

# DIGITAL NOTICE BOARD

## PROJECT REPORT

Submitted in partial fulfilment of the requirements for the award of the Degree of

**BACHELOR OF COMPUTER APPLICATIONS**

of the University of Calicut

Submitted By

**Sajil Mohamed.V      (NZAUBCA009)**

Guided By

**Vineesha Venugopalan**



**DEPARTMENT OF COMPUTER SCIENCE**

**NASRA COLLEGE OF ARTS AND SCIENCE**

**TIRURKAD – 679325**

**(March 2023)**

**DEPARTMENT OF COMPUTER SCIENCE**  
**NASRA COLLEGE OF ARTS & SCIENCE**

(Affiliated to University of Calicut)

**TIRURKAD - 679325**



**PROJECT WORK**  
**CERTIFICATE**

Certified that this is a bonafied record of the project work

**DIGITAL NOTICE BOARD**

Done by

**Sajil Mohamed. V**

**(NZAUBCA009)**

Submitted in partial fulfilment of the requirements for the  
award of the Degree Bachelor of Computer Application  
of the University of Calicut

**Faculty Guide**

**Head Of the Department**

**External Examiner**

DEPARTMENT OF COMPUTER SCIENCE

NASRA COLLEGE OF ARTS AND SCIENCE

TIRURKAD, MALAPPURAM – 679325



## CERTIFICATE

This is to certify that this Project report entitled “ Digital Notice Board ” is a bonafied record of the project done by ,Sajil Mohamed .V, of sixth semester BCA, Nasra College Of Arts And Science, Tirurkad, towards partial fulfilment of the requirement for the award of degree of Bachelor of Computer Applications University of Calicut.

### Internal Guide

**Mrs.Vineesha Venugopalan** (Asst.Professor)

Department of Computer Science

Nasra College of Arts & Science, Tirurkad

### Head Of Department

**Azhar Mohammed Basheer** (HOD)

Department of Computer Science

Nasra College of Arts & Science, Tirurkad

Submitted for Viva – Voce Examination held on .....

### INTERNAL EXAMINER

**Name :**

**Signature :**

**Date :**

### EXTERNAL EXAMINER

**Name :**

**Signature :**

**Date :**

## **DECLARATION**

I hereby that this project work entitled ‘ **Digital Notice Board** ’ submitted at Nasra College of Arts & Science (Affiliated to University of Calicut) is a record of original work done by me under the supervision and guidance of **Mrs. Vineesha Venugopalan** Department of Computer Science.

**Name of student**

**Register number**

**Sajil Mohamed.V**

**NZAUBCA009**

**Signature of Candidate**

**Sajil Mohamed V :**

## ACKNOWLEDGEMENT

First of all ,I would like to express our sincere gratitude to God Almighty for the constant love and grace he has showered upon me. I express my gratitude to the respected principal **Dr. P. Zubair** , Nasra College of Arts & Science Tirurkad, for giving me this golden opportunity to undertake my course in this college.

I also express my great deal of gratitude to the Head Of Department and our internal guide, **Mrs. Vineesha Venugopalan** for her effective guidance, timely suggestions, and encouragement. I also express my gratitude to all teachers in Department of Computer Science.

## SYNOPSIS

The purpose of this document is to present a detailed description of the project **“Digital Notice Board”**. A digital notice board is an electronic display board that is used to convey information to a targeted audience in a digital format. It is designed to replace traditional paper-based notice boards in public places such as schools, universities, hospitals, offices, and other places where information needs to be communicated to a large group of people.

The project Digital Notice Board is a simple android PyCharm application project that can display a variety of content, including text, images, videos, and even live feeds from social media platforms. The content can be updated remotely using a computer, smartphone or tablet, which makes it easy for organizations to keep their audience informed about the latest news, events, and announcements.

The Digital Notice Board Android Application is an innovative project that aims to improve communication and information dissemination within educational institutions. Traditional notice boards are often unreliable and inefficient, as they require manual updates and are limited in their reach. The goal of our application is to provide a more efficient and reliable solution for maintaining and controlling notices all over an organization.

# INDEX

<b>CHAPTER 1: INTRODUCTION</b>	<b>08</b>
1.1 Introduction	09
1.2 Objectives	10
1.3 Identification of the Problem	11
1.4 Aim	11
1.5 Existing System	12
1.6 Proposed System	12
1.7 Feasibility Study	13
<b>CHAPTER 2: SOFTWARE REQUIREMENT SPECIFICATION</b>	<b>15</b>
2.1 Software Requirements	16
2.1.1 For Admin	16
2.1.2 For Users	16
2.2 Hardware Requirements	16
<b>CHAPTER 3: SOFTWARE IMPLEMENTATION</b>	<b>17</b>
3.1 Front-End	18
3.2 Back-End	18
3.3 Database	20
3.4 IDE	21
<b>CHAPTER 4: DESIGN AND PLANNING</b>	<b>24</b>
4.1 System Analysis	25
4.2 Input Design	26
4.3 Output Design	26
4.4 General Overview	27

4.5 Flowchart	28
4.6 Class Diagram	29
4.7 Table Structure	30
4.8 Data Flow Diagram	31
4.8.1 Level: 0	32
4.8.2 Level: 1.1	33
4.8.3 Level: 1.2	34
<b>CHAPTER 5: PROGRAM CODE AND RESULTS</b>	<b>35</b>
5.1 Back-End	36
5.2 Front-End	48
5.3 Results	54
<b>CHAPTER 6: TESTING</b>	<b>59</b>
6.1 System Testing	60
6.1.1 Test Cases	60
6.2 Testing Strategy	61
6.2.1 Unit Testing	62
6.2.2 Integration Testing	62
6.2.3 System Testing	62
6.2.4 Acceptance Testing	62
6.3 Testing Environment	62
<b>CHAPTER 7: SYSTEM IMPLEMENTATION</b>	<b>63</b>
<b>CHAPTER 8: LIMITATIONS</b>	<b>65</b>
<b>CHAPTER 9: FUTURE ENHANCEMENT</b>	<b>67</b>
<b>CHAPTER 10: CONCLUSION</b>	<b>69</b>
<b>CHAPTER 11: BIBLIOGRAPHY</b>	<b>71</b>



# **CHAPTER 1**

## **INTRODUCTION**

## **1.1 INTRODUCTION**

Notice board is an essential information gathering system in our life. In our day-to-day life, we can see notice boards in various places like educational institutions, railway stations, shopping malls, Bus stations, offices etc. So we can say that Notice boards are the places to leave public information such as advertise events, announce events or provide attention to the public, etc. Nowadays a Separate person is needed to stick that information on the notice board. It will lead to loss of time as well as usage of manpower. In conventional analogue type notice boards paper is the main medium for information exchange. We know that information counts are endless. So there is a usage of a huge amount of paper for displaying those endless counts of the information.

The problems faced by the wooden or conventional type notice boards are resolved by the implementation of our digital notice board. It will bring an advanced means of passing notices around in the world in a much easier and efficient way. Due to the popularity of the internet, we choose the internet as a medium for transferring information. The Internet of things (IoT) is the network of physical devices, vehicles, home appliances and other items embedded with electronics. Software, which enables these objects to connect and exchange data. Each device is uniquely identifiable through its Embedded computing system but is able to interoperate within the existing Internet infrastructure For providing security, we add username and password type authentication system. So only respective authority can send information. Raspberry Pi which is the Heart of our system. A monitor is interfaced with Raspberry Pi. So the information in the form of text, image and pdf can display on the large screens. Our primary aim is to get more people's attention on the display. By the usage of high definition display devices, people can get more attention on the notice board rather than conventional notice boards. In conventional wireless notice board can display only texted messages. But in our newly implemented system can display images and pdf documents in addition to text messages. Because in Educational institutions the majority of information given from the higher authorities in the form of images or pdf format. So displaying these types of information make our system more user-friendly. Due to the utilization of the internet, the sender can send a message anywhere in the world. There is no range limitation for the successful exchange of information.

## 1.2 Objectives

The Digital Notice Board Android Application is an innovative project that aims to improve communication and information dissemination within educational institutions. Traditional notice boards are often unreliable and inefficient, as they require manual updates and are limited in their reach. The goal of our application is to provide a more efficient and reliable solution for maintaining and controlling notices all over an organization. It is developed using the softwares such as Android Studio, PyCharm and SQLyog.

A digital notice board is an electronic display board that is used to display information, announcements, news, updates, and other relevant messages in a dynamic and interactive way. It is a modern alternative to traditional notice boards that use paper-based notices.

Digital notice boards come in different types and sizes, with LED, LCD, or plasma screens being the most common. They are connected to a computer or network of computers, which control the content displayed on the board. This enables easy updates and changes to the displayed content.

The “**Digital notice board**” has two modules. The first module is admin which can manage all the data belonging to the system and use all system functions or features. The second module is teachers which can add important notices in the form of documents, audio, video, images etc.

In this project, we will explore the development and implementation of the Digital Notice Board Android Application. We will start with an analysis of the requirements, followed by the design and development of the application. We will also test the application to ensure its functionality, reliability, and security. Finally, we will assess the effectiveness of the application in improving communication and information dissemination within educational institutions

### **1.3 Identification of the problem**

The education system has evolved significantly in recent years, with technological advancements transforming the way teaching and learning is done. However, despite the advances in technology, many educational institutions still rely on traditional notice boards for communication with students and faculty members. This creates a number of problems, including the inefficiency of manually updating and maintaining notice boards, the limitations in terms of the number of notices that can be displayed at a time, and the lack of accessibility to notices for students who are not physically present on campus.

Moreover, traditional notice boards also limit the type of information that can be shared, as they are typically limited to paper notices and flyers. This means that important updates, such as changes in the academic calendar, exam schedules, and assignment deadlines, may not reach all students on time, leading to confusion and missed deadlines.

### **1.4 Aim**

The aim of this project is to develop a digital notice board android application that will assist teachers in an educational environment to communicate important academic information to students and faculty members in a more efficient and accessible manner. The application will provide a centralized platform for uploading and displaying notices, as well as sending notifications to users, ensuring that everyone stays informed and up-to-date with the latest academic affairs. The goal is to replace traditional notice boards with a modern and user-friendly solution that can handle a wide range of information formats, including videos, images, and documents, and provide easy-to-use features for managing notices. Ultimately, the aim of the project is to enhance communication and improve the overall educational experience for students and faculty members.

## **1.5 Existing system**

The existing system in many educational institutions relies on traditional notice boards for communication with students and faculty members. These notice boards are physical displays typically made of cork, wood, or metal, where paper notices and flyers are posted manually. However, this system has several limitations that can be addressed by a digital notice board application.

Firstly, traditional notice boards are inefficient in terms of updating and maintenance. Notices have to be manually posted and removed, which can be time-consuming and prone to errors. Additionally, only a limited number of notices can be displayed at a time, which can lead to important information being missed by students and faculty members.

Secondly, traditional notice boards are limited in the type of information that can be shared. They are typically restricted to paper notices and flyers, which can be difficult to read and easily misplaced. This means that important updates, such as changes in academic schedules or exam dates, may not reach all students in a timely manner.

Lastly, traditional notice boards also require physical presence on campus to access the information posted on them. This can be a limitation for students who are not able to be on campus all the time, or for faculty members who are working remotely.

