CHAPTER 1

INTRODUCTION

This chapter gives an overview of the College App project, including the background, problem statement, objectives, and scope. The Admin App is a platform designed for college administrators to upload notices, images, and PDFs. The app includes functionalities like updating and deleting faculty and notices, and managing the college's image Gallery. The Student App is designed for students to access information related to their college. It includes functionalities like user registration and login, password retrieval, and various fragments showcasing college information such as notice, faculty, gallery, about, videos lectures, Ebooks, website, and more.

1.1 PROJECT AIMS AND OBJECTIVES

The Project aims and objectives that will be achieved after completion of this project are discussed in this Subchapter. The aims and objectives are as follows:

- Design and develop an Admin App for efficient communication of college-related information to the community.
- Design and develop a Student App for easy access to essential college-related information and resources.
- Provide an authentication system for secure access and management of college-related information in the Admin App.
- Provide a user registration and login system for secure access to college-related information and resources in the Student App.
- Provide functionalities for the Admin App, including uploading notices, images, and PDFs, managing the college's image gallery, and updating and deleting faculty and notices.
- Provide functionalities for the Student App, including access to college-related information such as notice, faculty, gallery, about, videos lectures, ebooks, website, and more.
- Create a responsive user interface for both the Admin App and Student App for a seamless user experience across different devices.
- Ensure the security and privacy of user data by implementing appropriate security measures.

1.2 BACKGROUND OF PROJECT

- The traditional methods of communication in colleges and universities, such as email and notice boards, have limitations in terms of accessibility, timeliness, and accuracy.
- An efficient and effective communication system is essential for colleges and universities to provide timely and accurate information to students, faculty, staff, and administrators.
- This project aims to develop an Admin App and Student App that provide an easy-to-use, efficient, and secure platform for college administrators to upload and manage college-related information and for students to access and view this information.
- The Admin App provides an interface for college administrators to upload and manage college-related information, including notices, images, and PDFs.
- The Student App provides an interface for students to access and view college-related information, including notices and PDFs uploaded by college administrators to Firebase.
- The use of Firebase for storing college-related information ensures that the information is accessible and searchable by authorized users, thereby enhancing the communication system and the learning experience for college students.
- The use of traditional communication methods such as notice boards, flyers, and emails, can be tedious and time-consuming for college administrators to disseminate information.
- With the help of Firebase cloud storage, students can access college-related information such as notices and PDFs on the go, and stay informed about the latest developments in their college.
- The Admin App and Student App are designed to provide a seamless user experience for administrators and students, with an intuitive interface that is easy to navigate and use.
- The Admin App and Student App leverage the power of cloud computing to provide real-time access to college-related information, making it easier for students to stay up-to-date with college events, news, and important notices.

1.3 OPERATION ENVIRONMENT

Category	Admin App	Student App
Processor	1.2 GHz dual-core processor or higher	1.4 GHz quad-core processor or higher
Operating System	Android 7.0 (Nougat) or higher	Android 6.0 (Marshmallow) or higher
Memory	Minimum of 2 GB RAM	Minimum of 1 GB RAM
Hard Disk Space	At least 100 MB free space	At least 50 MB free space
Database	Firebase Realtime Database	Firebase Realtime Database
Languages	Java, XML	Java, XML
Internet Connection	Required	Required
Mobile-friendly	Yes	Yes
Testing	Unit testing and integration testing using JUnit	Unit testing and integration testing using JUnit
Scalability	App should be scalable for future enhancements and updates	App should be scalable for future enhancements and updates
Security	User authentication with email and password, database security measures	User authentication with email and password, database security measures
Performance	Fast loading and smooth navigation of the app	Fast loading and smooth navigation of the app
User Interface	Easy to use interface with intuitive design	Easy to use interface with intuitive design
Social Interaction	None	Share and rate the app feature
Integration	Firebase services integration (Analytics, Crashlytics, etc.)	Firebase services integration (Analytics, Crashlytics, etc.)

1.4 PROBLEM STATEMENT

- Lack of efficient communication between the college administration and students.
- Inefficient dissemination of information such as notices and updates to students.
- Time-consuming and outdated methods of sharing information with students.
- Difficulty in maintaining up-to-date faculty information.
- Limited access to college resources and information by students.
- Absence of a centralized platform for students to access college information.
- Inconvenience in accessing information such as class schedules and faculty details.
- Difficulty in tracking and managing attendance of students.
- Lack of a platform for students to provide feedback and suggestions to college administration.
- Need for a user-friendly and mobile-friendly platform for students to access college information.
- Difficulty in managing and organizing college events and activities.
- Lack of transparency in college administration and decision-making processes.
- Inefficient utilization of college resources and facilities.
- Inability to track academic progress and performance of students.
- Difficulty in managing and tracking student records and information.
- Inconvenience in accessing and submitting academic documents and forms.
- Need for a platform to facilitate online discussions and collaboration among students.
- Difficulty in managing and monitoring student behavior and disciplinary issues.
- Inability to provide personalized and customized information to students based on their interests and preferences.

1.5 SCOPE ON THE PROJECT

1. Development of an Admin app that will allow the admin to securely login using their email and password to upload notices, gallery images, and PDFs.

2. The Admin app will also include the functionality to update and delete faculty and notices as per the requirements.

3. The Admin app will be developed using Java language and Android Studio.

4. Development of a Student app that will allow the students to register and fill in their details like name, email, and password, which will be securely stored in Firebase.

5. The Student app will have a login functionality for the registered students, and they can also reset their password in case they forget it.

6. The Student app will display information about the college, such as its departments, contact details, and location on the map, through the home fragment.

7. The notice fragment will show the latest notices uploaded by the admin on top for the students to access.

8. The faculty fragment will display the information about the faculty members uploaded by the admin, such as their name, post, email, and profile photo.

9. The gallery fragment will display the images uploaded by the admin in different sections like sports activities, cultural events, and other events, for the students to view.

10. The about fragment will show the information about the college, its available branches and streams like civil, computer science, and mechanical, and other details.

11. The navigation drawer will allow the students to access other features like video lectures, ebooks, the college website, and the option to share and rate the app.

12. The project will integrate with Firebase, allowing for easy storage and retrieval of data, and enabling real-time updates to the app.

13. The project will ensure proper authentication and security measures to prevent any unauthorized access to the app.

CHAPTER 2

LITERATURE REVIEW

The literature review included studies and research papers related to mobile app development, Android Studio, Java programming language, Firebase, and other related technologies. It helped in identifying the best practices, tools, and techniques that can be used for the development of the admin and student app.

Overall, the literature review helped in gaining a comprehensive understanding of the app development process, relevant technologies, and best practices, which were used to inform the design and development of the admin and student app.

2.1 INTRODUCTION

The literature review for this project aims to explore existing research and studies related to the development of admin and student mobile applications in the education sector. This will provide insights on the best practices, challenges, and opportunities for building effective and user-friendly apps that meet the needs of college administrators and students. The review will also examine the use of technologies such as Firebase, navigation drawers, and fragments, which are essential components of the apps. By drawing on this literature, we aim to build an app that leverages the latest developments and knowledge in the field.

The Student app is designed to be used by students of the college. It includes a registration process for new users, where they can fill in their details such as name, email, and password, which will be stored on Firebase. Students can log in using their registered email and password, and the app also includes a password reset feature. The app includes several features such as a home screen with a slider of college images, department details, college contact information, and location on the map.

2.2 OVERVIEW

The admin app is a platform where an admin can log in using an email and password to upload notices, images, and PDFs, update or delete faculty information, and delete notices.
The student app is designed for college students who can register using their name, email, and password, which is stored in Firebase. The app features a home fragment that displays college images, departments, contact details, and location on a map. Other features include notice and faculty fragments, gallery fragment, about fragment, and a navigation drawer with video lectures, uploaded PDFs, website of the college, and more.

Together, these two apps provide a comprehensive system for managing college-related information for both the administration and the students. The literature review will explore existing literature and studies related to similar systems and technologies, highlighting their features, advantages, and limitations.

2.3 TECHNICAL ASPECTS

Login authentication: Admin app requires email and password for authentication, which is verified against the registered admin credentials. Student app also requires email and password for authentication, and uses Firebase Authentication to securely store and validate user login information.

Cloud storage: Both admin and student apps use Firebase Cloud Storage to store and manage uploaded files such as notices, images, and PDFs. This allows for efficient retrieval and sharing of data across the apps.

Database management: Both admin and student apps use Firebase Realtime Database to store and manage app data such as faculty information, notice details, and user registration information. This allows for easy retrieval and updating of app data in real time.

User interface: Both admin and student apps use Android Studio and XML to design and implement their user interfaces. The UI is designed to be user-friendly and intuitive, with easy navigation and access to important features and information.

Integration of features: The admin app includes various features such as uploading notices, images, and PDFs, updating and deleting faculty details, and deleting notices. The student app integrates these features by displaying the uploaded data in the relevant sections such as the notice board, faculty section, and gallery.

Third-party libraries: The project may use various third-party libraries to enhance functionality or improve performance. For example, Glide may be used to efficiently load and display images, and Retrofit may be used to make API requests to external services.

Testing: The project may use various testing frameworks such as JUnit and Espresso to ensure that the apps function as intended and are free of bugs or errors. Testing is an important part of the development process to ensure a high-quality final product.

2.4 ADVANTAGES & DISADVANTAGES

	Advantages	**Disadvantages**
1.	Provides a centralized platform for communication	Potential security vulnerabilities
2.	Student app provides easy access to important college information, including notices, faculty details, and location.	Student app requires a stable internet connection to retrieve data from the cloud.
3.	Firebase cloud storage ensures data is accessible from anywhere and can be updated in real-time.	Dependency on Firebase cloud storage may result in data loss or corruption if the servers experience downtime or other issues.
4.	Navigation drawer allows for easy access to various app features, including videos, e-books, and the college website.	The app may consume a significant amount of device storage if many images or PDFs are uploaded.
5.	Admin app provides easy deletion of outdated or incorrect information.	App functionality may be limited if the user's device does not meet the minimum system requirements.

2.5 USER BEHAVIOR AND PREFERENCES

- Users prefer easy and quick login and registration processes.
- Users expect a clean and organized user interface that is easy to navigate.
- Users prioritize the security and privacy of their personal information.
- Users value timely and relevant notifications, such as important college updates and events.
- Users appreciate the ability to customize their profile and preferences.
- Users want easy access to important college resources, such as faculty information and course materials.
- Users prefer a seamless and uninterrupted user experience, without bugs or crashes.
- Users expect timely updates and bug fixes from app developers.
- Users appreciate the ability to provide feedback and suggest improvements to the app.
- Users prefer apps that integrate with other tools and technologies they use, such as social media and email.

2.6 LEGAL AND ETHICAL ISSUES

1. **Data Privacy:** The project involves the collection and storage of personal information of both the admin and the students. It is essential to ensure that the privacy of the user's data is maintained, and proper security measures are in place to protect against any unauthorized access.

2. **Intellectual Property Rights:** The project may involve the use of copyrighted material such as images, videos, and documents. It is important to obtain the necessary permissions and licenses to use such materials.

3. **Cybersecurity:** The project may be vulnerable to cyber attacks, such as data breaches, hacking, and malware. It is important to implement robust security measures to prevent such incidents from occurring.

4. **Accessibility:** The project should be accessible to all users, including those with disabilities. It is important to ensure that the app follows accessibility guidelines to make it easy to use for all users.

5. **Ethical Issues:** The project should not promote any unethical behavior, such as discrimination, harassment, or plagiarism. The app should promote ethical behavior and follow the established norms and values of the institution.

.

CHAPTER 3

METHODOLOGY

This chapter provides a detailed explanation of the methodology used in the development of our College App. The following points will be covered:

- Define the scope of the project and identify the key features for both the admin and student apps.
- Choose appropriate technologies and platforms for development, such as Android Studio for building the apps and Firebase for storing user information.
- Create wireframes and design mockups for both apps to visualize the layout and functionality.
- Implement the admin app, starting with login authentication and then adding functionality for uploading notices, gallery images, PDFs, and faculty information. Also, include delete notice functionality and a logout option.
- Develop the student app, starting with user registration and login authentication. Retrieve data from the Firebase database to display notices, faculty information, and gallery images in the appropriate fragments.
- Test the apps thoroughly to identify and fix any bugs or issues that arise. This includes testing for user authentication, data retrieval, and app functionality.
- Deploy the apps to the Google Play Store or other distribution platforms, and monitor feedback from users to make any necessary updates or improvements.
- Continuously maintain and update the apps as needed, including adding new features and addressing any security concerns.

