# Table of Content

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Topic** | **Page. No** |
| **1.** | 1. **INTRODUCTION**    1. About the Company    2. Problem statement    3. Methodology    4. Organization | 1-2 |
| **2.** | **LITERATURE SURVEY**    2.1 Related work | 3-9 |
| **3.** | 1. **SYSTEM DEVELOPMENT**      * 1. Introduction to College ERP System   2. Problem Description   3. Software Requirements   4. Data Flow Diagrams | 10-15 |
| **4.** | **PERFORMANCE ANALYSIS**    4.1 Output  4.1.1 Admin Module  4.1.2 Faculty Module  4.1.3 Student Module  4.2 Code | 16-32 |
| **5.** | **5. CONCLUSIONS** | 33 |

# Abstract

COLLEGE ERP System is the best of its type of ERP and the specialty in this is that it is best suited for the working of University, Colleges. It is made after extensive study of all the departments of Colleges and is provided with the extract of everything, a college requires for their department management and student/staff management.

ERP means performing college register functions by using portable electronic device. ERP has now become the backbone of every sphere of industries being in manufacturing unit, trading unit or the service arena. ERP as its name suggest is not only software to but also a planning tool that will suggest you the best way to do the same task in a better way.

**Chapter-1** **Introduction**

# 1.1 About the Company

We are market leaders in delivering mobility solutions to entrepreneurs and enterprises looking to provide an omni channel experience to their customers. Our mission is to engineer and maximize mobile ROI.

Click labs was started as an incubator in 2014 as a technology solution provider that has taken mobile first approach. The company is headquartered in Chandigarh with about 400 employees and has offices in San Fracisco, Seattle, Brisbane and London. The company is focussed on providing mobility solutions to entrepreneurs and enterprises who want to provide their customers an omni channel experience. It has developed expertise in mobile, analytics and cloud services. The company has launched about 300+ platforms. Their mission is to engineer and maximize mobile ROI. Their products span across three categories Saas solutions, Platform framework and investment in startups. Jugnoo, App virality are some of the major successful ventures the company has worked upon.

## 1.2 Problem Statement

We propose an approach to develop a college enterprise resource planning system which will have three modules as admin, faculty, student.Admin has the functionality to add the faculty,give notification to the students.Faculty has the authority to add students and their details like marks and attendance.

## 1.3 Methodology

The project followed software development lifecycle. Initial training over android development was given using android development tool kit. Later the problem to be worked upon was identified and brainstormed. The modules were defined after consulting the senior colleagues and US design was provided. Once the features were finalized the development was done with cooperation from team members. Later code was subject to testing and the final deliverables were tested by the concerned team for approval.

## 1.4 Organisation

**Chapter 1:** In this chapter, the introduction to various concepts and techniques used in the implementation is covered.

**Chapter 2**: This includes reviewing relevant work from various research papers, books, journals and conferences. In this chapter, the extracts from assorted research papers on various situations are taken.

**Chapter 3:** Discusses about proposing a model suitable for developing the College ERP System using Android as a basic language.

**Chapter 4:** In this chapter results and screenshots are revealed to depict and defend the proposed work.

**Chapter 5:** This section concludes the whole work and also elaborates the scope of work that can be done in future which leaves an opportunity for upcoming students and scholars to further enhance this work.

# Chapter-2 Literature Survey

**INTRODUCTION TO PROGRAMMING LANGUAGE**

**Dart Programming Language for Flutter**

Dart is a versatile, open-source programming language developed by Google, optimized specifically for front-end development. In the context of Flutter, Dart plays a crucial role as it is the primary language used to build applications. Here’s a quick overview of its key features and advantages for Flutter development:

1. **Simplicity and Familiar Syntax**  
   Dart has a clear and straightforward syntax, making it easier for developers coming from languages like Java, JavaScript, or C++ to learn quickly. Its syntax resembles a mix of Java and JavaScript, which contributes to a smooth learning curve.
2. **Object-Oriented and Strongly Typed**  
   Dart is an object-oriented language with strong typing. This structure helps create organized and modular code, making it easy to manage even large applications. The type system helps in catching errors early, leading to more stable and reliable applications.
3. **Asynchronous Programming with async and await**  
   Dart includes robust support for asynchronous programming, which is essential for mobile apps that need to perform tasks like network calls or file reading without blocking the main user interface. With async and await, developers can handle time-consuming tasks seamlessly.
4. **JIT and AOT Compilation**  
   Dart uses Just-In-Time (JIT) and Ahead-Of-Time (AOT) compilation techniques. In development mode, JIT allows for hot-reloading, which speeds up testing and iteration. For release mode, AOT compilation produces optimized machine code, resulting in high-performance apps.
5. **Rich Standard Library**  
   Dart has a comprehensive standard library that provides a wealth of functionality, from handling collections to working with async data and HTTP requests. This makes it easier to perform common programming tasks without relying on external libraries.
6. **Easy to Use with Flutter**  
   As Dart was designed to work seamlessly with Flutter, it provides the optimal foundation for building responsive and beautiful UIs across platforms. Dart’s language features make it easy to create Flutter widgets and manage UI logic efficiently.