Overall, the existing system of traditional notice boards has several limitations that can be addressed by a digital notice board android application.

## **1.6 Proposed system**

The proposed system is a digital notice board android application that will provide a centralized platform for communication between educational institutions, teachers, and students. The application will allow for easy uploading and management of notices, as well as sending notifications to users, ensuring that everyone stays informed and up-to-date with the latest academic affairs. The proposed system will address the limitations of the existing system, such as inefficiency in updating and maintenance, limited information sharing, and lack of accessibility.

The digital notice board android application will have several features, including:

- **Notice management:** The application will allow for easy uploading and management of notices, including the ability to create, edit, and delete notices. Users will be able to add a variety of information formats, including text, images, videos, and documents.

- **Notification system:** The application will have a notification system that will send alerts to users when new notices are uploaded or when there are updates to existing notices.
- **User authentication and access control:** The application will have a user authentication system to ensure that only authorized users can access and modify notices. Additionally, the access control system will allow different users to have different levels of access and control over notices.
- **Search and filtering:** The application will have a search and filtering system that will allow users to easily find the notices they are interested in by filtering notices based on keywords, tags, or categories.
- **Accessibility:** The application will provide easy access to notices for all users, regardless of their location or physical presence on campus. Users will be able to access notices from any device with an internet connection, such as a mobile phone, tablet, or computer.

Overall, the proposed system of a digital notice board android application will provide an efficient, user-friendly, and accessible platform for communication between educational institutions, teachers, and students.

## **1.7 Feasibility study**

### **Feasibility Study for "Digital Notice Board" Android Application**

#### **Introduction:**

This feasibility study aims to assess the viability of developing a "Digital Notice Board" Android application for educational institutions. The study will analyze various aspects of the proposed system, including technical, operational, economic, and scheduling factors.

#### **I. Technical Feasibility:**

The development of the proposed system requires a skilled team of developers with knowledge of Android app development and back-end systems. The team must also be proficient in integrating various features such as notice management, notification system, user authentication, search and filtering, and accessibility. The required hardware and

software for the project are readily available and can be acquired with ease.

**II. Operational Feasibility:**

The proposed system will require an operational infrastructure that includes servers, hosting, and maintenance facilities. Educational institutions will have to provide IT support to maintain the application and ensure the security of the system.

**III. Economic Feasibility:**

The project is economically feasible because it has a wide scope for revenue generation. Educational institutions will be charged for the use of the application, which can be a significant source of income. The development and operational costs of the application will be offset by the revenue generated by subscription fees.

**IV. Scheduling Feasibility:**

The proposed project can be completed within a reasonable timeframe with proper scheduling and management. The development team can deliver the project within the specified time by following the project plan and delivering the project in phases.

**Conclusion:**

The feasibility study concludes that the development of a "Digital Notice Board" Android application is technically feasible, operationally feasible, economically feasible, and schedulable. The development of the proposed system will provide a modern and efficient platform for communication in educational institutions.

# **CHAPTER 2**

## **SOFTWARE REQUIREMENT SPECIFICATION**



## **2.1 SOFTWARE REQUIREMENTS (Minimum)**

### **2.1.1 For Admin**

Operating System : Android OS Version 5.0 and above

Front end : Android Programming

Back end : Java, Python, MySQL

IDE : Android Studio, PyCharm, SQLyog

### **2.1.2 For Users**

Operating System : Android OS Version 5.0 and above

## **2.2 HARDWARE REQUIREMENTS (Minimum)**

- Hardware requirements of the admin
  - Processor : Exynos 990 or above
  - Memory : 1GB RAM with 4GB disk space or above
  - Misc : Internet access
  
- Hardware requirements for the user
  - Processor : Exynos 990 or above/Qualcomm snapdragon
  - Memory : 1GB RAM with 10 MB free ROM
  - Misc : Internet access

# **CHAPTER 3**

## **SOFTWARE IMPLEMENTATION**

## **3.1 Front End**

### **Android programming**

Android programming is a front-end development technology used to create user interfaces for Android applications. In the context of our project, the Digital Notice Board Android application, Android programming will be used to create the user interface for the application.

Android programming involves using XML to define the layout of user interface components, such as buttons, text fields, images, and other graphical elements. Android programming also includes using Java or Kotlin to add functionality to the user interface components, such as handling user input, displaying data, and connecting to back-end systems.

To create the front-end of the Digital Notice Board Android application, the developer will use Android Studio, which is an integrated development environment (IDE) specifically designed for Android development. Android Studio includes tools for designing user interfaces, writing code, and debugging applications.

The developer will use Android programming to create the different screens of the application, such as the login screen, the notice board screen, and the notice details screen. The developer will also use Android programming to add functionality to these screens, such as allowing users to create, edit, and delete notices, and displaying notifications to users.

## **3.2 Back end**

### **Java**

Java is a widely used programming language that can be used as the back-end technology for the Digital Notice Board Android application. In the context of our project, Java can be used to implement the server-side logic, database management, and communication between the front-end and back-end of the application.

The back-end of the Digital Notice Board Android application will be responsible for managing the storage and retrieval of notices, user authentication and authorization, and sending notifications to users. Java can be used to implement these functions by building a server-side application that runs on a web server.

Java provides a variety of frameworks and libraries that can be used to build robust and scalable back-end systems. For example, Spring framework provides a wide range of features for building web applications, including dependency injection, security, and database access. Hibernate is another Java-based library that can be used to manage database access and simplify the process of working with databases.

The back-end of the Digital Notice Board Android application will communicate with the front-end using APIs (Application Programming Interfaces). Java can be used to create these APIs by building RESTful services, which allow data to be transferred between the front-end and back-end in a standardized format.

## **Python**

Python is a popular programming language that can also be used as the back-end technology for the Digital Notice Board Android application. In the context of our project, Python can be used to implement the server-side logic, database management, and communication between the front-end and back-end of the application.

Python offers a number of benefits as a back-end language, including its ease of use, readability, and versatility. Python provides a variety of frameworks and libraries that can be used to build robust and scalable back-end systems. For example, Django is a popular Python-based web framework that provides a wide range of features for building web applications, including database access, security, and user authentication.

The back-end of the Digital Notice Board Android application will be responsible for managing the storage and retrieval of notices, user authentication and authorization, and sending notifications to users. Python can be used to implement these functions by building a server-side application that runs on a web server.

In addition, Python is well-suited to implementing machine learning algorithms and data analysis, which may be useful for your project. For example, you could use machine learning algorithms to automatically categorize notices or analyze user engagement with the app.

The back-end of the Digital Notice Board Android application will communicate with the front-end using APIs (Application Programming Interfaces). Python can be used to create these APIs by building RESTful services, which allow data to be transferred between the front-end and back-end in a standardized format.

## **MySQL**

MySQL is a popular relational database management system that can be used as the back-end technology for the Digital Notice Board Android application. In the context of your project, MySQL can be used to store and manage data related to notices, users, and other application-related information.

MySQL provides a range of features that make it a good choice for the back-end of web and mobile applications. These features include support for complex queries, scalability, security, and data integrity. MySQL can also be easily integrated with other programming languages such as Java, Python, and PHP.

The back-end of the Digital Notice Board Android application will use MySQL to manage the storage and retrieval of notices, user authentication and authorization, and other application-related data. For example, MySQL can be used to store information such as notice titles, descriptions, upload dates, and author information.

In addition, MySQL can be used to implement advanced features such as user authentication and authorization, which are essential for securing the application and ensuring that only authorized users can access sensitive information. MySQL can also be used to manage user preferences and settings, such as notification preferences and language preferences.

MySQL can be accessed and manipulated using SQL (Structured Query Language), which is a standard language for managing relational databases. The back-end of the Digital Notice Board Android application will use SQL to create, read, update, and delete data in the MySQL database.

Overall, MySQL is a powerful and scalable database management system that can be used as the back-end technology for the Digital Notice Board Android application. It provides a range of features and capabilities that are essential for managing data and ensuring the security and integrity of the application.

### **3.3 Database**

#### **MySQL**

The database of the Digital Notice Board Android application will use MySQL to manage the storage and retrieval of notices, user authentication and authorization, and other application-related data. For example, MySQL can be used to store information such as notice titles, descriptions, upload dates, and author information.

MySQL also provides features that can be used to optimize database performance, such as indexing and caching. These features can help to speed up queries and reduce the time it takes for the application to retrieve data from the database.

### **3.4 IDE**

#### **Android Studio**

Android Studio is an Integrated Development Environment (IDE) that can be used to develop the Digital Notice Board Android application. Android Studio is designed specifically for developing Android applications and provides a range of features that make the development process faster and more efficient.

Android Studio provides a user-friendly interface for developing Android applications, with a range of tools and features that make it easy to design, build, and test applications. The IDE includes a visual layout editor, code editor, debugger, emulator, and a range of other tools that make it easier to develop, test, and debug applications.

One of the key advantages of using Android Studio for our project is that it provides a range of features and tools that are specifically designed for developing Android applications. For example, Android Studio includes templates for creating different types of Android applications, such as blank activities, login screens, and navigation drawers. These templates can be customized to suit the specific requirements of our project, which can save time and effort during the development process.

In addition, Android Studio provides tools for debugging and testing applications, including the Android Emulator, which allows you to test our application on a virtual Android device. Android Studio also provides support for version control systems such as Git, which can be used to manage and track changes to your project code.

Overall, Android Studio is a powerful and efficient IDE for developing Android applications, and it provides a range of features and tools that can help you to develop your Digital Notice Board Android application more efficiently and effectively.

## **PyCharm**

PyCharm is an Integrated Development Environment (IDE) that can be used to develop the backend of the Digital Notice Board application using Python. PyCharm is specifically designed for Python development and provides a range of features that make the development process faster and more efficient.

PyCharm provides a user-friendly interface for developing Python applications, with a range of tools and features that make it easy to write, test, and debug code. The IDE includes a code editor, debugger, testing framework, and a range of other tools that make it easier to develop, test, and debug applications.

One of the key advantages of using PyCharm for our project is that it provides a range of features and tools that are specifically designed for Python development. For example, PyCharm includes intelligent code completion, code highlighting, and code analysis tools that can help you to write better quality code more quickly. PyCharm also provides support for virtual environments, which can help you to manage dependencies and keep your development environment clean and organized.

In addition, PyCharm provides integration with version control systems such as Git, which can be used to manage and track changes to our project code. PyCharm also provides support for popular Python web frameworks such as Flask and Django, which can be used to develop web applications.

Overall, PyCharm is a powerful and efficient IDE for developing Python applications, and it provides a range of features and tools that can help you to develop the backend of our Digital Notice Board application more efficiently and effectively.

## **SQLyog**

SQLyog is an Integrated Development Environment (IDE) that can be used as a graphical user interface (GUI) for managing and interacting with the MySQL database in your Digital Notice Board project. SQLyog is specifically designed for MySQL database management and provides a range of features that make it easy to create, modify, and manage MySQL databases.

SQLyog provides a user-friendly interface for managing MySQL databases, with a range of tools and features that make it easy to create tables, insert data, run queries, and manage the database structure. The IDE includes a query editor, schema designer, backup and restore tools, and a range of other tools that make it easier to manage and interact with MySQL databases.