Overall, this chapter provides a comprehensive overview of the methodology used to develop our College Android App, covering all aspects of the development process from project management to evaluation and feedback.

3.1 PROJECT MANAGEMENT

3.1.1TIMELINE:

Week	Task	Description
Week 1	Project Planning	Define project goals, requirements, and scope for both admin and student apps. Create project timeline and allocate resources. Establish communication channels and protocols with team members/stakeholders.
Week 2-3	Admin App Development	Develop login authentication functionality for admin app. Create material design card views for notice upload, gallery image upload, PDF upload, faculty upload, delete notice, and logout. Implement upload functionality for notice, gallery image, PDF, and faculty information.
Week 3	Admin App Testing	Conduct thorough testing of all admin app functionalities. Address any issues and bugs identified.
Week 4-5	Student App Development	Develop login authentication functionality for student app. Create multiple fragments for home, notice, faculty, gallery, and about. Implement retrieval of admin data for notice, faculty, and gallery.
Week 6	Student App Testing	Conduct thorough testing of all student app functionalities. Address any issues and bugs identified.
Week 7	Integration and Refinement	Integrate admin and student apps. Refine user interface and user experience. Conduct additional testing and address any issues or bugs.
Week 8	Deployment	Deploy the apps to respective app stores. Conduct final testing and quality assurance. Release the apps to the public.
Week 9	Evaluation and Feedback	Monitor app usage and gather user feedback. Continuously evaluate the apps' performance and functionality. Make necessary updates and improvements based on feedback and user behavior.

3.1.2 TEAM COLLABORATION:

Team collaboration is crucial for the success of any project, including the development of admin and student apps. Here are some ways in which the project team can collaborate effectively:

- Establish clear communication channels and protocols among team members
- Define roles and responsibilities for each team member
- Hold regular meetings to discuss progress, roadblocks, and next steps
- Use project management tools to track tasks, timelines, and milestones
- Foster a culture of collaboration and encourage open communication

3.1.3 BUDGET MANAGEMENT:

For the admin app and student app project, the budget was managed with great attention to detail, ensuring that all expenses were tracked and accounted for. The team prioritized the use of cost-effective solutions, such as open source tools and libraries, to minimize expenses while still meeting project requirements. Resources were allocated efficiently and tasks were completed within the set timeframe to ensure that the project remained within budget.

3.1.4 RISK MANAGEMENT:

The team identified potential risks and uncertainties at the beginning of the project anddeveloped contingency plans to address them. This includes:

- Identified potential risks that could impact the project timeline, budget, and quality
- Conducted a risk analysis to assess the likelihood and impact of each identified risk
- Developed risk management strategies to mitigate or eliminate identified risks
- Implemented risk management strategies throughout the project lifecycle
- Regularly monitored and evaluated the effectiveness of risk management strategies
- Communicated identified risks and risk management strategies to team members and stakeholders to ensure everyone was aware and prepared for potential risks

3.1.5 CHALLENGES:

- Integrating the login authentication functionality for both admin and student apps was a challenge due to the different requirements and databases.
- Designing and implementing the user interface and user experience for both admin and student apps was a challenge to ensure consistency and ease of use.
- Ensuring the security of user data and preventing unauthorized access was a challenge for both admin and student apps.
- Testing and debugging the apps on multiple devices and platforms to ensure compatibility and performance was a challenge.

The team faced several challenges during the development of the admin and student app project, but they collaborated effectively to overcome them. Through careful planning and efficient resource allocation, they were able to deliver two fully functional apps that met the requirements and were user-friendly within the given timeline.

3.1.6 LESSONS LEARNED:

- Testing is a critical part of app development and should be done regularly to ensure that the app is working as expected and that all functionalities are being implemented correctly.
- It is important to use appropriate design patterns and architecture to ensure that the app is scalable and maintainable in the long run.
- Clear communication channels and protocols should be established among team members and stakeholders to avoid confusion and delays.
- User feedback is valuable and should be taken into account when making updates and improvements to the app.
- Utilizing open source tools and libraries can help reduce costs and speed up development.
- The security of user data should be a top priority, and appropriate measures should be taken to ensure that user information is protected.
- Proper documentation of the app's code and functionalities is crucial for future maintenance and updates.

3.2 PROJECT FLOW CHART & FUNCTIONALITIES:

The project consists of two separate apps - the Admin app and the Student app. The Admin app is used by college staff to manage and upload information, while the Student app is used by students to access this information.

The Admin app has login authentication requirements, and administrators can access the app using their email and password. Once logged in, they can upload notices, images, and PDF files, and can also upload faculty information with update and delete functionality. The app also includes a feature to delete notices, and a logout functionality to exit the app.

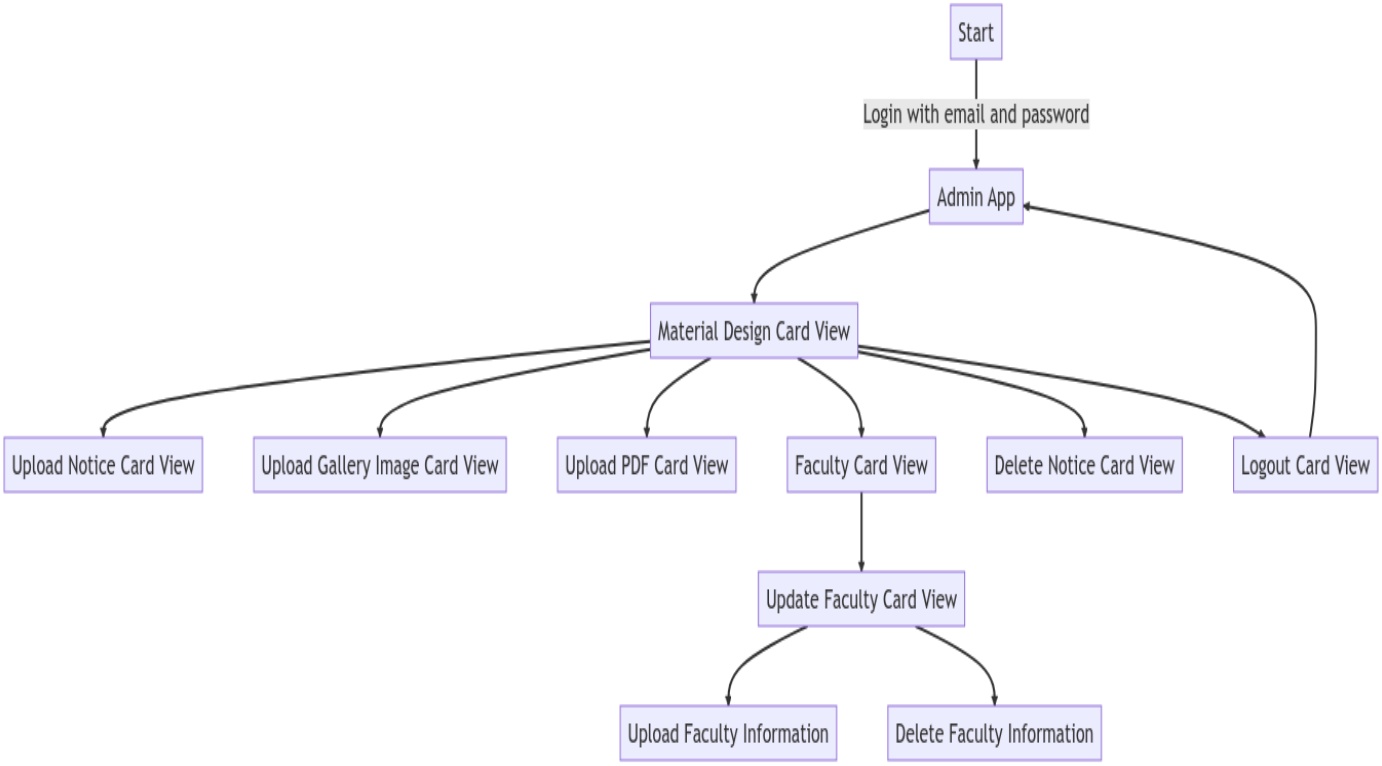
The Student app requires registration with basic information such as name, email, and password, which is stored in Firebase. Students can then log in using their email and password, and can also reset their password if needed.

The Student app has multiple fragments accessible through a navigation menu. The Home fragment shows a slider bar of college images, information about college departments, contact details, and the college's location on a map. The Notice fragment shows the latest notices uploaded by the Admin app, with the newest on top. The Faculty fragment displays uploaded faculty information such as name, post, email, and profile photo. The Gallery fragment displays uploaded images from the Admin app, divided into sections such as sport activities, cultural events, and other events. The About fragment shows information about the college, available branches, and streams such as civil, computer science, and mechanical engineering.

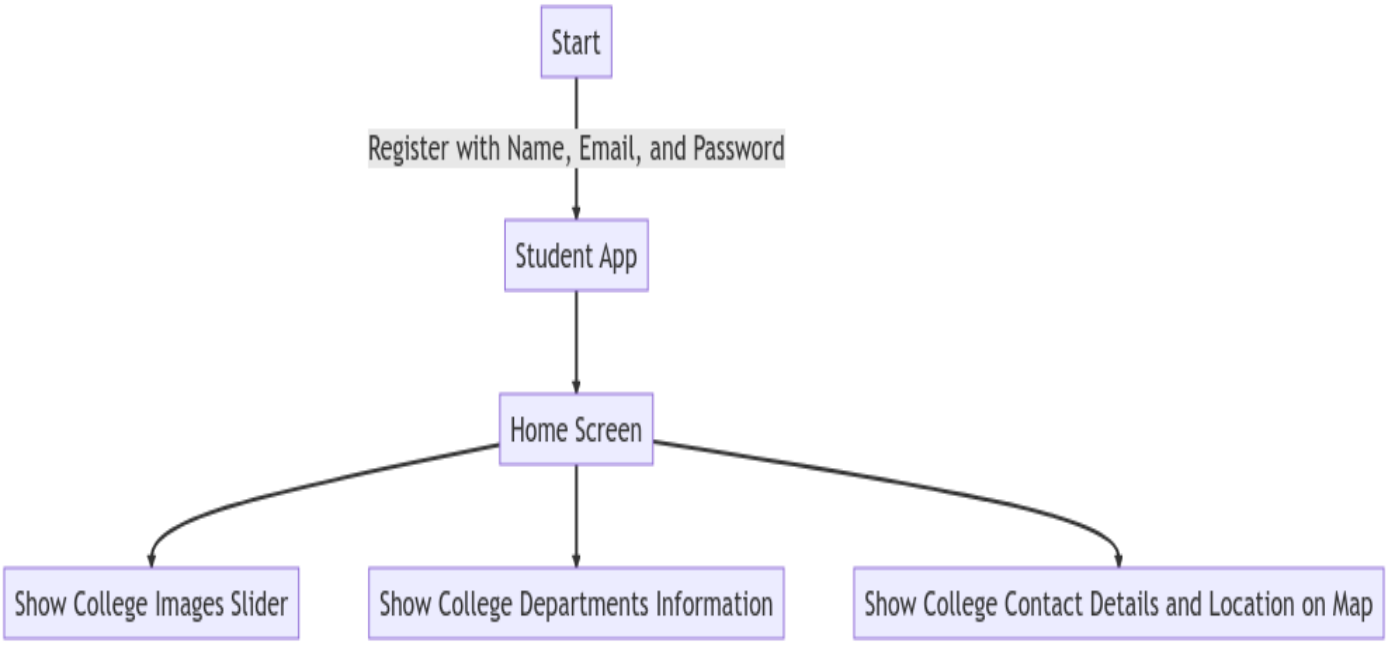
The Student app's navigation drawer also includes a video lectures fragment, an ebook fragment for PDF uploads from the Admin app, a website fragment to access the college's website, a share and rate us option, a developer option, and a logout functionality accessible through the options menu.

The project flow chart involves the Admin app being used to upload information, and this information being accessed by students through the Student app. The Admin app and Student app are separate entities, with the Admin app being responsible for managing information and the Student app for displaying it to students.

