt =(EditText)myView.findViewById(R.id.stuSemReg);
    regStudent.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            HashMap<String,String> map = new HashMap<>();
            map.put("name",stuNameEd.getText().toString());
            map.put("usn",stuUsnEd.getText().toString());
            map.put("sem",stuSemEdt.getText().toString());
            map.put("branch",stuBraEdt.getText().toString());
            Call<Void> call = retrofitInterface.executeSignup(map);
            call.enqueue(new Callback<Void>() {
                @Override
                public void onResponse(Call<Void> call, Response<Void> response) {
                    if(response.code() == 200){
                        Toast.makeText(activityObj, "Registration Successfull",
Toast.LENGTH_SHORT).show();
                        resetfields();
                    }
                    else if(response.code() == 400){
```

```

        Toast.makeText(activityObj, "Already registered",
Toast.LENGTH_SHORT).show();
    }
}
@Override
public void onFailure(Call<Void> call, Throwable t) {
    Toast.makeText(activityObj, t.getMessage(),
Toast.LENGTH_SHORT).show();
}
});
}
});
return myView;
}

```

## Add Timetable

```

addcls.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        HashMap<String,String> map = new HashMap<>();
        map.put("fid",fid);
        map.put("branch",branch);
        map.put("subcode",subCodeEdt.getText().toString());
        map.put("subname",subNameEdt.getSelectedItem().toString());
        map.put("sem",semDisp.getSelectedItem().toString());
        map.put("mon",mon.getSelectedItem().toString());
        map.put("tue",tue.getSelectedItem().toString());
        map.put("wed",wed.getSelectedItem().toString());
        map.put("thu",thu.getSelectedItem().toString());
        map.put("fri",fri.getSelectedItem().toString());
        map.put("sat",sat.getSelectedItem().toString());
        Call<Void> call = retrofitInterface.exefacAddCls(map);
        call.enqueue(new Callback<Void>() {
            @Override
            public void onResponse(Call<Void> call, Response<Void> response) {
                if(response.code() == 200){
                    Toast.makeText(activityObj, "TimeTable added successfully",
Toast.LENGTH_SHORT).show();
                }
                else if(response.code() == 201){
                    Toast.makeText(activityObj, "TimeTable updated successfully",
Toast.LENGTH_SHORT).show();
                }
                else if(response.code() == 400){
                    Toast.makeText(activityObj, "Timetable exists",
Toast.LENGTH_SHORT).show();
                }
                else if(response.code() == 403){
                    Toast.makeText(activityObj, "Subject Time table already added
by others", Toast.LENGTH_SHORT).show();
                }
                else{
                    Toast.makeText(activityObj, response.message(),
Toast.LENGTH_SHORT).show();
                }
            }
            @Override
            public void onFailure(Call<Void> call, Throwable t) {
                Toast.makeText(activityObj, t.getMessage(),
Toast.LENGTH_SHORT).show();
            }
        });
    }
}

```



## Timetable View