One of the key advantages of using SQLyog for our project is that it provides a range of features and tools that are specifically designed for MySQL database management. For example, SQLyog includes a schema designer tool that allows you to visually create and modify the structure of our database, which can save time and effort compared to creating tables and columns manually.

In addition, SQLyog provides support for importing and exporting data in a variety of formats, making it easy to migrate data between databases or share data with other applications. SQLyog also provides support for managing user privileges and security, which can help you to keep our database secure and protected.

Overall, SQLyog is a powerful and efficient IDE for managing MySQL databases, and it provides a range of features and tools that can help you to manage and interact with the MySQL database in our Digital Notice Board project more efficiently and effectively.



# **CHAPTER 4**

## **DESIGNING PLANNING**

## **4.1 System Analysis**

System analysis is a crucial part of the software development life cycle that helps to identify, define, and analyse the requirements and objectives of a software project. In the case of the Digital Notice Board project, system analysis would involve identifying and analysing the various requirements of the system, including the functional requirements and non-functional requirements.

Functional requirements of the system would include the specific features and functionalities that the Digital Notice Board application should provide, such as the ability to upload and display notices, news, and updates related to academic affairs. These functional requirements would also include features such as user authentication and authorization, data storage and retrieval, and notification and alert mechanisms.

Non-functional requirements of the system would include aspects such as performance, security, and usability. For example, the system should be able to handle a large number of users and notices without compromising on performance. It should also provide appropriate security measures such as encryption and access control to ensure the confidentiality and integrity of the data. Usability requirements would involve ensuring that the system is easy to use and navigate, with clear and concise instructions and user-friendly interfaces.

The system analysis would also involve identifying the various stakeholders of the system, including the end-users, administrators, and developers. The analysis would involve identifying their specific needs and requirements, and ensuring that the system is designed and developed to meet these needs.

In summary, system analysis is a critical step in the software development life cycle that involves identifying and analysing the various requirements and objectives of the system. In the case of the Digital Notice Board project, the system analysis would involve identifying the functional and non-functional requirements of the system, identifying the stakeholders, and ensuring that the system is designed and developed to meet their needs and requirements.

## **4.2 Input Design**

The input design for the Digital Notice Board application would include various forms and input fields for users to enter and upload the notices and updates related to academic affairs. These input fields would include the title of the notice, the date of the notice, the author of the notice, and the content of the notice.

In addition, the input design would also include fields for users to enter their login credentials, such as their username and password, in order to access the system and perform various actions such as uploading and deleting notices.

To ensure that the input design is user-friendly and easy to use, it should be designed with a clear and concise layout, with appropriate labels and instructions for each input field. The input design should also include appropriate validation checks to ensure that users enter valid and correct data into the system, such as date formats or character limits.

Finally, the input design should be tested thoroughly to ensure that it meets the requirements and needs of the end-users and that it is free from any errors or issues that could affect the functionality or usability of the system.

In summary, input design is a critical component of any software development project, and in the case of the Digital Notice Board project, it involves defining the various input fields and data elements that users would need to enter into the system. The input design should be user-friendly, easy to use, and tested thoroughly to ensure that it meets the needs and requirements of the end-users.

## **4.3 Output design**

The output design for the Digital Notice Board application would include various screens and interfaces that users would interact with to view and access the notices. These screens would display the notices in a clear and concise manner, with appropriate headings and descriptions. The output design would also include various sorting and filtering options to allow users to search and view notices based on specific criteria such as date, author, or category.

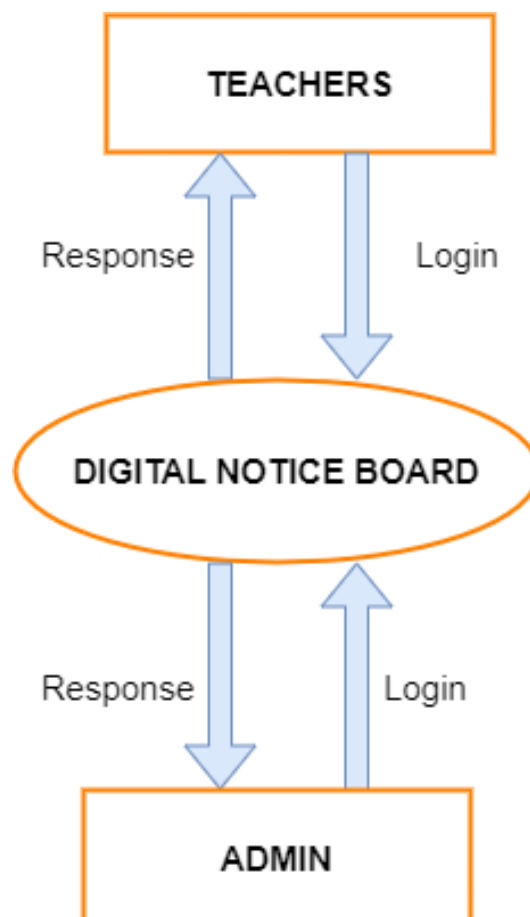
In addition, the output design would also include various notification and alert mechanisms, such as email or push notifications, to inform users of new notices or updates that have been posted on the notice board.

To ensure that the output design is user-friendly and easy to use, it should be designed with a clear and consistent layout, with appropriate colors and font sizes. The output design should also be responsive and adaptable to different screen sizes and resolutions, to ensure that it is accessible to all users.

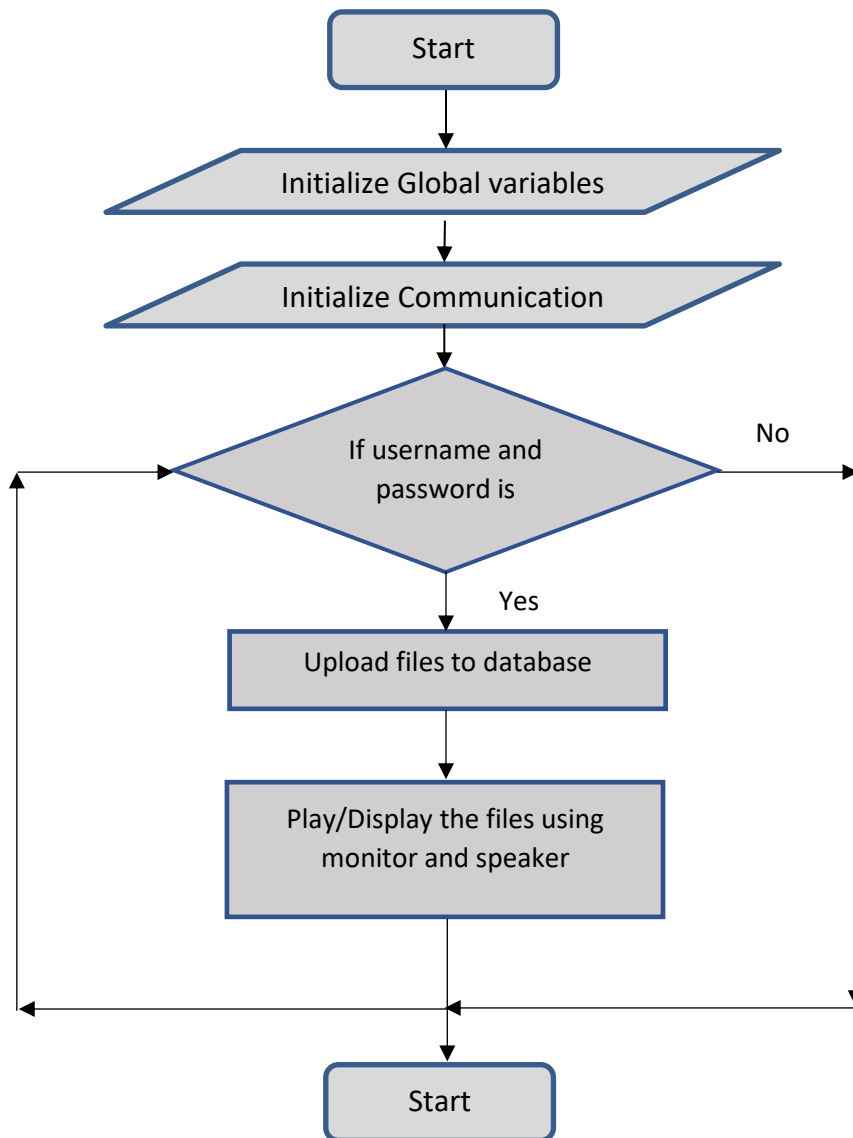
Finally, the output design should be tested thoroughly to ensure that it meets the needs and requirements of the end-users, and that it is free from any errors or issues that could affect the functionality or usability of the system.

In summary, output design is a critical aspect of software development that determines how the system presents information and data to the end-users. In the case of the Digital Notice Board project, the output design would involve defining the various ways in which the system would display the notices and updates related to academic affairs to the end-users, and ensuring that the output design is user-friendly, responsive, and tested thoroughly to meet the needs and requirements of the end-users.

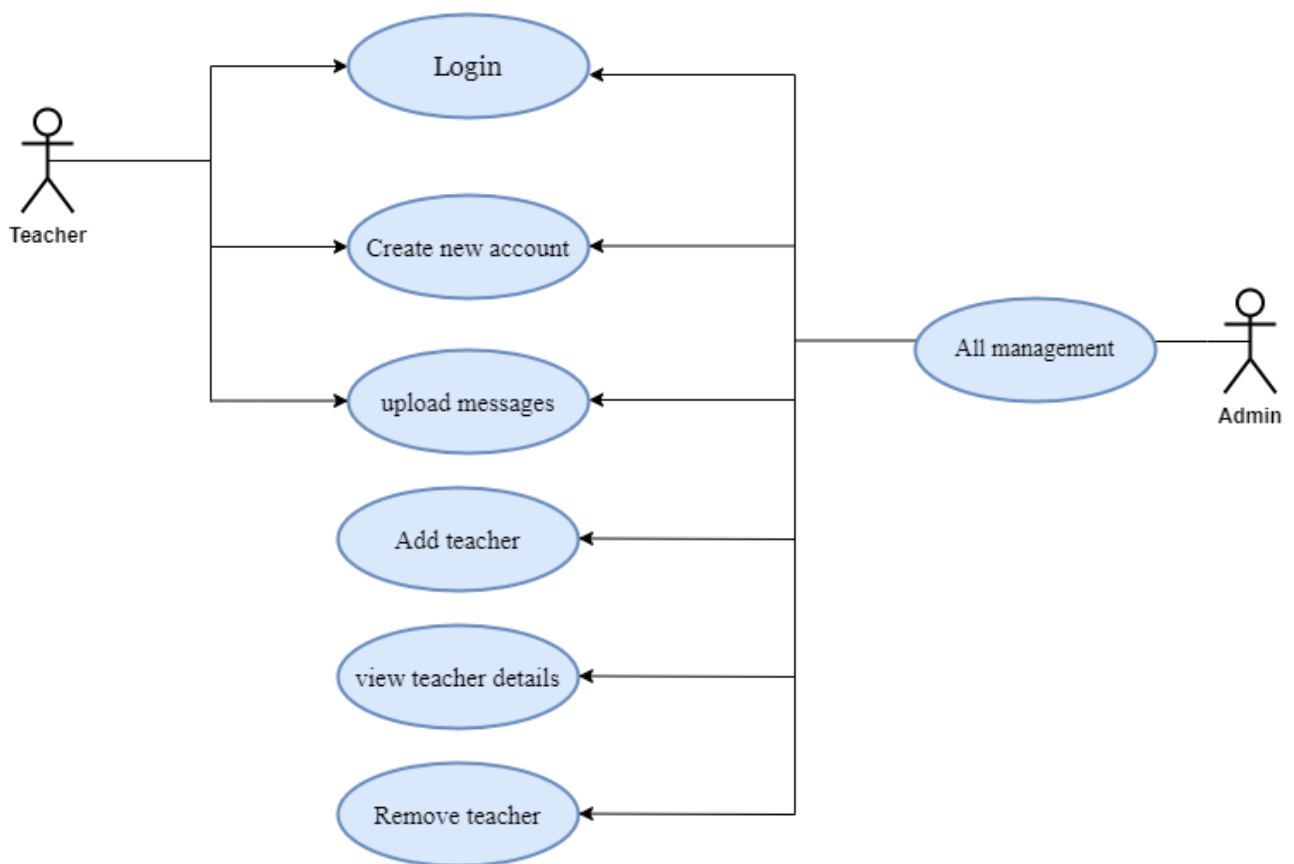
#### **4.4 General overview**



## 4.5 Flowchart



## 4.6 Class Diagram



## 4.7 Table Structure

Table No 1: Admin

NAME	DATA TYPE	CONSTRAINTS
Username	Varchar(30)	NULL
Password	Varchar(30)	NULL
User Type	Varchar(30)	NULL