**Flutter Framework Overview**

Flutter is an open-source UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. It enables developers to create visually stunning applications quickly with high performance. Here are some of the key features and benefits of Flutter:

1. **Single Codebase for Multiple Platforms**  
   Flutter allows you to write one codebase that can run on Android, iOS, web, and desktop platforms. This drastically reduces development time and effort, as you don’t need to create separate codebases for each platform.
2. **Beautiful, Customizable UI**  
   Flutter comes with a rich set of pre-designed widgets that make it easy to build beautiful, customized user interfaces. Flutter’s widgets are also highly flexible, allowing developers to tweak every detail to match their design vision. Whether you want a Material Design look for Android or a Cupertino style for iOS, Flutter makes it possible.
3. **Fast Development with Hot Reload**  
   One of Flutter’s standout features is Hot Reload. It lets developers see the effects of code changes instantly without restarting the app, which speeds up the development and testing process. Hot Reload improves productivity and encourages experimentation with new features.
4. **Native-Like Performance**  
   Since Flutter is compiled to native machine code, it delivers high performance that feels like a truly native application. This is especially important for applications that require smooth animations or handle large amounts of data.
5. **Rich Ecosystem and Plugin Support**  
   Flutter has a vast ecosystem with thousands of plugins that allow developers to extend functionality, integrate with device hardware (like cameras, GPS, and sensors), use Firebase for backend services, and much more. This plugin support reduces the need to write platform-specific code and speeds up development.
6. **Growing Community and Google Support**  
   Flutter is maintained by Google and has a growing, active developer community. Google continues to support and update Flutter, ensuring that it stays up-to-date with the latest technologies and platform requirements. Additionally, a large community means plenty of resources, tutorials, and packages to help with development.
7. **Built-in Testing Support**  
   Flutter offers robust testing tools for unit tests, widget tests, and integration tests, making it easy to verify the functionality of your code at every level. The integrated testing support helps ensure app quality and reliability across platforms.
8. **Use of Dart Language**  
   Flutter applications are built using the Dart language, which is easy to learn, especially for developers familiar with object-oriented languages. Dart’s structure allows for efficient coding and debugging in Flutter, making the framework an ideal choice for modern, high-performance applications.

## INTRODUCTION TO ANDROID

Android is one of the world’s most widely used operating systems for mobile devices. Developed by Google, it powers billions of smartphones and tablets, offering users a rich and flexible experience across a range of device types. From its origins as an open-source platform based on the Linux kernel, Android has grown into a robust, feature-rich operating system that emphasizes flexibility, customizability, and access to a vast library of apps and services.

Android's extensive compatibility and strong integration with Google services make it a preferred choice for both users and developers. Its customizable nature, security features, and developer-friendly environment have also cemented Android’s position as a leading platform for app development. Here’s a closer look at what makes Android unique:

1. **Wide Device Compatibility**  
   Android runs on a vast range of devices, from budget smartphones to high-end tablets and even wearables. This compatibility allows users of various economic backgrounds to access Android devices, making it widely available and versatile.
2. **Customizability and Flexibility**  
   Android’s open-source nature allows manufacturers and users to customize the user interface and functionality. Users can change themes, add widgets, and customize app permissions, while manufacturers can add their own custom skins, such as Samsung’s One UI or Xiaomi’s MIUI.
3. **Google Services Integration**  
   Android is deeply integrated with Google services, making it convenient for users to access apps like Gmail, Google Maps, YouTube, and Google Drive seamlessly. This integration enhances productivity, connectivity, and access to essential services for Android users.
4. **Google Play Store**  
   The Google Play Store is the official app store for Android, hosting millions of apps that cater to all sorts of needs, from social media to productivity and games. It also provides a secure platform for users to download and update applications, while offering developers a global distribution channel.
5. **High Level of Security**  
   Android has multiple layers of security, from built-in malware protection to regular security updates. The operating system also offers features like app permissions, data encryption, and Google Play Protect, which scans apps for security threats.
6. **Multitasking and Notification System**  
   Android’s multitasking capability allows users to switch between apps seamlessly, enhancing productivity and user experience. The notification system keeps users informed about important updates, messages, and activities, making it easier to stay connected and organized.
7. **Regular Updates and Support for New Technology**  
   Android continuously evolves with regular updates, introducing new features, performance improvements, and security patches. Android also supports the latest technology, including 5G, foldable screens, and advanced AI features, ensuring compatibility with modern devices.
8. **Developer-Friendly Platform**  
   Android provides extensive resources and tools for developers, including Android Studio, a powerful development environment. The Android SDK (Software Development Kit) enables developers to build apps that take full advantage of the OS features, and the Kotlin and Java programming languages make Android development accessible to a broad range of developers.
9. **Vast User Base**  
   With billions of active devices worldwide, Android has a vast user base, making it an attractive platform for developers and businesses to reach a global audience. This wide reach provides developers with immense opportunities to innovate and deliver apps that make a difference.
10. **Supports a Variety of App Types**  
    Android supports various types of applications, including gaming, educational, e-commerce, social media, and productivity apps. This flexibility makes Android a versatile platform where users can find applications to meet virtually any need.