```

public class Recycle_ttl_view extends RecyclerView.Adapter<Recycle_ttl_view.MyViewHolder>
{
    Context context;
    ArrayList<ModClass> dataList;
    public Recycle_ttl_view(Context context, ArrayList<ModClass> dataList) {
        this.context = context;
        this.dataList = dataList;
    }
    @NonNull
    @Override
    public Recycle_ttl_view.MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent,
int viewType) {
        View v =
LayoutInflater.from(context).inflate(R.layout.fac_time_table_list,parent,false);
        return new MyViewHolder(v);
    }
    @Override
    public void onBindViewHolder(@NonNull Recycle_ttl_view.MyViewHolder holder, int
position) {
        ModClass modClass = dataList.get(position);
        holder.semData.setText(modClass.getSem());
        holder.subCodeData.setText(modClass.getSubCode());
        holder.subNameData.setText(modClass.getSubName());
        holder.monData.setText(modClass.getMon());
        holder.tueData.setText(modClass.getTue());
        holder.wedData.setText(modClass.getWed());
        holder.thuData.setText(modClass.getThu());
        holder.friData.setText(modClass.getFri());
        holder.satData.setText(modClass.getSat());
    }
    @Override
    public int getItemCount() {
        return dataList.size();
    }
    public static class MyViewHolder extends RecyclerView.ViewHolder{
        TextView semData , subCodeData , subNameData , monData , tueData , wedData,
thuData, friData , satData;
        public MyViewHolder(@NonNull View itemView) {
            super(itemView);
            semData = itemView.findViewById(R.id.listSem);
            subCodeData = itemView.findViewById(R.id.listSubC);
            subNameData = itemView.findViewById(R.id.listSubN);
            monData = itemView.findViewById(R.id.listMon);
            tueData = itemView.findViewById(R.id.listTue);
            wedData = itemView.findViewById(R.id.listWed);
            thuData = itemView.findViewById(R.id.listThu);
            friData = itemView.findViewById(R.id.listFri);
            satData = itemView.findViewById(R.id.listSat);
        }
    }
}

```

## **CHAPTER 5**

### **5. TESTING**

#### **5.1 TYPES OF TESTING**

##### **5.1.1 UNIT TESTING**

In a testing level hierarchy, unit testing is the first level of testing done before integration and other remaining levels of the testing. In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. The developer writes a unit test that exposes either a software requirement or a defect. This test will fail because either the requirement isn't implemented yet, or because it intentionally exposes a defect in the existing code. Then, the developer writes the simplest code to make the test, along with other tests.

##### **5.1.2 INTEGRATION TESTING**

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units. First, determine the Integration Test Strategy that could be adopted and later prepare the test cases and test data accordingly. Study the Architecture design of the Application and identify the Critical Modules. These need to be tested on priority. Components of integration test plan includes Methods/Approaches to testing. Scopes and Out of Scopes Items of Integration Testing. Roles and Responsibilities. Pre-requisites for Integration testing. Testing environment.

## 5.2 TEST CASES AND RESULTS

Test case Id	Test case name	Input	Expected output	Actual output	Result
1	Admin Login	<a href="mailto:admin@gmail.com">admin@gmail.com</a> admin123	Login successful	Login successful	Pass
2	Admin Login	<a href="mailto:admin@gmail.com">admin@gmail.com</a> admin1	Invalid Password	Invalid Password	Pass
3	Faculty Login	FAC12 FAC12@123	Login Successful	Login Successful	Pass
4	Faculty Login	FAC12 FAC12	Invalid Password	Invalid Password	Pass
5	Student Login	4VV19IS045 4VV19IS045@123	Login Successful	Login Successful	Pass
6	Student Login	4VV19IS045 4VV19IS045@123	Invalid Password	Invalid Password	Pass
7	Faculty Registration	(FAC ID, Name, Sem, Branch)	Registration Successful	Registration Successful	Pass
8	Student Registration	(USN, Name, Sem, Branch)	Registration Successful	Registration Successful	Pass
9	Add Timetable	(Sem, Subcode, Subject name, Timings)	Timetable added successfully	Timetable added successfully	Pass
10	Update Timetable	(Sem, Exist Subcode, Subject name, Timings)	Timetable updated successfully	Timetable updated successfully	Pass

Table 5.2.1: Testcases and Results

## CHAPTER 6

### 6. SNAPSHOTS

#### Home Screen



Fig 6.0.1: This screen is common for all to login to their portal.

#### Admin Login Screen

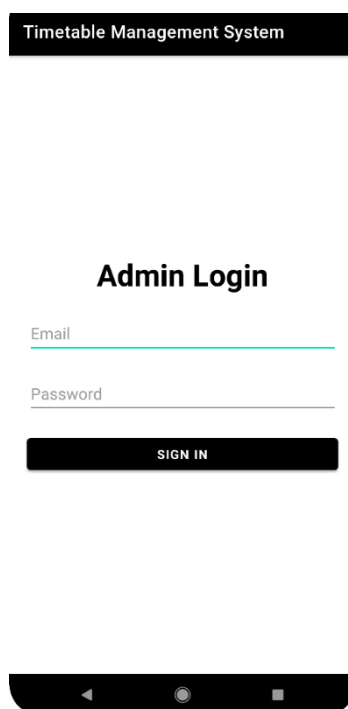


Fig 6.0.2: This screen is to login for the admin using email and password

## Student Register Screen

**Timetable Management System**

**Hii admin**

### Student Register


Name


USN

Branch

Sem

**REGISTER**

  
Student Register

  
Faculty Register


  
Profile

Fig 6.0.3: This screen is to add the student details for the database by the admin.

## Faculty Register Screen

**Timetable Management System**

**Hii admin**

### Faculty Register

Name


ID num


Semester :

☐ 1
 ☐ 2
 ☐ 3
 ☐ 4
 ☐ 5
 ☐ 6
 ☐ 7