Table No 2: Teacher

NAME	DATA TYPE	CONSTRAINTS
Username	Varchar(100)	NULL
Gender	Varchar(70)	NULL
Email	Varchar(50)	NULL
Phone	Varchar(50)	NULL
Department	Varchar(60)	NULL
Password	Varchar(60)	NULL

## 4.8 Data Flow Diagram

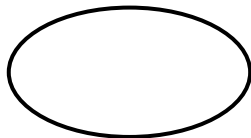
Data Flow Diagram are used to define the flow of system and its resources such as information. DFD's are a way of expressing system requirements in a graphical manner. DFD represents one of the most ingenious tools used for structured analysis. A DFD is also known as a bubble chart. A data flow diagram (DFD) is a visual representation of the flow of data through a system or process. It shows how data moves from one process to another, where it is stored, and how it is transformed along the way. DFDs are commonly used in software engineering and business analysis to model the flow of information within an organization, and to help identify potential inefficiencies or areas for improvement. A DFD consists of a series of symbols and connectors, which represent the various components of the system, and the arrows, which represent the flow of data between them. By creating a DFD, stakeholders can gain a better understanding of the system's architecture, identify potential issues or bottlenecks, and develop strategies for optimization and improvement. In the normal condition, logical DFD can be completed using only four symbols.



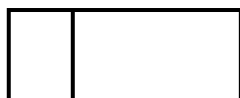
Represents source or destination data



Represents data flow



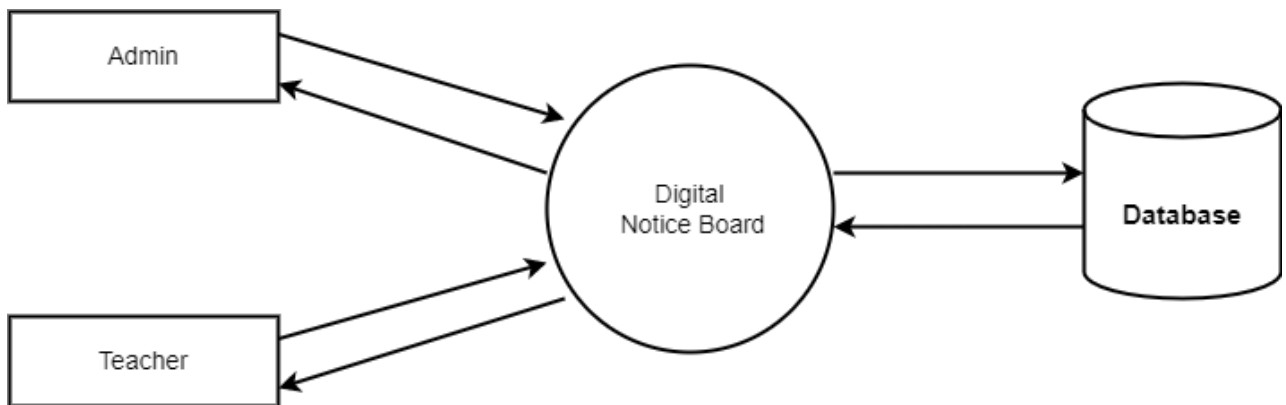
Represent a process that transforms incoming data



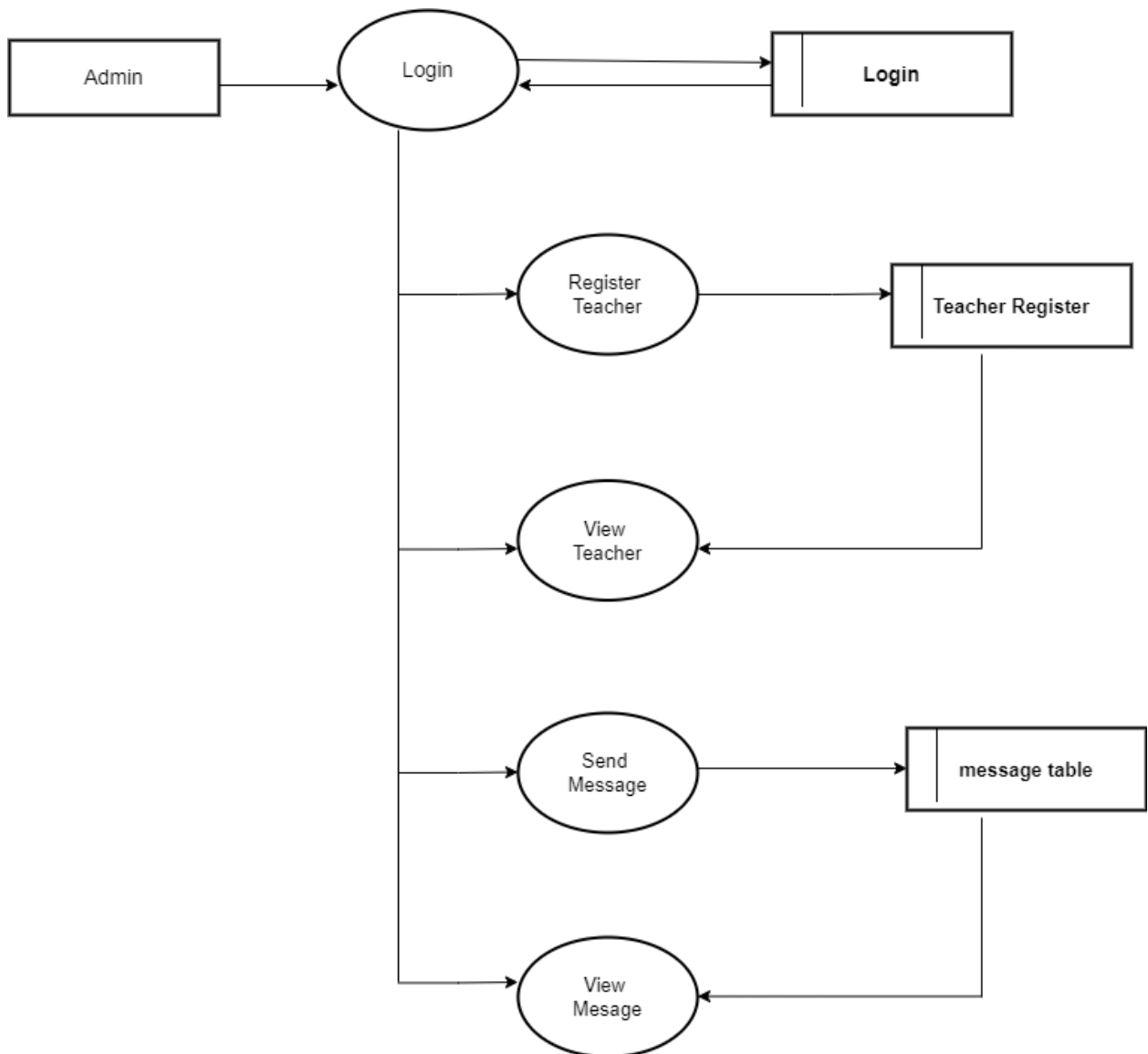
Represents data store



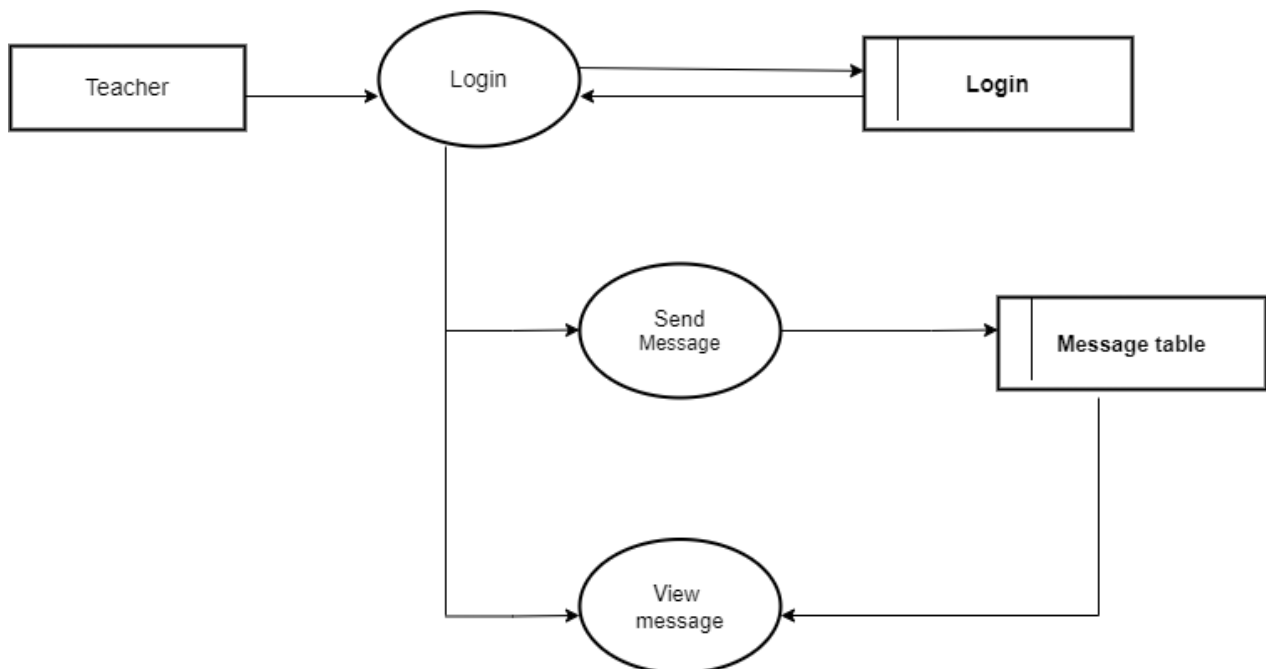
### 4.8.1 Level: 0



### 4.8.2 Level: 1.1



### 4.8.3 Level: 1.2



Overall, data flow diagrams (DFD) are a valuable tool in “Digital Notice Board” project, helping to design, develop, test, and communicate the digital notice board application effectively.