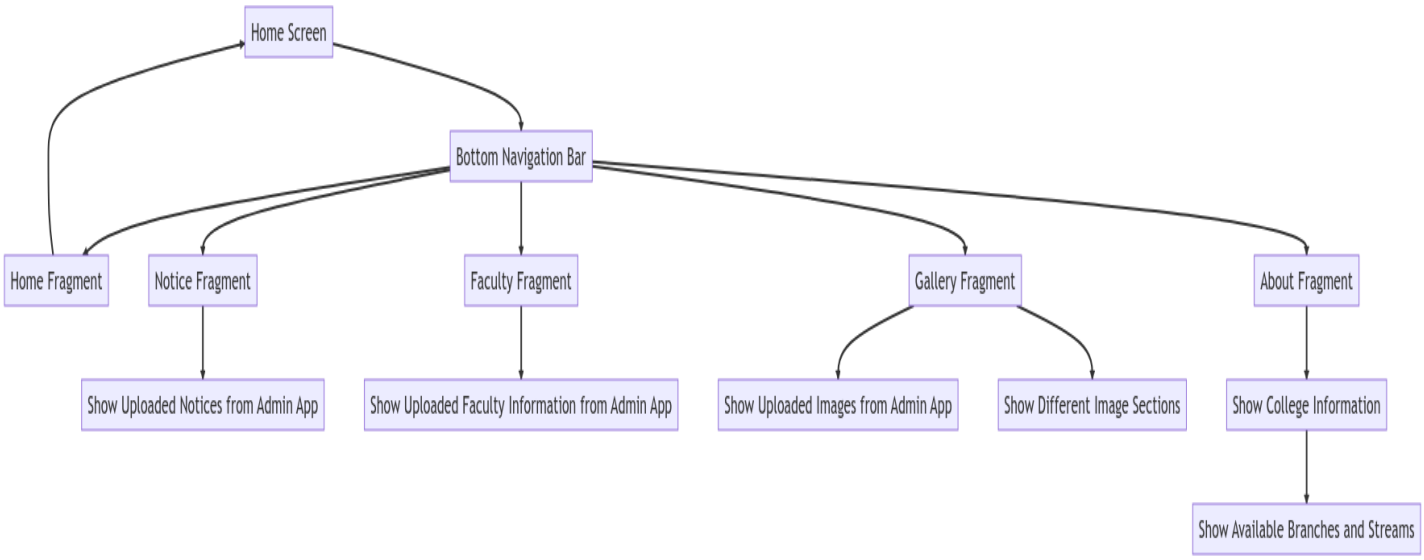
3.2.1 Flow Chart [ADMIN APP]:



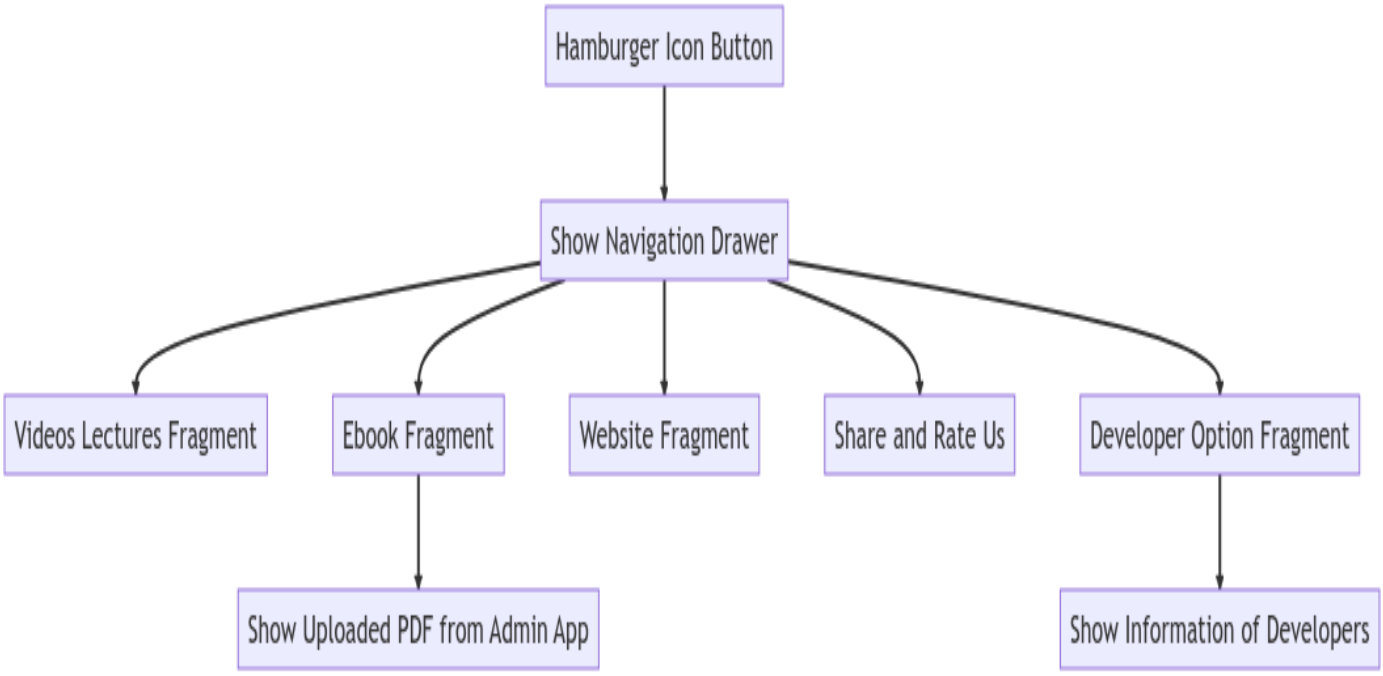
3.2.2 Flow Chart [STUDENT APP]:



3.2.3 Flow Chart [HOME SCREEN]:



3.2.4 Flow Chart [NAVIGATION DRAWER]:



3.2.5 FUNCTIONALITIES [CATEGORY & FEATURES]:

Admin App:

CATEGORY	FEATURES
Authentication	Email and password login for admin with admin@gmail.com and password 12345
Notice	Upload notice with title, delete notice
Gallery	Upload gallery images with title
PDF	Upload PDFs with title
Faculty	Update and delete faculty information
Logout	Logout functionality

COLLEGE APP

Student App:

CATEGORY	FEATURES
Authentication	Email and password login for students, password recovery
Home	Display college images slider, college departments, college contact details, and college location on a map
Notice	Show latest uploaded notices on top
Faculty	Show uploaded faculty information including name, post, email, and profile photo
Gallery	Show uploaded images in different sections like sport activities, cultural events, etc.
About	Display college information including available branches and streams like civil, CS, and mechanical, etc.
Navigation Drawer	Videos lectures fragment, ebook fragment to show uploaded PDFs, website fragment to show college website, share and rate us, developer option, etc.
Logout	Logout functionality accessible through option menu on the right corner of the student app

3.2.6 FUNCTIONALITIES [IN DETAIL]:

Admin App:

- Authentication: The admin app requires email and password authentication for login. The admin can log in using the credentials admin@gmail.com and password 12345.
- Notice Upload: The admin can upload notices with a title. This functionality allows the admin to share important information with the students.
- Gallery Upload: The admin can upload images to the gallery section with a title. This functionality enables the admin to share images related to various college events and activities.
- PDF Upload: The admin can upload PDFs with a title. This functionality allows the admin to share important study material with the students.
- Faculty Management: The admin can manage the faculty information by updating or deleting it. This functionality enables the admin to keep the faculty information up-to-date.
- Notice Deletion: The admin can delete any notice that is no longer relevant or required.
- Logout: The admin can log out of the app using the logout functionality.

Student App:

- Authentication: The student app requires email and password authentication for login. Students can register and fill in their information like name, email, and password, which will be stored in Firebase. They can also recover their password if they forget it.
- Home: The home fragment of the student app displays the college images slider, college departments' information, college contact details, and college location on a map.

COLLEGE APP

- Notice Display: The notice fragment shows the latest uploaded notices on top. This functionality enables the students to stay updated with important information related to the college.
- Faculty Display: The faculty fragment shows uploaded faculty information from the admin app, including their name, post, email, and profile photo. This functionality enables the students to know about the faculty members of the college.
- Gallery Display: The gallery fragment shows uploaded images from the admin app in different sections like sports activities, cultural events, and other events. This functionality allows the students to view images related to various college events and activities.
- About Display: The about fragment shows college information, including available branches and streams like civil, CS, and mechanical, etc. This functionality enables the students to know more about the college.
- Navigation Drawer: The navigation drawer of the student app includes multiple functionalities like videos lectures fragment, ebook fragment to show uploaded PDFs, website fragment to show the college website, share and rate us, developer option to show developer information like name, image, email, etc.
- Logout: The student app provides logout functionality in the option menu on the right corner. When clicked, the student will go to the student login page.

CHAPTER 4

IMPLEMENTATION

PROJECT DEVELOPMENT

3.3.1 SOFTWARE TOOLS & LANGUAGES:

XML:

XML (Extensible Markup Language) is used in Android app development as a way to describe and define the layout and structure of the user interface (UI) components. Here are some key points to keep in mind about XML in Android app development:

- XML files are used to define the UI layout and structure of an Android app.
- XML files can be edited in Android Studio's XML editor, which provides syntax highlighting, code completion, and validation tools.
- The XML files in Android apps follow a hierarchical structure, with the top-level element being the root element that contains other elements nested inside it.
- The UI components defined in XML files are called views, and they can be of different types, such as TextView, Button, ImageView, EditText, and more.
- Views can be arranged in different layouts, such as LinearLayout, RelativeLayout, GridLayout, and more.
- Views can be customized using attributes, such as text, color, font, size, padding, margin, and more.
- Views can be given unique IDs, which are used to reference them in Java code, allowing them to be manipulated programmatically.

Java:

Java is the primary programming language used in Android app development. Here are some key points to keep in mind about Java in Android app development:

- Java is an object-oriented programming language, which means it is organized around objects, data, and methods that manipulate that data.
- Java code is compiled into bytecode that can run on any platform that has a Java Virtual Machine (JVM).
- Android uses the Java Development Kit (JDK) to compile and run Java code on mobile devices.
- Java is used to implement the business logic of an Android app, such as handling user input, processing data, and communicating with other systems.

COLLEGE APP

- Java code in Android apps is organized into classes, which define the data and methods used in the app.
- Android apps typically have a main activity, which is the entry point of the app and handles the app's UI interactions.
- Java code in Android apps can interact with the UI components defined in XML files, allowing the app to respond to user input and update the UI.

GIT:

- Git is a distributed version control system that allows for easy collaboration and trackingof code changes.
- Git allows for easy branching and merging of code, making it easier to work on differentfeatures or bug fixes at the same time.
- The use of Git can help prevent conflicts and ensure code consistency across team members.
- Git provides a range of tools for code review and collaboration, such as pull requests andissues.
- Git allows for easy reverting to previous versions of the code if necessary.
- Git is widely used and supported, with a large community and
- documentation.Git can be integrated with different code editors and development tools.

GITHUB:

- GitHub is a web-based platform for hosting and sharing code repositories.
- GitHub provides tools for collaboration, such as pull requests, code reviews, and issuetracking.
- GitHub allows for easy integration with different development tools and services.
- GitHub provides built-in security features to protect code repositories and prevent unauthorized access.
- GitHub has a large community of developers and provides opportunities for networkingand contributing to open-source projects.
- GitHub provides hosting for documentation and other project resources.
- GitHub can be used for project management, with features such as milestones and projectboards.

COLLEGE APP

ANDROID STUDIO:

- Android Studio is the official Integrated Development Environment (IDE) for Android app development.
- It is based on the IntelliJ IDEA Community Edition and tailored specifically for Android development.
- Android Studio provides a rich set of tools for building and debugging Android apps, including an advanced code editor, a visual layout editor, a powerful debugger, and more.
- The IDE is available for Windows, macOS, and Linux operating systems.
- Android Studio supports various programming languages, including Java, Kotlin, and C++.
- Android Studio also includes a powerful Gradle-based build system that automates the process of building, testing, and deploying Android apps.
- The IDE provides integration with various Android SDK tools, including Android Emulator, Android Debug Bridge (ADB), and more.
- Android Studio provides support for version control systems such as Git, enabling developers to manage their source code effectively.