 ☐ 8

Branch

**REGISTER**

  
Student Register

  
Faculty Register


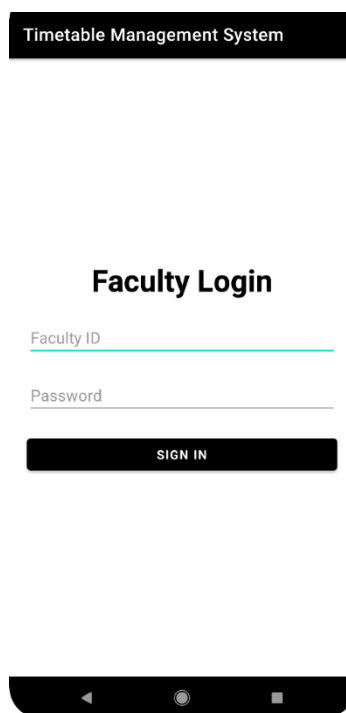
  
Profile

Fig 6.0.4: This screen is to add the faculty details for the database by the admin.

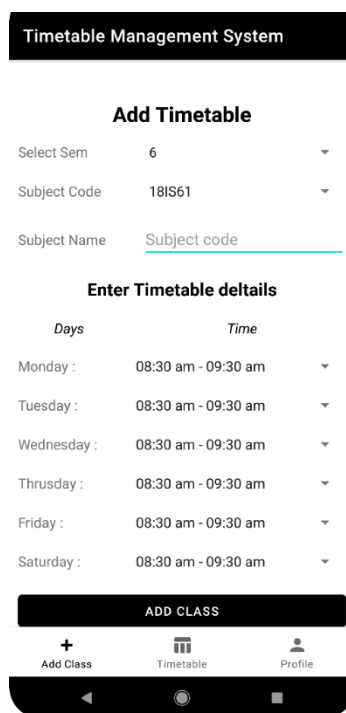
## Faculty Login Screen



The screenshot shows a mobile application interface for the Faculty Login screen. At the top, there is a black header bar with the text "Timetable Management System" in white. Below the header, the title "Faculty Login" is centered in a bold, black font. Underneath the title, there are two input fields: "Faculty ID" and "Password", each with a light blue underline. Below these fields is a black button with the text "SIGN IN" in white. At the bottom of the screen, there is a black navigation bar with three white icons: a left arrow, a circle, and a square.

Fig 6.0.5: This screen is to login for the faculty using faculty id and password

## Add Timetable Screen



The screenshot shows a mobile application interface for the Add Timetable screen. At the top, there is a black header bar with the text "Timetable Management System" in white. Below the header, the title "Add Timetable" is centered in a bold, black font. Underneath the title, there are three input fields: "Select Sem" with a dropdown arrow, "Subject Code" with a dropdown arrow, and "Subject Name" with a light blue underline. Below these fields, the title "Enter Timetable details" is centered in a bold, black font. Underneath this title, there is a table with two columns: "Days" and "Time". The table has seven rows, one for each day of the week. Each row has a dropdown arrow next to the time range. Below the table, there is a black button with the text "ADD CLASS" in white. At the bottom of the screen, there is a black navigation bar with three white icons: a plus sign, a calendar icon, and a person icon. Below the icons, there are three labels: "Add Class", "Timetable", and "Profile".

Fig 6.0.6: This screen comes when faculty login is success to add timetable.

## Faculty View Timetable Screen

**Timetable Management System**

**Timetable** REFRESH

*Subject*

Semester : 6

Sub Code : Cloud Computing

SubName : 18CS64X

*Timings*

Monday : 08:30 am - 09:30 am

Tuesday : 08:30 am - 09:30 am

Wednesday : 08:30 am - 09:30 am

Thursday : 08:30 am - 09:30 am

Friday : 08:30 am - 09:30 am

Saturday : 08:30 am - 09:30 am

+ Add Class   Timetable   Profile

Fig 6.0.7: This screen is for faculties to see timetable what they have added.

## Student Login Screen

**Timetable Management System**

**Student Login**

USN

Password

**SIGN IN**

Fig 6.0.8: This screen is to login for the student using usn and password

## Student View Timetable Screen

Timetable Management System

Timetable

Subject

Semester : 6

Sub Code : Cloud Computing

SubName : 18CS64X

Timings

Monday : 08:30 am - 09:30 am

Tuesday : 08:30 am - 09:30 am

Wednesday : 08:30 am - 09:30 am

Thursday : 08:30 am - 09:30 am

Friday : 08:30 am - 09:30 am

Saturday : 08:30 am - 09:30 am

Home

Profile

Fig 6.0.9: This screen is for students to see timetable what faculties have added.

## Student Profile Screen

Timetable Management System

Your Profile

Name : Manoj M

USN : 4VV19IS045

Branch ISE

Semester 6

Update Password

Old Password

New Password

UPDATE PASSWORD

Home

Profile

Fig 6.0.10: This screen is for students to view profile details and to update their password.



## **CHAPTER 7**

### **7. FUTURE WORK**

- Admin's can send notification to students and teachers.
- Teachers can upload syllabus copies to database and students will be able to access them.
- Teachers will be able to add notes in to the database.
- Teachers can update attendance status

## CONCLUSION

This methodology is an effective and efficient way to maintain and provide required information from the system as per need of head of institution. The proposed system reduces combust Ness of manual system with cutting edge technology that helps to visualize the complicacies of time table.

All the modules have been designed helped to retrieve data according to the given information available like Room No. semester or Time. The proposed system can be extended to make it web based.

Our project i.e., Timetable Management System is only a humble venture to satisfy the needs to manage their Timetable. Several user-friendly coding has also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the school. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progress.

## BIBLIOGRAPHY

- [https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express\\_Nodejs](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs)
- <https://www.mongodb.com/docs/drivers/node/current/quick-start/>
- <https://square.github.io/retrofit/>
- <https://www.geeksforgeeks.org/arrayadapter-in-android-with-example/>
- [https://www.researchgate.net/publication/345670589\\_Timetable\\_Management\\_System\\_for\\_Faculty\\_of\\_Computing](https://www.researchgate.net/publication/345670589_Timetable_Management_System_for_Faculty_of_Computing)
- <https://developer.android.com/reference/android/widget/ArrayAdapter>