# **CHAPTER 5**

## **PROGRAM CODE AND RESULT**

## 5.1 Back End

### 5.1.1 app.py

```
from tkinter import *
from flask import *
from gtts import gTTS
from playsound import playsound
import datetime
import threading
from PIL import ImageTk, Image
from pdf2image import convert_from_path
import pymysql
from tkvideo import tkvideo
from tkVideoPlayer import TkinterVideo
import moviepy.editor
import json
import os
import requests
from flask import *
import time
import random
import subprocess
import base64

app = Flask(__name__)
root = Tk()
root.geometry('600x600+400+40')

con = pymysql.connect(host='localhost', port=3306, user='root', passwd='root', db='demo')
cmd = con.cursor()

@app.route('/addVideo', methods=['get', 'post'])
def addVideo():
    r = {}
```

```
    r = {}
    v = request.form['video']
    filename = "C:\\Users\\sajil\\PycharmProjects\\noticeboard\\src\\static\\video\\sample.mp4"
    a = base64.b64decode(v)
    fh = open(filename, "wb")
    fh.write(a)
    fh.close()
    videoload()
    r['status'] = "success"
    return jsonify(r)

@app.route('/addImage', methods=['get', 'post'])
def addImage():
    r = {}
    v = request.form['photo']
    filename = "C:\\Users\\sajil\\PycharmProjects\\noticeboard\\src\\static\\image\\702.jpg"
    a = base64.b64decode(v)
    fh = open(filename, "wb")
    fh.write(a)
    fh.close()
    imageload()
    r['status'] = "success"
    return jsonify(r)

@app.route('/addAudio', methods=['get', 'post'])
def addAudio():
```

```

r = {}
v = request.form['audio']
filename = "C:\\Users\\sajil\\PycharmProjects\\noticeboard\\src\\static\\soundfiles\\nasra.mp3"
a = base64.b64decode(v)
fh = open(filename, "wb")
fh.write(a)
fh.close()
p()

r['status'] = "success"
return jsonify(r)

@app.route('/addPdf', methods=['get', 'post'])
def addPdf():
    r = {}
    v = request.form['video']
    filename = "C:\\Users\\sajil\\PycharmProjects\\noticeboard\\src\\static\\pdf\\sample.pdf"
    a = base64.b64decode(v)
    fh = open(filename, "wb")
    fh.write(a)
    fh.close()
    pdfload()
    r['status'] = "success"
    return jsonify(r)

@app.route('/login', methods=['get', 'post'])
def login():
    uname = request.form["username"]
    pswd = request.form["password"]

```

```

cmd.execute("select * from login where username='" + uname + "' and password='" + pswd + "'")
account = cmd.fetchone()
if account is not None:
    global lid
    lid = account[0]
    return jsonify({'status': 'ok', 'lid': account[0], 'type': account[3]})
else:
    return jsonify({'status': 'error'})

@app.route('/action_register', methods=['get', 'post'])
def action_register():
    name = request.form['name']
    gender = request.form['gender']
    email = request.form['email']
    phone = request.form['phone']
    spinner = request.form['spinner']
    password = request.form['password']

    cmd.execute("INSERT INTO login VALUES(NULL, '" + email + "', '" + password + "', 'user')")
    login_id = con.insert_id()
    cmd.execute(
        "INSERT INTO students VALUES(NULL, '" + name + "', '" + gender + "', '" + email + "', '" + phone + "', '" + spinner + "', \
'" + password + "', '" + str(login_id) + "')"
    con.commit()
    return jsonify({'status': "success"})

@app.route('/view', methods=['GET', 'POST'])
def view_votes():

```

## 5.1.2 IP address.java

```
package com.example.digitalnoticeboard;

import ...

public class ip_address extends AppCompatActivity {

    EditText e1;
    Button b1;
    SharedPreferences sharedPreferences;

    //to grand permission of app
    public static boolean hasPermissions(Context context,String... permissions)
    {
        if (context!=null & permissions!=null)
        {
            for (String permission : permissions)
            {
                if(ActivityCompat.checkSelfPermission(context,permission)!= PackageManager.PERMISSION_GRANTED)
                {
                    return false;
                }
            }
        }
        return true;
    }

    @SuppressWarnings("MissingInflatedId")

    //connection
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_ip_address);

            e1=(EditText) findViewById(R.id.editTextTextPersonName);
            b1=(Button) findViewById(R.id.button6);
            sharedPreferences=getSharedPreferences( name: "",Context.MODE_PRIVATE);

            //GRANTING THE PERMISSION TO READ AND WRITE
            int PERMISSION_ALL=1;
            String[] PERMISSIONS={Manifest.permission.READ_EXTERNAL_STORAGE,Manifest.permission.WRITE_EXTERNAL_STORAGE};
            if(!hasPermissions( context: this,PERMISSIONS))
            {
                ActivityCompat.requestPermissions( activity: this,PERMISSIONS,PERMISSION_ALL);
            }

            //automatically stores previous ip address
            e1.setText(sharedPreferences.getString( s: "ip", s1: ""));
            b1.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    //matching the ip address
                    String ip=e1.getText().toString();
                    if(!Patterns.IP_ADDRESS.matcher(ip).matches())
                    {
                        e1.setError("enter valid ipaddress");
                    }
                    else
                    {
                        SharedPreferences.Editor ed=sharedPreferences.edit();
                        ed.putString( s: "ip",ip);
                    }
                }
            });
        }
    }
}
```

### 5.1.3 login.java

```
package com.example.digitalnoticeboard;

import ...

public class login extends AppCompatActivity {

    EditText e1,e2;
    Button btn1;
    SharedPreferences sharedPreferences;
    String url="";

    @SuppressWarnings("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        e1=(EditText)findViewById(R.id.username);
        e2=(EditText)findViewById(R.id.pswd);
        btn1=(Button) findViewById(R.id.login);

        sharedPreferences=getSharedPreferences( "name: ", Context.MODE_PRIVATE);
        url="http://"+sharedPreferences.getString( "ip", " ")+":5000/login";

        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
```

```
                SharedPreferences.Editor edt = sharedPreferences.edit();
                edt.putString( "lid", lid);
                edt.commit();

                if (type.equalsIgnoreCase( anotherString: "admin")) {

                    startActivity(new Intent(getApplicationContext(),adminhome.class));
                    Toast.makeText(getApplicationContext(), text: "Welcome Admin", Toast.LENGTH_LONG).show();

                }

                else {
                    startActivity(new Intent(getApplicationContext(),upload1.class));
                    Toast.makeText(getApplicationContext(), text: "Welcome", Toast.LENGTH_LONG).show();
                }

            }

            else {
                Toast.makeText(getApplicationContext(), text: "You entered an invalid Email or Password.. !! please check & try again",
            }

        } catch (Exception e) {
            Log.d( tag: "cfccc", e.getMessage());
        }

    }

}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Toast.makeText(getApplicationContext(), error.getMessage(), Toast.LENGTH_LONG).show();
    }
}
```



### 5.1.4 Register.java

```
package com.example.digitalnoticeboard;

import ...

public class login extends AppCompatActivity {

    EditText e1,e2;
    Button btn1;
    SharedPreferences sharedPreferences;
    String url="";

    @SuppressWarnings("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        e1=(EditText)findViewById(R.id.username);
        e2=(EditText)findViewById(R.id.pswd);
        btn1=(Button) findViewById(R.id.login);

        sharedPreferences=getSharedPreferences( name: "", Context.MODE_PRIVATE);
        url="http://"+sharedPreferences.getString( s: "ip", s1: "")+":5000/login";

        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
```

```
                SharedPreferences.Editor edt = sharedPreferences.edit();
                edt.putString( s: "lid", lid);
                edt.commit();

                if (type.equalsIgnoreCase( anotherString: "admin")) {

                    startActivity(new Intent(getApplicationContext(),adminhome.class));
                    Toast.makeText(getApplicationContext(), text: "Welcome Admin", Toast.LENGTH_LONG).show();

                }

                else {
                    startActivity(new Intent(getApplicationContext(),upload1.class));
                    Toast.makeText(getApplicationContext(), text: "Welcome", Toast.LENGTH_LONG).show();
                }

            }

            else {
                Toast.makeText(getApplicationContext(), text: "You entered an invalid Email or Password.. !! please check & try again",
            }

        } catch (Exception e) {
            Log.d( tag: "cfccc", e.getMessage());
        }

    }

}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Toast.makeText(getApplicationContext(), error.getMessage(), Toast.LENGTH_LONG).show();
    }
}
```

### 5.1.5 upload.java

```
package com.example.digitalnoticeboard;

import ...

public class upload1 extends AppCompatActivity {
    Button btn1,btn2,btn3,btn4;

    @SuppressWarnings("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_upload1);
        btn1=(Button) findViewById(R.id.photos);
        btn2=(Button) findViewById(R.id.video);
        btn3=(Button) findViewById(R.id.audio);
        btn4=(Button) findViewById(R.id.doc);
        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(getApplicationContext(),image.class));
            }
        });
        btn2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(getApplicationContext(),video.class));
            }
        });
        btn3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(getApplicationContext(),audio.class));
            }
        });
        btn4.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(getApplicationContext(),document.class));
            }
        });
    }
}
```

### 5.1.6 adminhome.java

```
package com.example.digitalnoticeboard;

import ...

public class adminhome extends AppCompatActivity {

    Button b1,b2,b3;

    @SuppressWarnings("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_adminhome);

        b1=(Button) findViewById(R.id.button3);
        b2=(Button) findViewById(R.id.button4);
        b3=(Button) findViewById(R.id.button5);

        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(getApplicationContext(),Register.class));
            }
        });
        b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new Intent(getApplicationContext(),teacher.class));
            }
        });

        b3.setOnClickListener(new View.OnClickListener() {
            @Override
```

## 5.1.7 video.java

```
package com.example.digitalnoticeboard;

import ...

public class video extends AppCompatActivity {
    Button b1,b2;
    EditText ed1;
    SharedPreferences sh;
    static String attach,path="";
    byte[] videoBytes=null;
    String video,photo="",url="";

    private File outputFile=null;

    private static final int REQUEST_CODE_SELECT_VIDEO=1;
    private String videoFilePath=null;
    private String base64EncodedString =null;

    RequestQueue rq;

    private String pdfFilePath=null;
    // private String base64EncodedString = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_video);
        b1=(Button) findViewById(R.id.browse);
        b2=(Button) findViewById(R.id.send);
        ed1=(EditText) findViewById(R.id.videoloc);

        sh=getSharedPreferences( name: "", Context.MODE_PRIVATE);
        url="http://"+sh.getString( s: "ip", s1: "")+":5000/addVideo";
    }
}
```

```
    if (requestCode == REQUEST_CODE_SELECT_VIDEO && resultCode == RESULT_OK && data != null) {
        handleSelectedVideo(data.getData());
    }
}

private void reg(){
    StringRequest stringRequest = new StringRequest(Request.Method.POST, url,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try
                {
                    JSONObject job=new JSONObject(response);
                    String status=job.getString( name: "status");
                    Toast.makeText(getApplicationContext(), job.toString(), Toast.LENGTH_SHORT).show();

                    if (status.equalsIgnoreCase( anotherString: "success")){
                        Toast.makeText(getApplicationContext(), text: "Successfully saved",Toast.LENGTH_LONG).show();
                        Intent i=new Intent(getApplicationContext(),login.class);
                        startActivity(i);
                    }
                    else
                    {
                        Toast.makeText(getApplicationContext(), text: "No Data",Toast.LENGTH_LONG).show();
                    }
                } catch (JSONException e) {
                    Toast.makeText(getApplicationContext(), text: "Error..1..." +e.toString(), Toast.LENGTH_SHORT).show();
                }
            }
        }
    ),
}
```