ANDROID STUDIO:

- Firebase is a cloud-based mobile and web application development platform that provides developers with various tools and services to build, improve and grow their apps.
- Firebase was acquired by Google in 2014 and has since become an integral part of the Android development ecosystem.
- Firebase offers a range of features for building high-quality, scalable and secure mobile applications, including authentication, real-time databases, storage, cloud messaging, and analytics.
- Firebase Authentication provides easy-to-use authentication services for app users, including email and password, Google Sign-In, Facebook Login, and more.
- Firebase Realtime Database is a NoSQL cloud-hosted database that enables developers to store and synchronize data in real-time across multiple clients and platforms.
- Firebase Storage is a cloud-based storage solution that allows developers to store and retrieve user-generated content, such as images, videos, and audio files.
- Firebase Cloud Messaging (FCM) is a cross-platform messaging solution that enables app developers to send push notifications and messages to their app users across multiple platforms, including Android, iOS, and the web.
- Firebase Performance Monitoring is a tool that helps developers identify and fix performance issues in their apps by providing real-time data on app performance and user engagement.
- Firebase Test Lab is a testing solution that enables developers to test their apps on real devices in the cloud, allowing them to identify issues early and ensure their apps work seamlessly

across multiple devices and platforms.

3.3.2 PROJECT FOLDER STRUCTURE:

```
com.example.studentapp
├── authentication
│   ├── ForgetPasswordActivity.java
│   ├── LoginActivity.java
│   └── RegisterActivity.java
├── ebook
│   ├── Adapter.java
│   ├── FileinModel.java
│   └── RetrivePDFActivity.java
└── ui
    ├── about
    │   ├── AboutFragment.java
    │   ├── BranchAdapter.java
    │   └── BranchModel.java
    ├── faculty
    │   ├── FacultyFragment.java
    │   ├── TeacherAdapter.java
    │   └── TeacherData.java
    ├── gallery
    │   ├── GalleryAdapter.java
    │   └── GalleryFragment.java
    ├── home
    │   └── HomeFragment.java
    ├── notice
    │   ├── NoticeAdapter.java
    │   ├── NoticeData.java
    │   └── NoticeFragment.java
    ├── MainActivity.java
    ├── Developers.java
    └── FullImageView.java
```

COLLEGE APP

- app
  - src
    - main
      - java
        - com.example.admincollegeapp
          - faculty
            - AddTeacher.java
            - TeacherAdapter.java
            - TeacherData.java
            - UpdateFaculty.java
            - UpdateTeacherActivity.java
          - notice
            - DeleteNoticeActivity.java
            - NoticeAdapter.java
            - NoticeData.java
            - UploadNotice.java
          - LoginActivity.java
          - MainActivity.java
          - UploadImage.java
          - UploadPdfActivity.java
      - res
        - layout
          - activity_add_teacher.xml
          - activity_delete_notice.xml
          - activity_login.xml
          - activity_main.xml
          - activity_update_faculty.xml
          - activity_update_teacher.xml
          - activity_upload_image.xml
          - activity_upload_pdf.xml
          - row_teacher.xml
          - row_notice.xml
        - drawable
          - image1.png
          - image2.png
          - ...
        - values
          - colors.xml
          - strings.xml
          - styles.xml

3.3.4 PROJECT CODE:

Admin App

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.admincollegeapp">
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@drawable/logo"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.AdminCollegeApp"
        tools:targetApi="31">
        <activity
            android:name=".LoginActivity"
            android:exported="true"
            android:theme="@style/Theme.MaterialComponents.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".faculty.UpdateTeacherActivity"
            android:exported="false" />
        <activity
            android:name=".notice.DeleteNoticeActivity"
            android:exported="false" />
        <activity
            android:name=".faculty.AddTeacher"
            android:exported="false" />
        <activity
            android:name=".faculty.UpdateFaculty"
            android:exported="false" />
        <activity
            android:name=".UploadPdfActivity"
            android:exported="false" />
        <activity
            android:name=".UploadImage"
            android:exported="false" />
        <activity
            android:name=".notice.UploadNotice"
            android:exported="false" />
        <activity
            android:name=".MainActivity"
            android:exported="false">
        </activity>
    </application>

</manifest>
```

Admin App

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:gravity="center"
        android:orientation="horizontal">

        <!--        add Notice Card View -->
        <com.google.android.material.card.MaterialCardView
            android:id="@+id/addNotice"
            android:layout_width="130dp"
            android:layout_height="150dp"
            android:layout_margin="10dp"
            app:cardElevation="5dp">

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

                <ImageView
                    android:layout_width="64dp"
                    android:layout_height="64dp"
                    android:background="@drawable/circle_green"
                    android:padding="15dp"
                    android:src="@drawable/ic_notice" />

                <View
                    android:layout_width="match_parent"
                    android:layout_height="1dp"
                    android:layout_marginTop="10dp"
                    android:background="@color/lightGray" />

                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="10dp"
                    android:padding="5dp"
                    android:text="Upload Notice"
                    android:textColor="@color/textColor"
                    android:textStyle="bold" />

            </LinearLayout>

        </com.google.android.material.card.MaterialCardView>
```

```
<!--         addGalleryImage Card View -->
        <com.google.android.material.card.MaterialCardView
            android:id="@+id/addGalleryImage"
            android:layout_width="130dp"
            android:layout_height="150dp"
            android:layout_margin="10dp"
            app:cardElevation="5dp">

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

                <ImageView
                    android:layout_width="64dp"
                    android:layout_height="64dp"
                    android:background="@drawable/circle_purple"
                    android:padding="15dp"
                    android:src="@drawable/ic_gallery" />

                <View
                    android:layout_width="match_parent"
                    android:layout_height="1dp"
                    android:layout_marginTop="10dp"
                    android:background="@color/lightGray" />

                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="10dp"
                    android:padding="5dp"
                    android:text="Upload Image"
                    android:textColor="@color/textColor"
                    android:textStyle="bold" />

            </LinearLayout>

        </com.google.android.material.card.MaterialCardView>


    </LinearLayout>
```

COLLEGE APP

Admin App

MainActivity.java

```
package com.example.admincollegeapp;

import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;

import com.example.admincollegeapp.faculty.UpdateFaculty;
import com.example.admincollegeapp.notice.DeleteNoticeActivity;
import com.example.admincollegeapp.notice.UploadNotice;


public class MainActivity extends AppCompatActivity implements
View.OnClickListener {

    CardView uploadNotice, addGalleryImage, addEbook, faculty, deleteNotice,
logout;

    private SharedPreferences sharedPreferences;
    private SharedPreferences.Editor editor;

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

        sharedPreferences = this.getSharedPreferences("login", MODE_PRIVATE);
        editor = sharedPreferences.edit();

        if (sharedPreferences.getString("isLogin", "false").equals("no")) {
            openLogin();
        }


        uploadNotice = findViewById(R.id.addNotice);
        addGalleryImage = findViewById(R.id.addGalleryImage);
        addEbook = findViewById(R.id.addEbook);
        faculty = findViewById(R.id.faculty);
        deleteNotice = findViewById(R.id.deleteNotice);
        logout = findViewById(R.id.logout);


        uploadNotice.setOnClickListener(this);
        addGalleryImage.setOnClickListener(this);
        addEbook.setOnClickListener(this);
        faculty.setOnClickListener(this);
        deleteNotice.setOnClickListener(this);
        logout.setOnClickListener(this);
    }
```

```
private void openLogin() {
        startActivity(new Intent(MainActivity.this, LoginActivity.class));
        finish();
    }


    @SuppressLint("NonConstantResourceId")
    @Override
    public void onClick(View view) {
        Intent intent;
        switch (view.getId()) {
            case R.id.addNotice:
                intent = new Intent(MainActivity.this, UploadNotice.class);
                startActivity(intent);
                break;
            case R.id.addGalleryImage:
                intent = new Intent(MainActivity.this, UploadImage.class);
                startActivity(intent);
                break;
            case R.id.addEbook:
                intent = new Intent(MainActivity.this, UploadPdfActivity.class);
                startActivity(intent);
                break;

            case R.id.faculty:
                intent = new Intent(MainActivity.this, UpdateFaculty.class);
                startActivity(intent);
                break;

            case R.id.deleteNotice:
                intent = new Intent(MainActivity.this,
DeleteNoticeActivity.class);
                startActivity(intent);
                break;

            case R.id.logout:
                editor.putString("isLogin", "false");
                editor.commit();
                openLogin();
                break;


        }
    }
}
```

Student App

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE"
        tools:ignore="ScopedStroage" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@drawable/clguserapplogo"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.StudentApp"
        tools:targetApi="31">
        <activity
            android:name=".authentication.ForgetPasswordActivity"
            android:exported="false" />
        <activity
            android:name=".authentication.LoginActivity"
            android:exported="false" />
        <activity
            android:name=".authentication.RegisterActivity"
            android:exported="false" />
        <activity
            android:name=".ebook.RetrievePDFActivity"
            android:exported="false" />
        <activity
            android:name=".Developers"
            android:exported="false" />
        <activity
            android:name=".FullImageView"
            android:exported="false" />
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

COLLEGE APP

Student App

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>

<androidx.drawerlayout.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawerLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <androidx.constraintlayout.widget.ConstraintLayout
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=".MainActivity">

        <fragment
            android:id="@+id/frame_layout"
            android:name="androidx.navigation.fragment.NavHostFragment"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            app:defaultNavHost="true"
            app:layout_constraintBottom_toTopOf="@+id/bottomNavigationView"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:navGraph="@navigation/mobile_navigation" />


        <com.google.android.material.bottomnavigation.BottomNavigationView
            android:id="@+id/bottomNavigationView"
            android:layout_width="match_parent"
            android:layout_height="50dp"

            android:background="?android:attr/windowBackground"
            app:labelVisibilityMode="selected"
            app:layout_constraintBottom_toBottomOf="parent"

            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:menu="@menu/bottom_nav" />

    </androidx.constraintlayout.widget.ConstraintLayout>

    <com.google.android.material.navigation.NavigationView
        android:id="@+id/navigation_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/drawer_header"
        app:itemTextAppearance="?android:attr/textAppearanceMedium"
        app:menu="@menu/navigation_drawer" />

</androidx.drawerlayout.widget.DrawerLayout>
```

COLLEGE APP

Student App

MainActivity.java

```
package com.example.studentapp;

import android.annotation.SuppressLint;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBarDrawerToggle;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.app.AppCompatDelegate;
import androidx.core.view.GravityCompat;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.ui.NavigationUI;

import com.example.studentapp.authentication.LoginActivity;
import com.example.studentapp.ebook.RetrievePDFActivity;
import com.google.android.material.bottomnavigation.BottomNavigationView;
import com.google.android.material.dialog.MaterialAlertDialogBuilder;
import com.google.android.material.navigation.NavigationView;
import com.google.firebase.auth.FirebaseAuth;

import java.util.Objects;


public class MainActivity extends AppCompatActivity implements
NavigationView.OnNavigationItemSelectedListener {

    private BottomNavigationView bottomNavigationView;
    private NavController navController;

    private DrawerLayout drawerLayout;
    private ActionBarDrawerToggle toggle;
    private NavigationView navigationView;