## 5.1.8 image.java

```
package com.example.digitalnoticeboard;

import ...

public class image extends AppCompatActivity {
    Button b1,b2;
    EditText ed1;
    ImageView im1;
    String photo="";
    SharedPreferences sh;
    RequestQueue rq;
    byte[] photoBytes=null;
    private final int RESPONSE_OK=200;
    private static final int CAMERA_REQUEST=1888,GALLERY_CODE=201;
    private Uri mImageCaptureUri;
    private File outputFile=null;
    static String path,attach,url="";
    private static final int MY_CAMERA_PERMISSION_CODE=100;

    @SuppressWarnings("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_image);
        b1=(Button) findViewById(R.id.browse);
        b2=(Button) findViewById(R.id.send);
        ed1=(EditText) findViewById(R.id.name);
        im1=(ImageView) findViewById(R.id.imageView);
        sh=getSharedPreferences( name: "", Context.MODE_PRIVATE);
        url="http://"+sh.getString( s: "ip", s1: "")+":5000/addImage";
        rq= Volley.newRequestQueue( context: this);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
public Uri getImageUri(Context inContext, Bitmap inImage) {
    ByteArrayOutputStream bytes = new ByteArrayOutputStream();
    inImage.compress(Bitmap.CompressFormat.JPEG, quality: 100, bytes);
    String path = MediaStore.Images.Media.insertImage(inContext.getContentResolver(), inImage, title: "Title", description: null);
    return Uri.parse(path);
}

private String getRealPathFromURI(Uri contentURI) {
    Cursor cursor = getContentResolver()
        .query(contentURI, projection: null, selection: null, selectionArgs: null, sortOrder: null);
    if (cursor == null)
        path=contentURI.getPath();

    else {
        cursor.moveToFirst();
        int idx = cursor.getColumnIndex(MediaStore.Images.ImageColumns.DATA);
        path=cursor.getString(idx);
    }
    if(cursor!=null)
        cursor.close();
    return path;
}

private Bitmap decodeFile(File f) {
    try {
        // decode image size
        BitmapFactory.Options o = new BitmapFactory.Options();
        o.inJustDecodeBounds = true;
        BitmapFactory.decodeStream(new FileInputStream(f), outPadding: null, o);

        // Find the correct scale value. It should be the power of 2.
        final int REQUIRED_SIZE = 512;
```

## 5.1.9 audio.java

```
package com.example.digitalnoticeboard;

import ...

public class audio extends AppCompatActivity {
    Button b1,b2;
    EditText ed1;
    SharedPreferences sh;
    RequestQueue rq;
    private static final int REQUEST_CODE_SELECT_AUDIO=2;
    private String audioFilePath=null;
    private String base64EncodedString=null;
    String url="";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_audio);
        b1=(Button) findViewById(R.id.browse);
        b2=(Button) findViewById(R.id.send);
        ed1=(EditText) findViewById(R.id.name);

        sh=getSharedPreferences( name: "", Context.MODE_PRIVATE);
        url="http://"+sh.getString( s: "ip", s1: "")+":5000/addAudio";
        Toast.makeText(getApplicationContext(),url,Toast.LENGTH_LONG).show();
        rq= Volley.newRequestQueue( context: this);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) { showAudioChooser(); }
        });
        b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) { reg(); }
        });
    }
}
```

```
private void reg(){
    StringRequest stringRequest = new StringRequest(Request.Method.POST, url,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try
                {
                    JSONObject job=new JSONObject(response);
                    String status=job.getString( name: "status");
                    Toast.makeText(getApplicationContext(), job.toString(), Toast.LENGTH_SHORT).show();

                    if (status.equalsIgnoreCase( anotherString: "success")){
                        Toast.makeText(getApplicationContext(), text: "Successfully saved",Toast.LENGTH_LONG).show();
                        Intent i=new Intent(getApplicationContext(),upload1.class);
                        startActivity(i);
                    }
                    else
                    {
                        Toast.makeText(getApplicationContext(), text: "No Data",Toast.LENGTH_LONG).show();
                    }
                } catch (JSONException e) {
                    Toast.makeText(getApplicationContext(), text: "Error..1..." +e.toString(), Toast.LENGTH_SHORT).show();
                }
            }
        },
        new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(getApplicationContext(), text: "Error..1..." +error.toString(), Toast.LENGTH_SHORT).show();
            }
        }
    );
}
```

### 5.1.10 document.java

```
package com.example.digitalnoticeboard;

import ...

public class document extends AppCompatActivity {
    Button b1,b2;
    EditText ed1;
    SharedPreferences sh;
    RequestQueue rq;
    private static final int REQUEST_CODE_SELECT_PDF=1;
    private String pdfFilePath=null;
    private String base64EncodedString=null;
    String url="";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_document);

        b1=(Button) findViewById(R.id.browse);
        b2=(Button) findViewById(R.id.send);
        ed1=(EditText) findViewById(R.id.ed11223);

        sh=getSharedPreferences( name: "", Context.MODE_PRIVATE);
        url="http://"+sh.getString( s: "ip", s1: "")+":5000/addPdf";

        rq = Volley.newRequestQueue( context: this);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) { selectPdf(); }
        })
    }
}
```

```
}
private void reg(){
    StringRequest stringRequest = new StringRequest(Request.Method.POST, url,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try
                {
                    JSONObject job=new JSONObject(response);
                    String status=job.getString( name: "status");
                    Toast.makeText(getApplicationContext(), job.toString(), Toast.LENGTH_SHORT).show();

                    if (status.equalsIgnoreCase( anotherString: "success")){
                        Toast.makeText(getApplicationContext(), text: "Successfully saved",Toast.LENGTH_LONG).show();
                        Intent i=new Intent(getApplicationContext(),login.class);
                        startActivity(i);
                    }
                    else
                    {
                        Toast.makeText(getApplicationContext(), text: "No Data",Toast.LENGTH_LONG).show();
                    }
                } catch (JSONException e) {
                    Toast.makeText(getApplicationContext(), text: "Error..1..." +e.toString(), Toast.LENGTH_SHORT).show();
                }
            }
        }
    ),
    new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Toast.makeText(getApplicationContext(), text: "Error..1..." +error.toString(), Toast.LENGTH_SHORT).show();
        }
    }
}
```

### 5.1.11 teacher.java

```
package com.example.digitalnoticeboard;

import ...

public class teacher extends AppCompatActivity {

    private Handler handler;
    ListView lv;
    SharedPreferences sharedPreferences;
    String url = "";

    //    JSONParser parser = new JSONParser();

    ArrayList<String> name,gender,email,phone,department,password;

    @SuppressWarnings("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_teacher);

        lv = (ListView) findViewById(R.id.listView);

        sharedPreferences=getSharedPreferences( name: "", Context.MODE_PRIVATE);
        url="http://"+sharedPreferences.getString( s: "ip", s1: "")+":5000/view";
        handler = new Handler();
        updateDateDisplay();

        try {
            if (android.os.Build.VERSION.SDK_INT > 9) {
                StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
                StrictMode.setThreadPolicy(policy);
            }
        }

        public void onResponse(String response) {
            try {
                JSONArray jsonArray=new JSONArray(response);
                if (jsonArray.length() > 0) {

                    name = new ArrayList<String>();
                    gender = new ArrayList<String>();
                    email= new ArrayList<String>();
                    phone = new ArrayList<String>();
                    department = new ArrayList<String>();
                    password = new ArrayList<String>();

                    for (int i = 0; i < jsonArray.length(); i++) {
                        JSONObject jo = jsonArray.getJSONObject(i);
                        name.add(jo.getString( name: "name"));
                        gender.add(jo.getString( name: "gender"));
                        email.add(jo.getString( name: "email"));
                        phone.add(jo.getString( name: "phone"));
                        department.add(jo.getString( name: "department"));
                        password.add(jo.getString( name: "password"));
                    }

                    lv.setAdapter((ListAdapter) new custom(getApplicationContext(),name,gender,email,phone,department,password));
                    Toast.makeText(getApplicationContext(),response, Toast.LENGTH_LONG).show();
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
```



## 5.2 Front End

### 5.2.1 IP address.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_gravity="center"
    android:background="@drawable/ip_address"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ip_address">

    <TextView
        android:id="@+id/textView5"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="280dp"
        android:layout_marginRight="20dp"
        android:height="30dp"
        android:paddingLeft="10dp"
        android:text="IP address"
        android:textColor="@color/white"
        android:textSize="20dp" />

    <EditText
        android:id="@+id/editTextTextPersonName"
        android:layout_width="match_parent"
        android:layout_height="50dp"
        android:ems="10"
        android:layout_marginTop="20dp"
```

### 5.2.2 login.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    tools:context=".login"
    android:background="@drawable/welcmnasra">

    <EditText
        android:id="@+id/username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="100dp"
        android:layout_marginLeft="60dp"
        android:layout_marginTop="100dp"
        android:layout_marginEnd="100dp"
        android:drawableLeft="@drawable/ic_baseline_person_24"
        android:hint="username"
        android:textColor="@color/white"
        android:textColorHint="@color/white" />

    <EditText
        android:id="@+id/pswd"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/username"
```

### 5.2.3 register.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/register"
    tools:context=".Register">
```

```
    <TextView
        android:id="@+id/register"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:fontFamily="@font/amaranth"
        android:text="Create Account"
        android:textSize="30dp"
        android:textStyle="bold"
        android:layout_marginTop="50dp"/>
```

```
    <EditText
        android:id="@+id/name"
        android:paddingLeft="10dp"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:layout_below="@+id/register"
        android:layout_marginHorizontal="50dp"
        android:layout_marginTop="30dp"
        android:background="@drawable/input_bg"
        android:ems="10"
```

```
    <Spinner
        android:id="@+id/dpt"
        android:layout_below="@+id/phone"
        android:layout_marginTop="30dp"
        android:layout_marginHorizontal="50dp"
        android:layout_width="match_parent"
        android:layout_height="45dp" />
```

```
    <EditText
        android:id="@+id/editTextTextPassword"
        android:paddingLeft="10dp"
        android:inputType="textPassword"
        android:layout_below="@+id/dpt"
        android:background="@drawable/input_bg"
        android:layout_marginHorizontal="50dp"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:ems="10"
        android:hint="create a new password"
        android:textColorHint="@color/white"
        android:layout_marginTop="30dp"
        android:paddingRight="10dp"
        android:drawableRight="@drawable/ic_baseline_remove_red_eye_24"/>
```

```
    <Button
        android:id="@+id/button2"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:layout_below="@+id/editTextTextPassword"
        android:layout_marginHorizontal="50dp"
        android:layout_marginTop="20dp"
```

### 5.2.4 adminhome.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:background="@drawable/register"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".adminhome">

    <Button
        android:id="@+id/button3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="20dp"
        android:layout_marginRight="20dp"
        android:backgroundTint="#605050"
        android:text="teacher register" />

    <Button
        android:id="@+id/button4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginRight="20dp"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="30dp"
        android:text="view teacher"
        android:backgroundTint="#605050" />
```

### 5.2.5 upload.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".upload1"
    android:background="@drawable/upload">

    <Button
        android:id="@+id/photos"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="200dp"
        android:layout_weight="1"
        android:backgroundTint="#686161"
        android:paddingLeft="10dp"
        android:layout_marginHorizontal="100dp"
        android:text="image" />

    <Button
        android:id="@+id/video"
        android:layout_below="@+id/photos"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="30dp"
        android:layout_marginHorizontal="100dp"
        android:layout_centerHorizontal="true"
```

## 5.2.6 video.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".image"
    android:background="@drawable/register">

    <Button
        android:id="@+id/browse"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="250dp"
        android:layout_weight="1"
        android:backgroundTint="#686161"
        android:paddingLeft="10dp"
        android:layout_marginHorizontal="100dp"
        android:text="Browse" />

    <EditText
        android:id="@+id/videoLoc"
        android:paddingLeft="10dp"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:layout_below="@+id/browse"
        android:layout_marginHorizontal="50dp"
        android:layout_marginTop="30dp"
        android:background="@drawable/input_bg"
        android:ems="10"
```

## 5.2.7 image.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".image"
    android:background="@drawable/register">

    <Button
        android:id="@+id/browse"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="250dp"
        android:layout_weight="1"
        android:backgroundTint="#686161"
        android:paddingLeft="10dp"
        android:layout_marginHorizontal="100dp"
        android:text="Browse" />

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/send"
        android:id="@+id/imView"
        tools:srcCompat="@tools:sample/avatars" />

    <EditText
        android:id="@+id/name"
        android:paddingLeft="10dp"
        android:layout_width="match_parent"
```

## 5.2.8 audio.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".audio"
    android:background="@drawable/register">

    <Button
        android:id="@+id/browse"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="250dp"
        android:layout_weight="1"
        android:backgroundTint="#686161"
        android:paddingLeft="10dp"
        android:layout_marginHorizontal="100dp"
        android:text="Browse" />

    <EditText
        android:id="@+id/name"
        android:paddingLeft="10dp"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:layout_below="@+id/browse"
        android:layout_marginHorizontal="50dp"
        android:layout_marginTop="30dp"
        android:background="@drawable/input_bg"
        android:ems="10"
```

## 5.2.9 document.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".document"
    android:background="@drawable/register">

    <Button
        android:id="@+id/browse"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="250dp"
        android:layout_weight="1"
        android:backgroundTint="#686161"
        android:paddingLeft="10dp"
        android:layout_marginHorizontal="100dp"
        android:text="Browse" />

    <EditText
        android:id="@+id/ed11223"
        android:paddingLeft="10dp"
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:layout_below="@+id/browse"
        android:layout_marginHorizontal="50dp"
        android:layout_marginTop="30dp"
        android:background="@drawable/input_bg"
        android:ems="10"
```

## 5.2.10 teacher.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <HorizontalScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical">

            <TableRow
                android:layout_width="match_parent"
                android:layout_height="50dp"
                android:layout_marginLeft="10dp"
                android:gravity="center_vertical">

                <TextView
                    android:id="@+id/name"
                    android:layout_width="90dp"
                    android:layout_height="wrap_content"
                    android:text="Name"
                    android:textAlignment="center" />

                <TextView
                    android:id="@+id/gender"
                    android:layout_width="90dp"
                    android:layout_height="wrap_content"
                    android:text="Gender"
                    android:layout_marginLeft="10dp" />
            </TableRow>
        </LinearLayout>
    </HorizontalScrollView>
</ScrollView>
```

```

        <TextView
            android:id="@+id/email"
            android:layout_width="90dp"
            android:layout_height="wrap_content"
            android:text="Email"
            android:layout_marginLeft="10dp"
            android:textAlignment="center" />

        <TextView
            android:id="@+id/phone"
            android:layout_width="90dp"
            android:layout_height="wrap_content"
            android:text="Phone"
            android:layout_marginLeft="10dp"
            android:textAlignment="center" />

        <TextView
            android:id="@+id/dpt"
            android:layout_width="90dp"
            android:layout_height="wrap_content"
            android:text="Department"
            android:layout_marginLeft="10dp"
            android:textAlignment="center" />

        <TextView
            android:id="@+id/Password"
            android:layout_width="90dp"
            android:layout_height="wrap_content"
            android:text="Password"
            android:layout_marginLeft="10dp"
            android:textAlignment="center" />
    </TableRow>

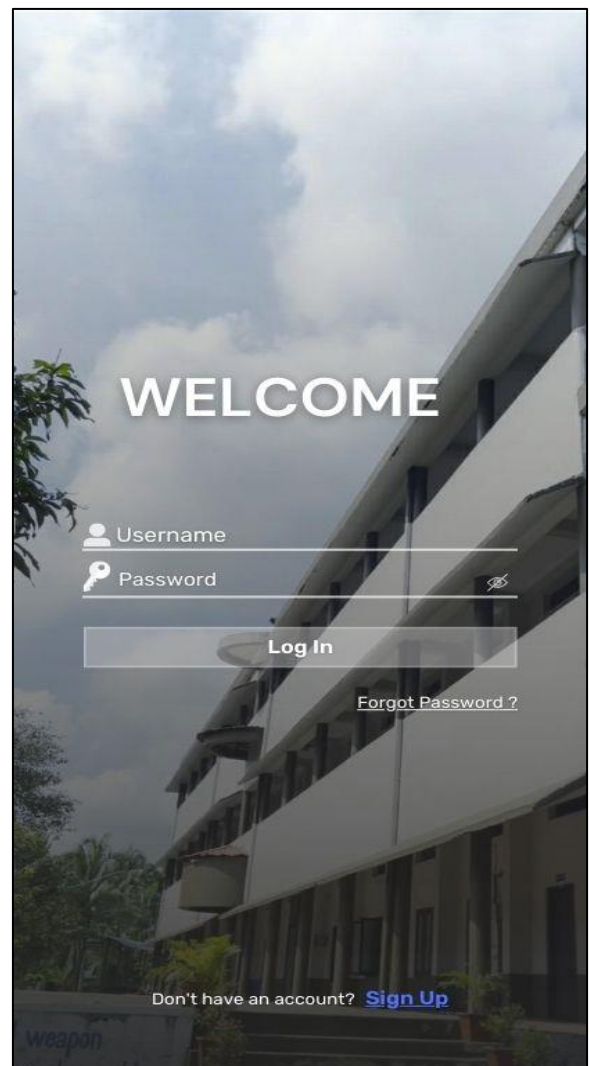
    <TableRow>
```

## 5.3 Results

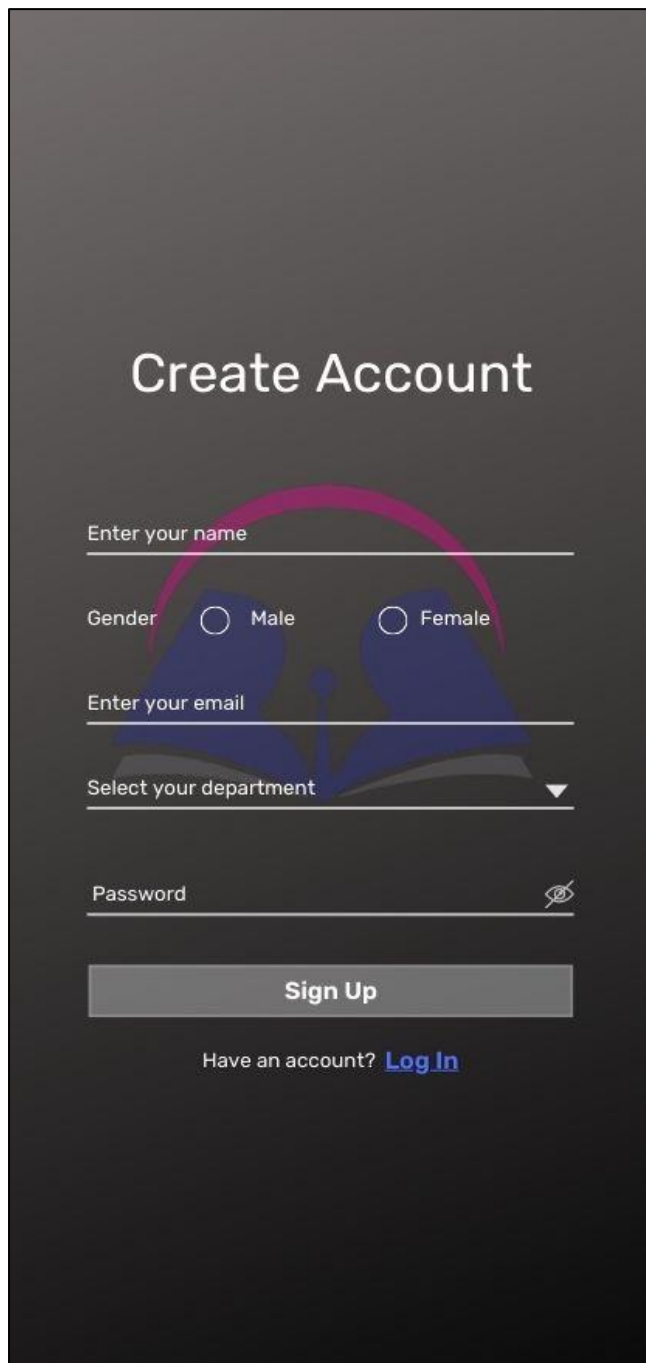
### 5.3.1 IP Address



### 5.3.2 Login



### 5.3.3 Register



The registration form is titled "Create Account" in a large, white, sans-serif font. Below the title, there are four input fields: "Enter your name", "Enter your email", "Select your department" (with a dropdown arrow), and "Password" (with an eye icon for toggling visibility). The "Gender" section has two radio buttons labeled "Male" and "Female". A "Sign Up" button is positioned below the password field. At the bottom, there is a link "Have an account? Log In" in blue text. A faint background watermark of an open book is visible.