    // for theme
    private SharedPreferences sharedPreferences;
    private SharedPreferences.Editor editor;
    private int checkedItem;
    private String selected;

    private final String CHECKEDITEM = "checked_item";

    private FirebaseAuth auth;
    @SuppressLint("MissingInflatedId")
    @Override
```

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        auth = FirebaseAuth.getInstance();

        sharedPreferences = this.getSharedPreferences("themes",
Context.MODE_PRIVATE);
        editor = sharedPreferences.edit();

        switch (getCheckedItem()) {
            case 0:

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_FOLLOW_SYSTEM);
                break;
            case 1:

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_YES);
                break;
            case 2:

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
                break;
        }

        bottomNavigationView = findViewById(R.id.bottomNavigationView);
        navController = Navigation.findNavController(this, R.id.frame_layout);

        drawerLayout = findViewById(R.id.drawerLayout);
        navigationView = findViewById(R.id.navigation_view);

        toggle = new ActionBarDrawerToggle(this, drawerLayout, R.string.start,
R.string.close);

        drawerLayout.addDrawerListener(toggle);
        toggle.syncState();

//        menu showing using below line

Objects.requireNonNull(getSupportActionBar()).setDisplayHomeAsUpEnabled(true);

        navigationView.setNavigationItemSelectedListener(this);

        NavigationUI.setupWithNavController(bottomNavigationView, navController);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.option_menu, menu);
        return true;

    }

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        if (toggle.onOptionsItemSelected(item)) {
            return true;
```

```
if (item.getItemId() == R.id.logout) {
            auth.signOut();
            openLogin();
        }
        return true;
    }

    private void openLogin() {
        startActivity(new Intent(MainActivity.this, LoginActivity.class));
        finish();
    }

    @Override
    protected void onStart() {
        super.onStart();
        if (auth.getCurrentUser() == null) {
            openLogin();
        }
    }

    @SuppressLint("NonConstantResourceId")
    @Override
    public boolean onNavigationItemSelected(@NonNull MenuItem item) {
        switch (item.getItemId()) {
            case R.id.navigation_developer:
                try {
                    Intent i = new Intent(MainActivity.this, Developers.class);
                    startActivity(i);
                } catch (Exception e) {
                    Toast.makeText(this, "Unable to open\n" + e.getMessage(),
Toast.LENGTH_SHORT).show();
                }

                break;

            case R.id.navigation_video:
//                Toast.makeText(this, "Video Lectures",
Toast.LENGTH_SHORT).show();

                try {
                    Intent i = new Intent(Intent.ACTION_VIEW);

i.setData(Uri.parse("https://www.youtube.com/@clgjodhpursumerpur/featured"));
                    startActivity(i);
                } catch (Exception e) {
                    Toast.makeText(this, "Unable to open\n" + e.getMessage(),
Toast.LENGTH_SHORT).show();
                }
                break;

            case R.id.navigation_rate:
//                Toast.makeText(this, "Rate us", Toast.LENGTH_SHORT).show();
                try {
                    Uri uri =
Uri.parse("https://play.google.com/store/apps/details?id=" +
getApplicationContext().getPackageName());
                    Intent i = new Intent(Intent.ACTION_VIEW, uri);
                    startActivity(i);
                } catch (Exception e) {
                    Toast.makeText(this, "Unable to open\n" + e.getMessage(),
```

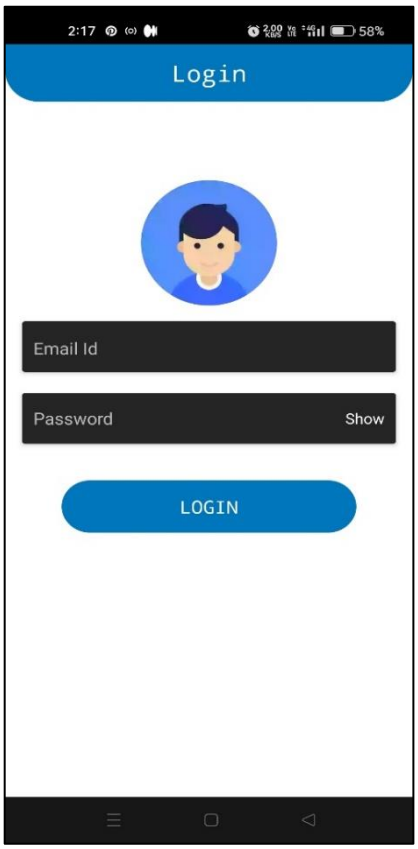
CHAPTER 5

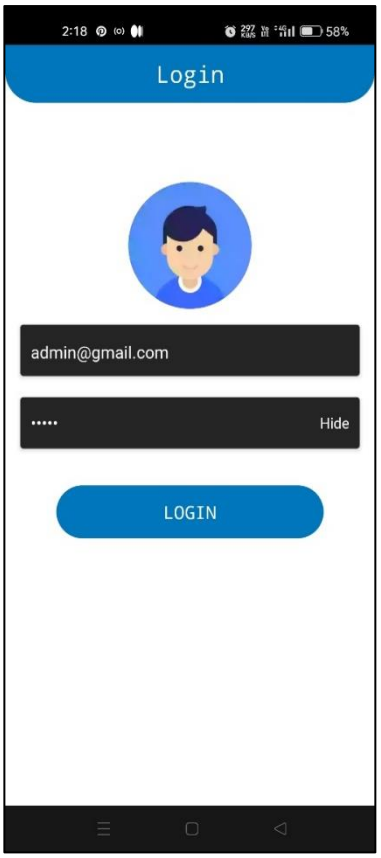
PROJECT SHOWCASE

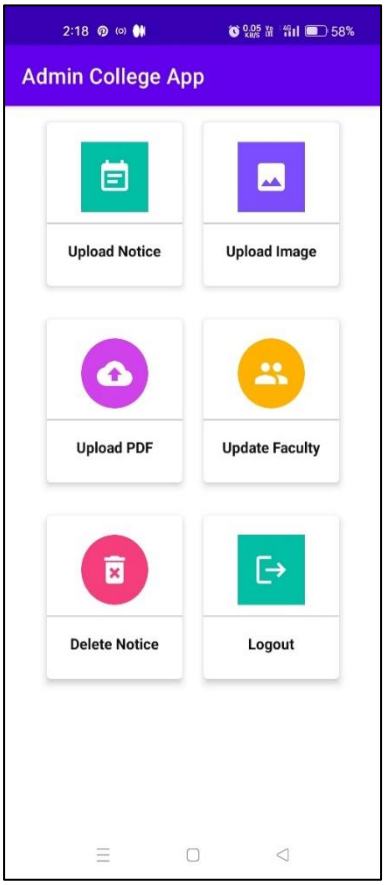
6.1 Admin App

II. Admin College App
A. Login authentication
B. Notice upload functionality
C. Gallery image upload functionality
D. PDF upload functionality
E. Faculty update and delete functionality
F. Notice delete functionality

A. Login Authentication The Admin App provides secure login authentication using email and password. The email is set to admin@gmail.com and the password is 12345. The login page validates the user's credentials before allowing access to the app.

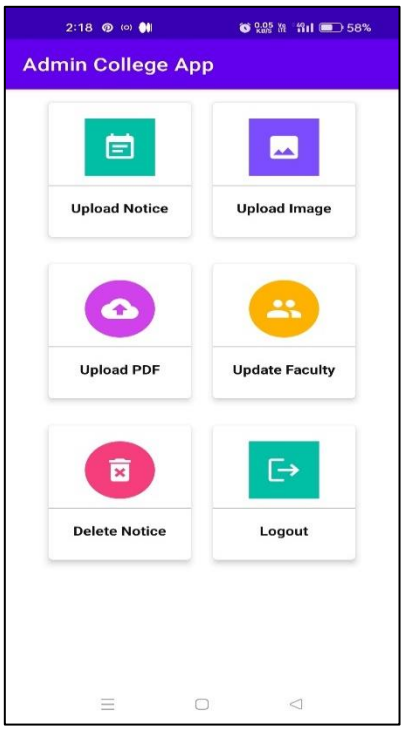


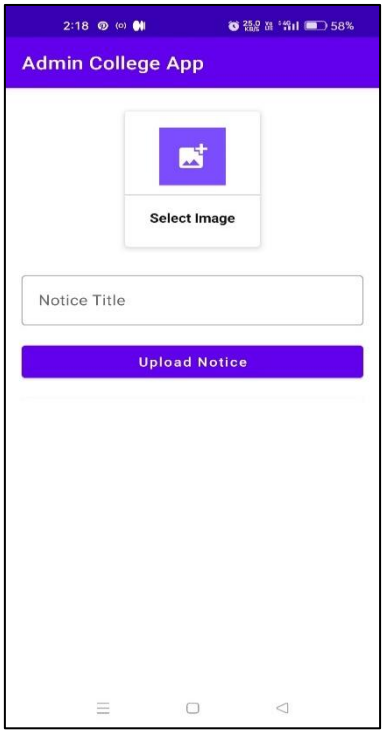


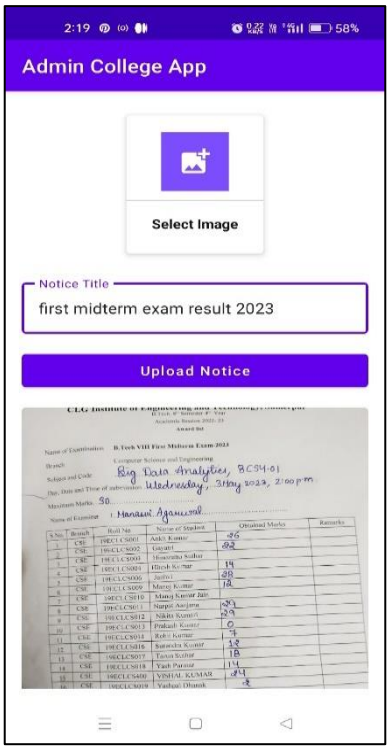


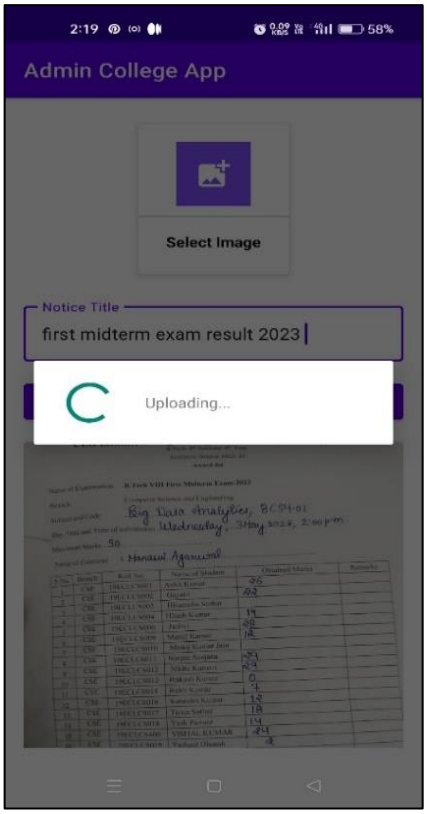
COLLEGE APP

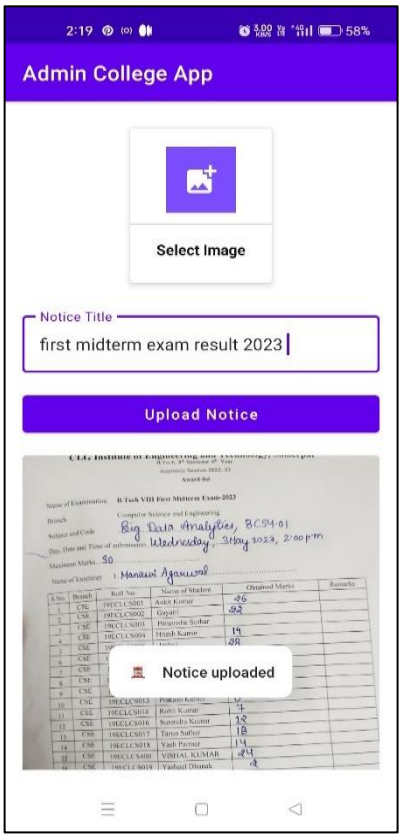
B. Notice Upload Functionality The Admin App allows the user to upload notices, which are then displayed on the Student App. The notice upload functionality is accessible from the app's home screen. Once the user selects the upload notice option, they can enter the title and description of the notice, select the target audience (i.e., all students, specific department students), and upload a notice image if desired.