Create Account

Enter your name

Gender ☐ Male ☐ Female

Enter your email

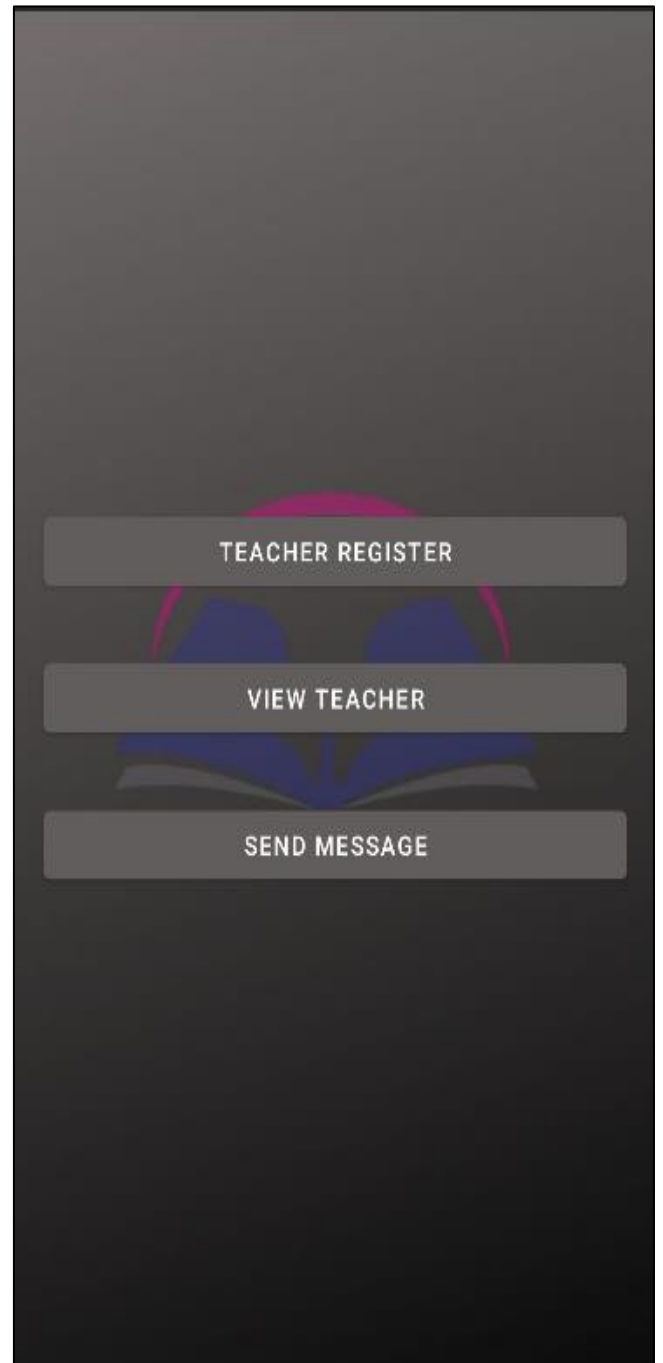
Select your department ▼

Password 

Sign Up

Have an account? [Log In](#)

### 5.3.4 Admin Home



The Admin Home dashboard features three large, light gray buttons stacked vertically. The buttons are labeled "TEACHER REGISTER", "VIEW TEACHER", and "SEND MESSAGE" in a bold, black, sans-serif font. A faint background watermark of an open book is visible.


TEACHER REGISTER

VIEW TEACHER

SEND MESSAGE



### 5.3.5 Video



**STUDIOS**

SCENE	TAKE	ROLL
DATE	SOUND	
PROD. CO.		
DIRECTOR		
CAMERAMAN		


## Video

**browse**

upload your video here

**send**

### 5.3.6 Images




## Images

**browse**

upload your images here

**send**

### 5.3.7 Audio




## Audio

**browse**

upload your audio here

**send**

### 5.3.8 Document



## Document

**browse**

upload your doc here

**send**

### 5.3.9 View Teacher

Name	Gender	Email	Phone	Depa
Item 1 Sub Item 1				
Item 2 Sub Item 2				
Item 3 Sub Item 3				
Item 4 Sub Item 4				
Item 5 Sub Item 5				
Item 6 Sub Item 6				
Item 7 Sub Item 7				

# **CHAPTER 6**

# **TESTING**

## **6.1 System Testing**

The purpose of the system testing is to ensure that the digital notice board Android application meets the functional and non-functional requirements specified in the project requirements document.

Scope:

The system testing will cover the following areas of the digital notice board Android application:

- Uploading and displaying text messages
- Uploading and displaying images
- Uploading and displaying videos
- Uploading and playing audio recordings
- Scheduling posts for specific dates and times
- Allowing students to comment or reply to posts

### **6.1.1 Test Cases:**

#### **1. Uploading and displaying text messages**

Verify that teachers can successfully create and upload text messages to the notice board.

Verify that students can view the text messages on the notice board.

#### **2. Uploading and displaying images**

Verify that teachers can successfully upload images to the notice board.

Verify that students can view the images on the notice board.

#### **3. Uploading and displaying videos**

Verify that teachers can successfully upload videos to the notice board.

Verify that students can view the videos on the notice board.

#### **4. Uploading and playing audio recordings**

Verify that teachers can successfully upload audio recordings to the notice board.

Verify that students can play the audio recordings on the notice board.

Scheduling posts for specific dates and times

Verify that teachers can successfully schedule posts for specific dates and times.

Verify that the posts are displayed on the notice board at the scheduled date and time.

#### 5. Allowing students to comment or reply to posts

Verify that students can successfully comment or reply to posts on the notice board.

Verify that teachers can view the comments and replies on the notice board.

## **6.2 Testing Strategy**

The purpose of the testing strategy is to define the overall approach and plan for testing the digital notice board Android application. The testing strategy will cover the following types of testing: unit testing, integration testing, system testing, and acceptance testing.

Objectives:

- To ensure that the digital notice board Android application meets the functional and non-functional requirements specified in the project requirements document.
- To identify and eliminate defects and issues in the digital notice board Android application.
- To ensure that the digital notice board Android application is stable, secure, and reliable.
- To ensure that the digital notice board Android application performs well and is scalable.

### **6.2.1 Unit Testing**

Unit testing will be performed by developers during the development process to ensure that individual components work as expected.

### **6.2.2 Integration Testing**

Integration testing will be performed by developers to ensure that the components of the digital notice board Android application work together as intended.

### **6.2.3 System Testing**

System testing will be performed by a team of testers who will use the digital notice board Android application according to the requirements outlined in the project requirements document.

### **6.2.4 Acceptance Testing**

Acceptance testing will be performed by the user or client to ensure that the digital notice board Android application meets their requirements and expectations.

The testing process will be repeated until all test cases have been executed and all issues or defects have been resolved.

## **6.3 Testing Environment**

The testing environment will include the following:

- Android Studio IDE
- Emulator for different Android devices
- Physical Android devices if available
- Testing tools such as JUnit, Espresso, and Mockito

# **CHAPTER 7**

## **SYSTEM IMPLEMENTATION**



## 7. System Implementation

System implementation is the process of transforming the design of a system into a working product. In the context of our "Digital Notice Board" project, the system implementation phase involves creating the actual Android application and setting up the back-end infrastructure.

### Overview

- **Set up the development environment:** Install and configure the necessary software and tools for Android app development, such as Android Studio, Java Development Kit (JDK), and Android SDK.
- **Develop the front-end:** Use Android Studio to create the user interface (UI) for the Digital Notice Board app. This includes designing screens, creating layouts, and adding functionality to enable the user to interact with the app.
- **Develop the back-end:** Create the back-end infrastructure using MySQL, Python, and/or Java. This includes setting up the database, creating server-side scripts for handling requests, and implementing the required algorithms and logic.
- **Integrate the front-end and back-end:** Connect the front-end UI with the back-end infrastructure. This involves making API calls to the server to fetch data and display it on the app. You will also need to enable users to submit data, such as uploading notices and attachments to the server.
- **Test the system:** Conduct thorough testing to ensure that the system is functioning as expected. This includes unit testing, integration testing, and system testing.
- **Deploy the system:** Once the testing phase is complete, deploy the system to the target environment. This may involve setting up a web server, configuring database access, and performing other tasks necessary to get the system up and running in production.

# **CHAPTER 8**

# **LIMITATIONS**

## **8. LIMITATIONS**

- **Hardware and Software Requirements:** The digital notice board Android application may require specific hardware or software configurations to run effectively, which could limit its use on certain devices or platforms.
- **Internet Connectivity:** The application requires internet connectivity to upload and download messages. Poor internet connectivity or lack of internet access in some areas may limit the use of the application.
- **User Adoption:** The success of the digital notice board Android application relies on user adoption. If users do not find the application useful or easy to use, they may not use it, which could limit the impact of the application.
- **Data Privacy and Security:** The application may collect sensitive data, such as user login credentials, which could potentially be compromised. If the application does not implement adequate security measures, it could limit the trust that users have in the application.
- **Maintenance and Support:** The digital notice board Android application requires ongoing maintenance and support to ensure that it functions correctly and remains up-to-date with the latest technology trends. Without proper maintenance and support, the application may become outdated and unusable.
- **Resource Constraints:** The development and deployment of the digital notice board Android application may require significant resources, such as time, personnel, and funding. Resource constraints could limit the scope or quality of the application.

# **CHAPTER 9**

## **FUTURE ENHANCEMENT**

## **9. Future Enhancement**

The “Digital Notice Board” project has some potential features. Some of the future enhancement of the project are:

- ❖ **Push Notifications:** Implement push notifications to alert users about new messages on the notice board. This will make it easier for users to stay up-to-date with important information.
- ❖ **Multilingual Support:** Add multilingual support to the application to make it more accessible to users who speak different languages.
- ❖ **Accessibility Features:** Implement accessibility features such as screen readers, larger fonts, and high-contrast themes to make the application more user-friendly for individuals with disabilities.
- ❖ **Chatbot Integration:** Add a chatbot feature to the application to enable users to ask questions and receive immediate assistance. This will help to improve user experience and reduce the workload of the support team.
- ❖ **Offline Support:** Add offline support to the application to allow users to view previously downloaded messages even when they don't have an internet connection.
- ❖ **Artificial Intelligence:** Integrate artificial intelligence to enable smart recommendations for messages and to provide personalized experiences for users.
- ❖ **Integration with Other Systems:** Integrate the digital notice board Android application with other systems, such as learning management systems, to improve communication and streamline workflows.

# **CHAPTER 10**

# **CONCLUSION**

## **10. CONCLUSION**

The “Digital Notice Board” Android application project is a valuable tool that can significantly improve communication between teachers and students in educational institutions. The application allows teachers to upload text, videos, audio, and images, which students can access from their mobile devices.

Throughout the development of this project, we have explored various aspects such as system requirements, design, implementation, testing, and limitations. We have also identified potential future enhancements to improve the functionality and usability of the application.

The digital notice board Android application has the potential to revolutionize the way educational institutions communicate with their students. By providing a centralized platform for communication, the application can help to streamline workflows, reduce administrative overheads, and improve collaboration between teachers and students.

Overall, the digital notice board Android application project has been an exciting and challenging journey, and we hope that it will be a useful tool for educational institutions and the wider community. We look forward to continued development and improvements to make the application even more effective and user-friendly.

# **CHAPTER 11**

## **BIBLIOGRAPHY**



# BIBLIOGRAPHY

## BOOKS

- Database System Concepts – Henry. F. Korth(4<sup>th</sup> Edition).
- Elements of System Analysis And Design – Marvin Gore and John W Stubbe.
- System Analysis and Design – Robert. E. Leslie.
- SQL The Complete reference, 3<sup>rd</sup> Edition by James R Groff (Author), Paul N. Weinberg (Author), and Oppel (Author) Paperback – Import,1<sup>st</sup> Sept 2009.

## WEBSITES

- [www.google.com](http://www.google.com)
- <https://www.wikipedia.org/>
- <https://codepen.io/>
- <https://www.coursera.org/in>
- <https://www.w3schools.com>