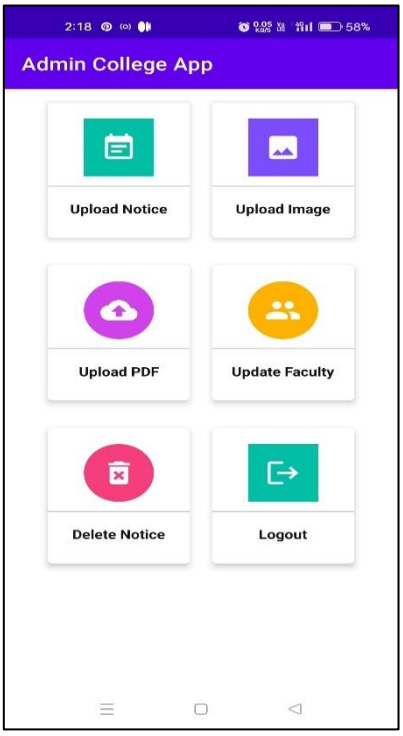


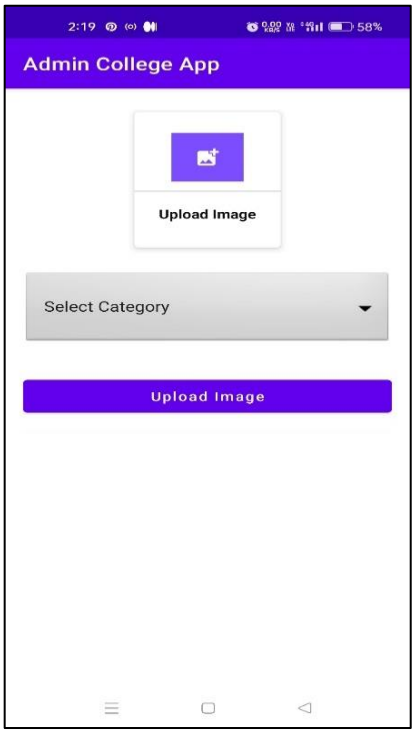


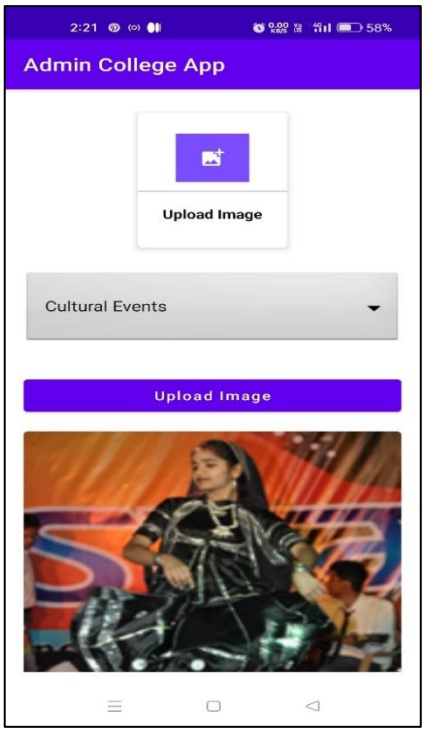


COLLEGE APP

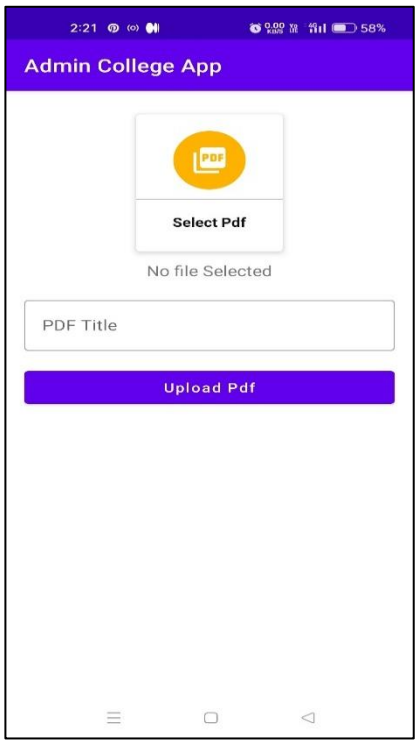
C. Gallery Image Upload Functionality The Admin App also allows the user to upload images to the app's gallery. The gallery image upload functionality is accessible from the app's home screen. The user can select the upload gallery option, choose the category for the image (e.g., sports, cultural events), and upload an image.

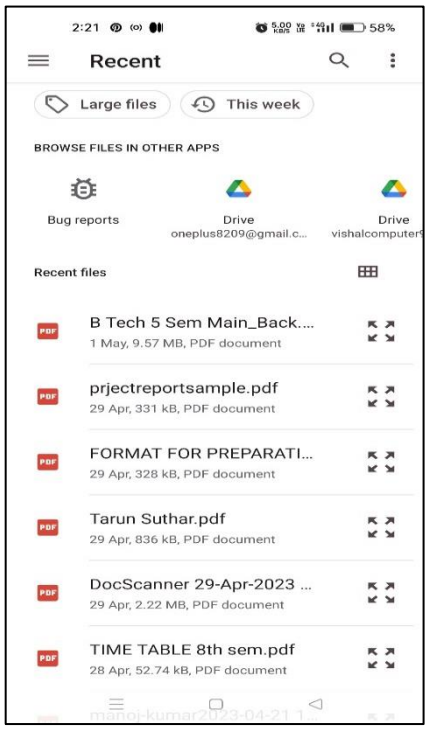


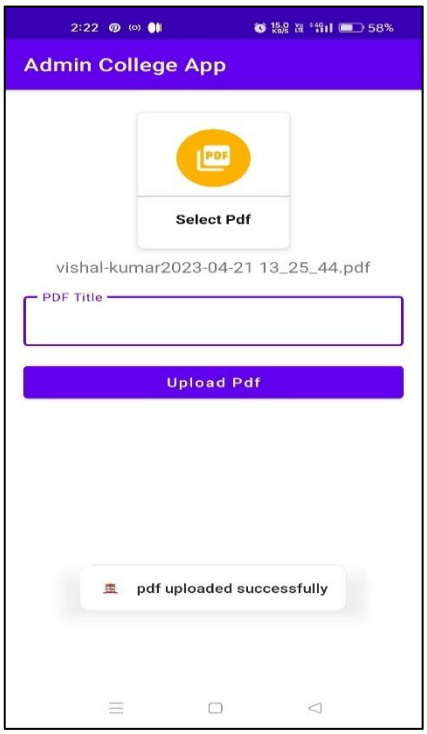




D. PDF Upload Functionality The Admin App includes a PDF upload functionality that allows the user to upload PDF files. This functionality is accessible from the app's home screen. The user can select the upload PDF option, enter the name and description of the PDF, and upload the PDF file.

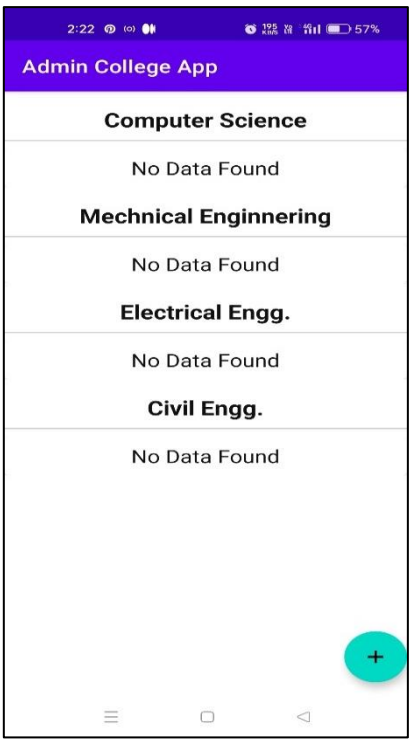


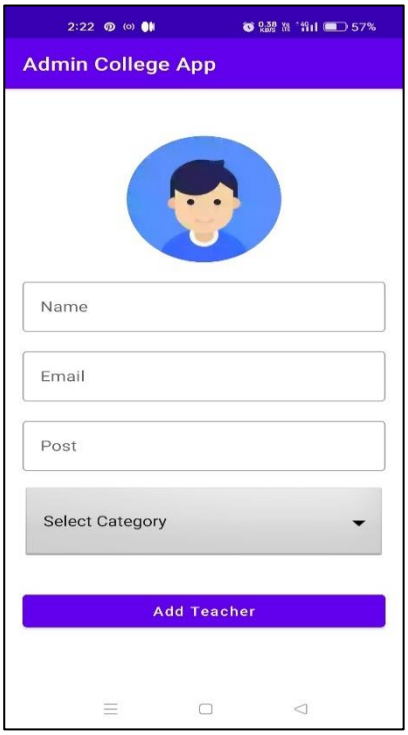


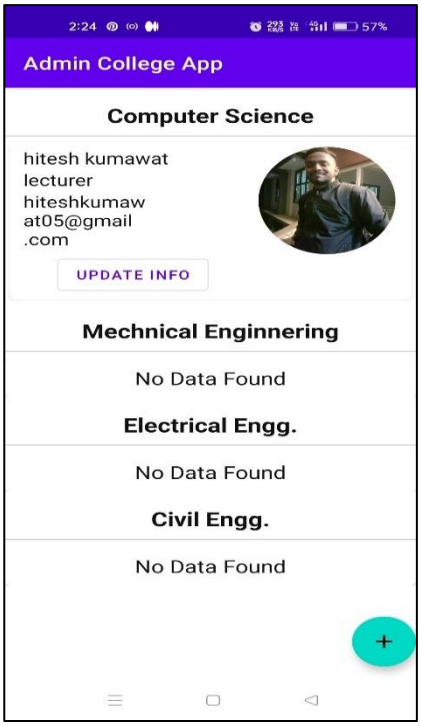


COLLEGE APP

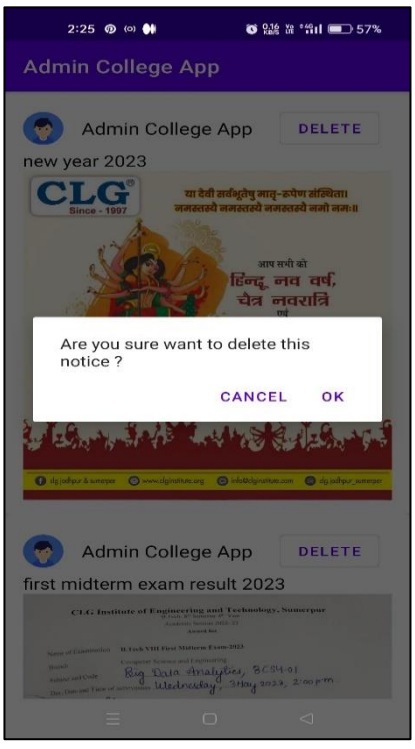
E. Faculty Update and Delete Functionality The Admin App provides functionality to update and delete faculty information. This functionality is accessible from the app's home screen. The user can select the faculty update option, select the faculty member to update or delete, and make the necessary changes.

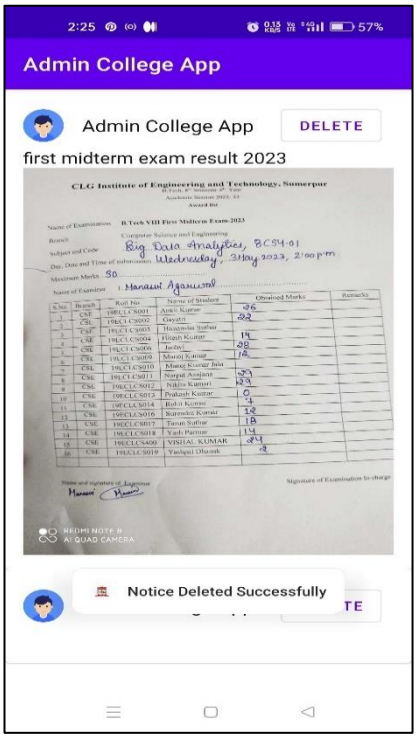






F. Notice Delete Functionality The Admin App also provides functionality to delete notices. This functionality is accessible from the app's home screen. The user can select the delete notice option, select the notice to delete, and confirm the deletion.

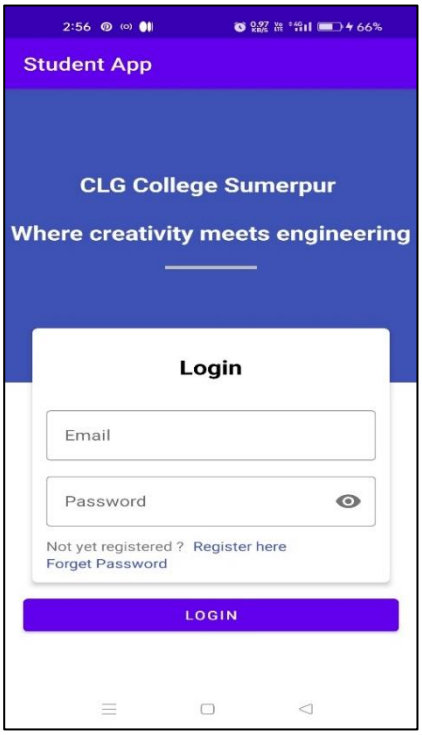




6.2 Student App

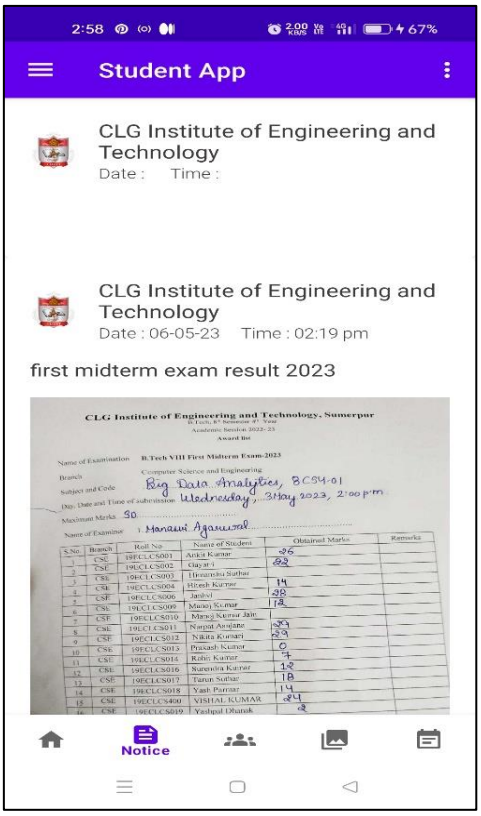
III. Student App

A.Registration and login authentication

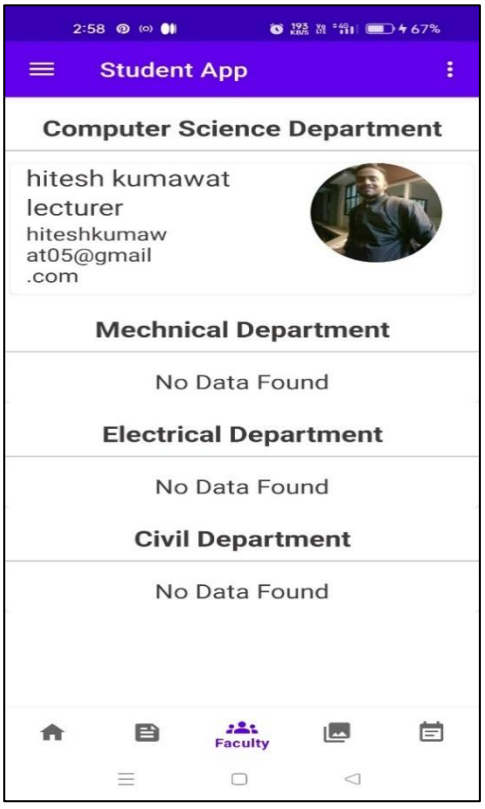


B. Home Screen

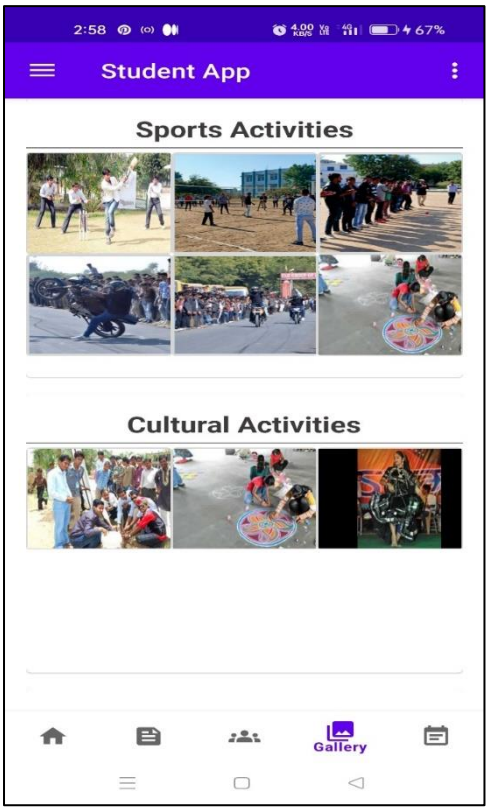
C. **Notice fragment (Display uploaded notices from Admin App)**



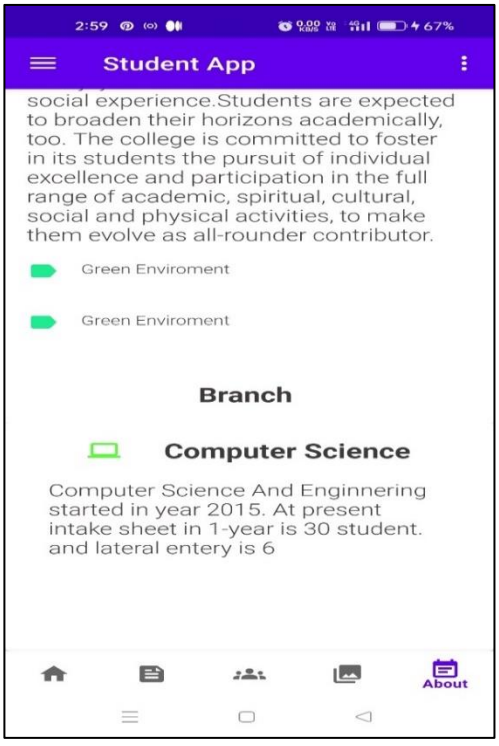
D. Faculty fragment(Display uploaded faculty information like Name, post, email, and profile photo from Admin App)



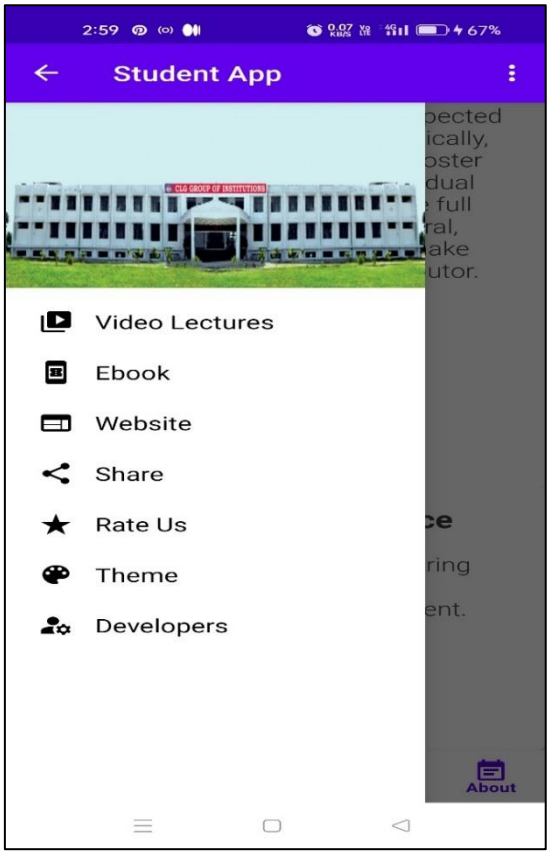
E. Gallery fragment (Display uploaded images in Different sections for sport activities, cultural events, etc from Admin App.



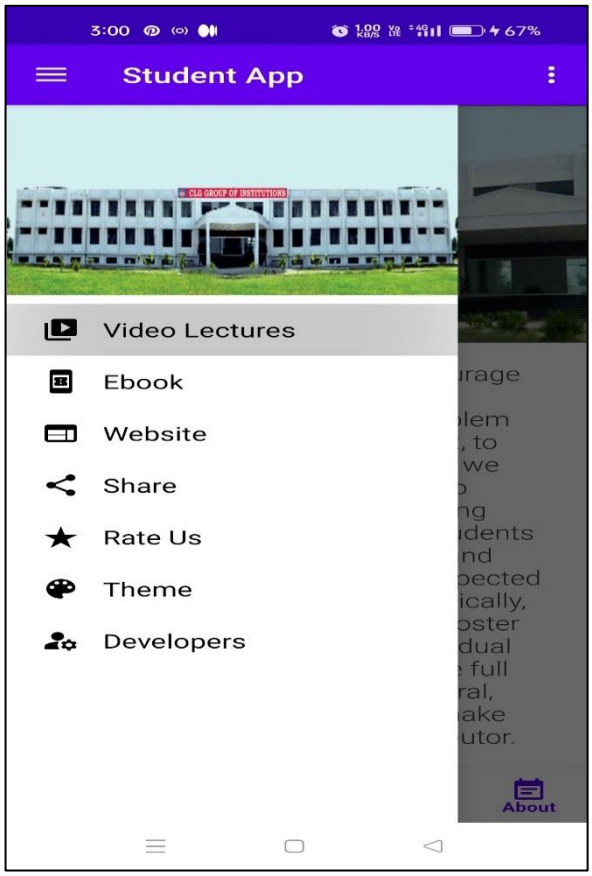
F. About fragment (Display college information and Available branches and streams like civil, CS, and mechanical)

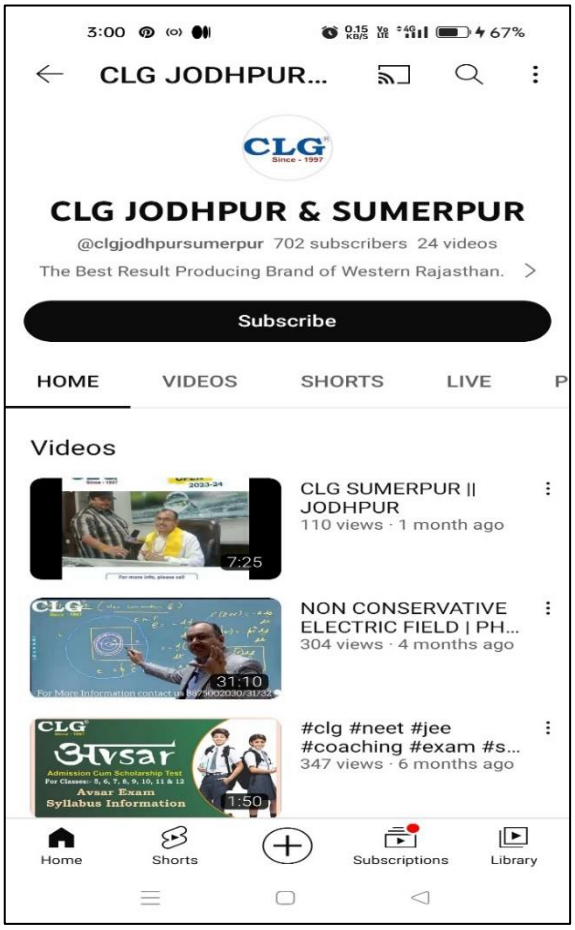


G. Navigation drawer



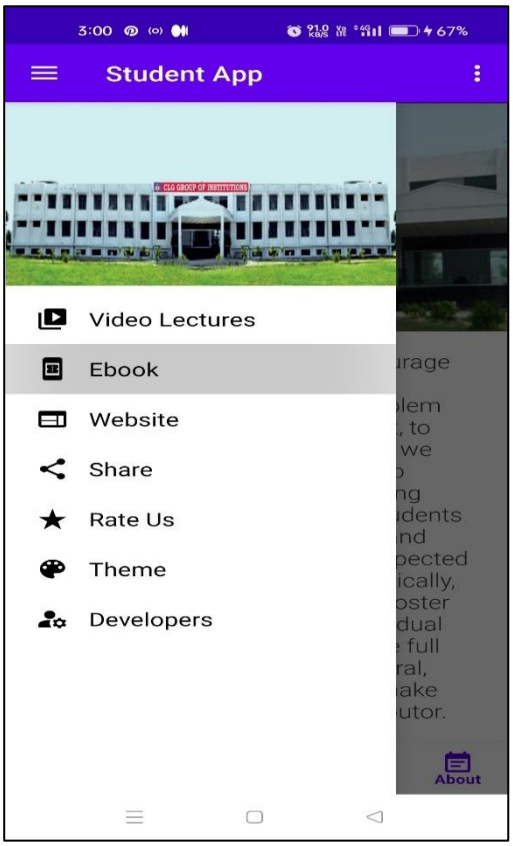
H. Videos lectures fragment

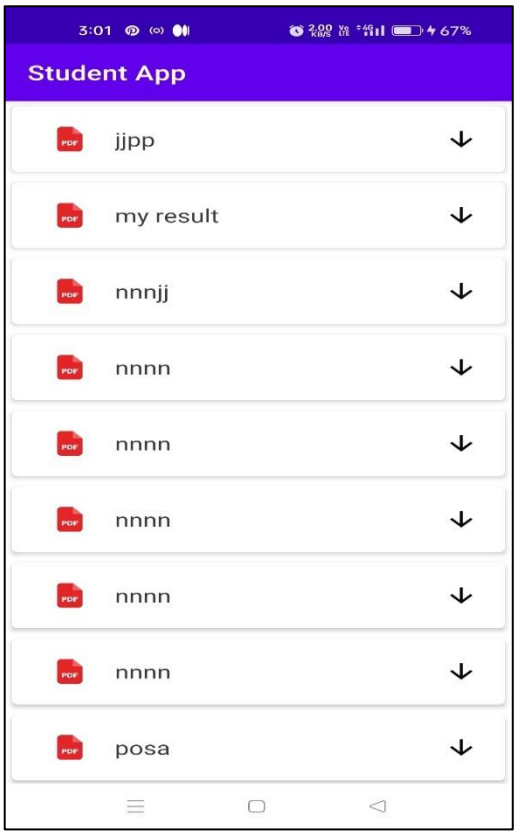




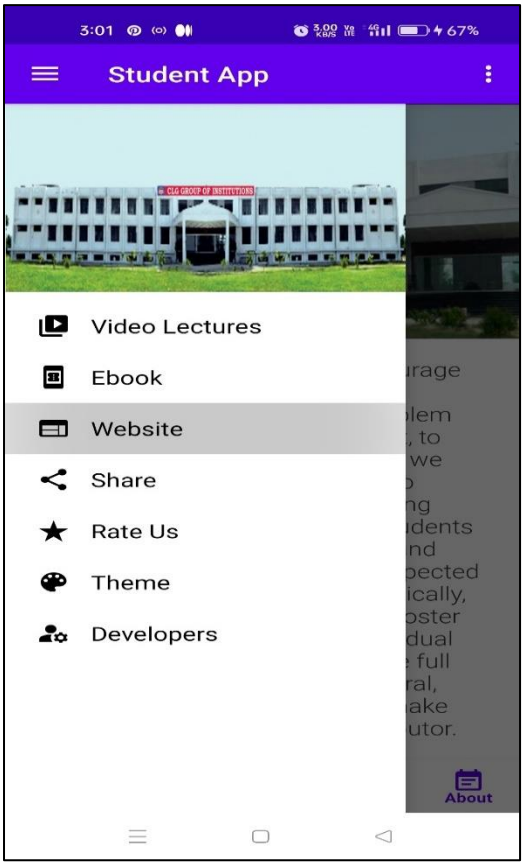
COLLEGE APP

I.Ebook fragment to display uploaded PDFs from Admin App

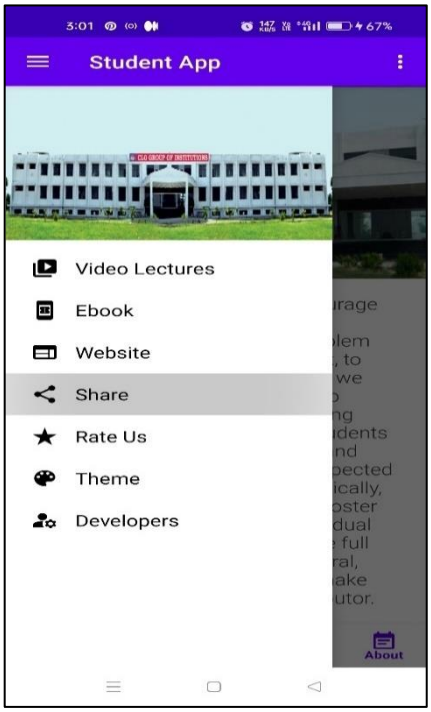


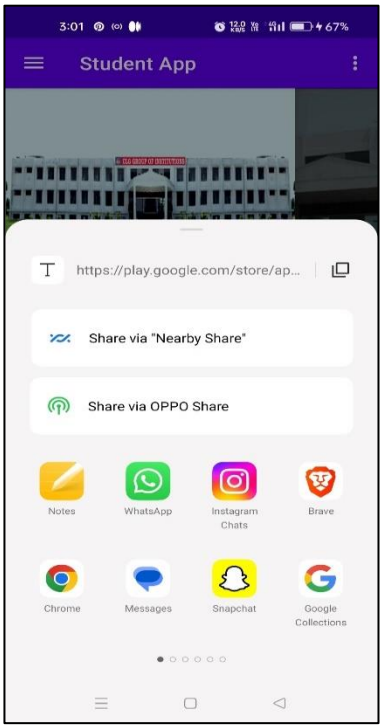


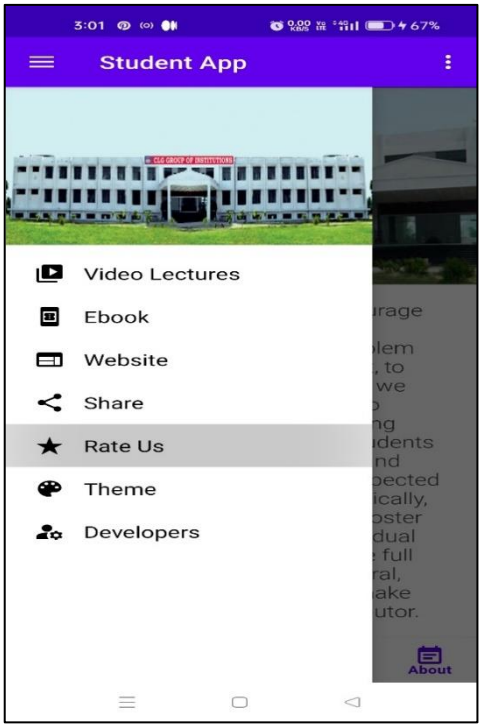
J. Website fragment to show the college website



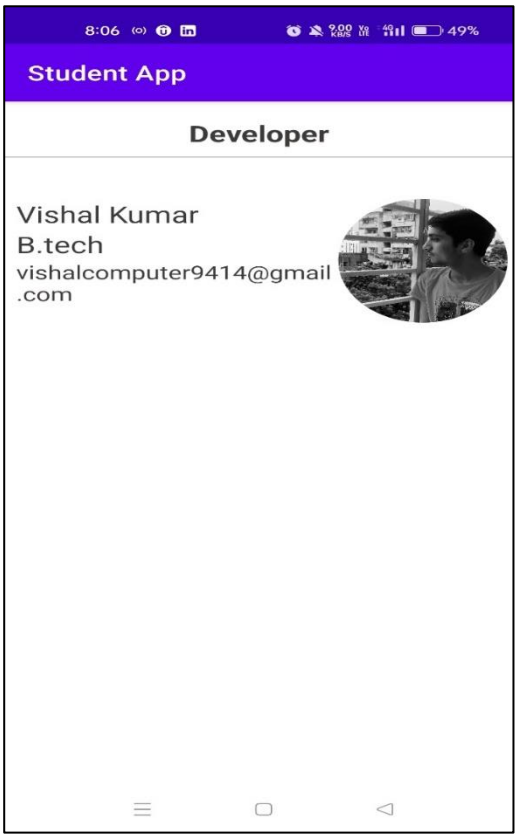
K. Share and rate us functionality

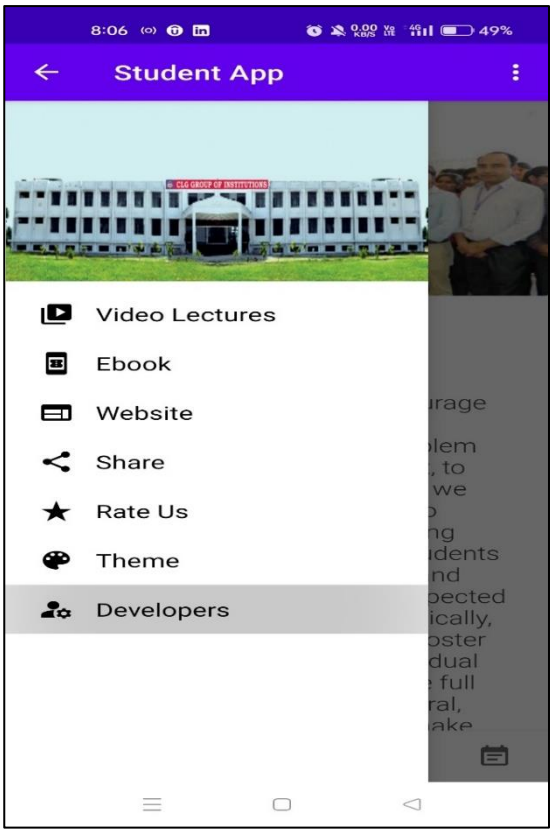






L. Developer option





CHAPTER 6

CONCLUSION & FUTURE SCOPE

CONCLUSION:

- The admin app provides login authentication for administrators using email and password.
- It includes functionality for uploading notices, gallery images, and PDFs with titles.
- The admin app also manages faculty information, allowing updates and deletions.
- Administrators can delete notices and log out from the app.

The student app offers the following features:

- Students can register using their name, email, and password, which is stored in Firebase.
- They can log in using their registered email and password or recover a forgotten password.
- The student app retrieves data from the admin app.
- It includes multiple fragments such as home, notice, faculty, gallery, and about.
- The home fragment showcases college images in a slider, displays college department information, contact details, and the college's location on a map.
- The notice fragment shows the latest notices uploaded by the admin, with the newest notices at the top.
- The faculty fragment displays faculty information, including names, positions, emails, and profile photos.
- The gallery fragment showcases various uploaded images categorized into sports activities, cultural events, and other events.
- The about fragment provides information about the college, available branches, and streams like civil, computer science, and mechanical.
- The navigation drawer offers options for video lectures, ebooks, the college's website, sharing and rating the app, and developer information.
- The option menu in the right corner provides a logout functionality that redirects students to the login page.

COLLEGE APP

FUTURE SCOPE:

Admin App:

- Implement a notification system to notify students about new notices or important updates.
- Introduce role-based access control to allow different levels of administration.
- Enhance the user interface and user experience for a more intuitive and visually appealing app.
- Integrate analytics to gather insights on app usage and user behavior.

Student App:

- Add a chat or discussion feature to facilitate communication among students and with faculty.
- Implement a calendar or timetable feature to display academic schedules, events, and deadlines.
- Enable offline access to important app features and cached data for better usability.
- Integrate a search functionality to allow students to easily find specific information within the app.
- Implement a feedback system to gather student opinions and suggestions for improvement.
- Expand the app to support multiple colleges or educational institutions, providing a customizable experience for each.
- Integrate social media sharing options to allow students to share content from the app with their peers.
- Improve accessibility features to ensure the app is usable for all students, including those with disabilities.
- Continuously update and maintain the app to support new OS versions and security measures.

CHAPTER 7

REFERENCES

- Android Developer Documentation. (n.d.). Retrieved from https://developer.android.com/docs
- Java Documentation. (n.d.). Retrieved from https://docs.oracle.com/en/java/
- Firebase Documentation. (n.d.). Retrieved from https://firebase.google.com/docs
- Android Studio Documentation. (n.d.). Retrieved from https://developer.android.com/studio/intro
- Agile Manifesto. (n.d.). Retrieved from https://agilemanifesto.org/
- Balci, O. (2019). Software engineering practices in industry: A systematic literature review. Information and Software Technology, 107, 252-266.
- Pressman, R. S., & Maxim, B. R. (2014). Software engineering: A practitioner's approach. McGraw-Hill Education.
- Sommerville, I. (2016). Software engineering. Pearson Education Limited.
- Ambler, S. W., & Lines, M. (2012). Disciplined agile delivery: A practitioner's guide toagile software delivery in the enterprise. IBM Press.
- Bass, L., Clements, P., & Kazman, R. (2012). Software architecture in practice. PearsonEducation.
- Graham, I. (2015). Agile project management: QuickStart guide. ClydeBank Media LLC.
- Martin, R. C. (2008). Clean code: A handbook of agile software craftsmanship. PearsonEducation.