## Introduction to SQLite

SQLite is a lightweight, embedded database engine commonly used for mobile and embedded applications. Unlike other databases, SQLite is serverless, self-contained, and operates as a simple library that’s integrated directly into the application. It doesn’t require a separate server to manage data, making it ideal for small to medium-sized applications, including mobile apps and embedded systems.

SQLite is especially popular in mobile development frameworks like Android and Flutter, as it allows applications to store data locally on a device. Here’s a closer look at the key features and benefits of SQLite:

1. **Lightweight and Fast**  
   SQLite is a compact database engine with a small footprint, making it suitable for devices with limited storage and memory. It is highly optimized for performance, allowing applications to quickly perform database operations like reading and writing data.
2. **Serverless and Zero Configuration**  
   SQLite doesn’t require a dedicated server or configuration. It operates as a single-file database that can be embedded directly in the app. This simplifies deployment and reduces the need for complex database setup or management.
3. **Transactional Support with ACID Compliance**  
   SQLite is fully transactional and follows the ACID (Atomicity, Consistency, Isolation, Durability) principles, ensuring data integrity even in cases of unexpected app shutdowns. This means that all database changes are completed fully or not at all, providing reliable data handling.
4. **SQL-Based Language**  
   SQLite supports SQL (Structured Query Language) for data operations, making it easy for developers familiar with SQL to create, read, update, and delete data. This standard language simplifies database operations and allows for powerful data manipulation.
5. **Self-Contained and Cross-Platform**  
   SQLite is platform-independent and can run on nearly any device or operating system, making it ideal for cross-platform applications. It is written in C, allowing it to be used in a wide variety of environments, from mobile devices to IoT systems.
6. **Single Database File**  
   All data, including tables, indexes, and the entire database structure, is stored in a single file. This makes it easy to back up, move, or share data, as everything is contained in one file.
7. **Low Memory Usage**  
   SQLite is designed to use minimal memory, which is crucial for mobile and embedded systems with limited resources. It’s efficient enough to run smoothly even on low-end devices.
8. **Ideal for Local Storage in Mobile Apps**  
   SQLite is widely used for offline data storage in mobile applications, such as caching user preferences, storing session data, and recording offline transactions that can sync later when online. Its reliability and compact size make it perfect for use cases where an app needs to operate independently of an internet connection.
9. **Easy Integration with Mobile Development Frameworks**  
   SQLite integrates seamlessly with frameworks like Flutter (using the sqflite plugin) and Android, making it easy for developers to add local database functionality to their apps. The sqflite plugin in Flutter provides a simple API for performing database operations in a Flutter app.

### INTRODUCTION TO SDLC

The main function of a lifecycle model is to establish the order in which a project specifies, prototypes, designs, implements, reviews, tests, and performs its other activities. It establishes the criteria that you use to determine whether to proceed from one task to the next.

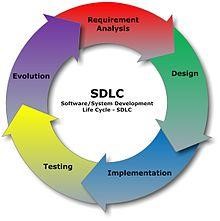
This includes the steps of

The main function of a lifecycle model is to establish the order in which a project specifies, prototypes, designs, implements, reviews, tests, and performs its other activities. It establishes the criteria that you use to determine whether to proceed from one task to the next.

All of the activities involved are highly related and interdependent, Therefore, in actual practice, several developmental activities can occur at the same time. So, different parts of a development project can be at different stages of the development cycle.

**SEVEN STEPS OF SDLC**

Given below are some phases, which are common in every SDLC model. Let us go through them to get an idea.

1. **Requirement Gathering and Analysis:** Requirement gathering and analysis is the first stage of any SDLC model. It is a brainstorming phase consisting of several sub stages to understand how feasible the idea if implemented is. In projects which requires software to be updated and modified then issues underlying of that software are found and resolved by finding ways to solve it. If brand new software is going to be developed, then every minute requirement regarding that software is looked in to.
2. **System Analysis:** This is the second phase of SDLC where the entire system is defined in detail. Here a detailed blueprint of all the processes is developed. As required, the system is divided into smaller parts to make it manageable and easier for software professionals who are currently working and will be latter working on the project.

1. **System Design:** Here, system analysts and designers using some tools create the logical design which is used to frame physical system.
2. **Coding:** This is the final stage of project development lifecycle where User testing is done by running the software on various systems and if non errors are found then it is considered to be a successful launch.
3. **Testing:** This is the final stage of project development lifecycle where User testing is done by running the software on various systems and if non errors are found then it is considered to be a successful launch.
4. **Implementation:** This is the final stage of project development lifecycle where User testing is done by running the software on various systems and if non errors are found then it is considered to be a successful launch.
5. **Maintenance:** When the software is implemented, it does not mean that the software is good as it is. All SDLC models include maintenance since there are absolutely no way that software will be working perfectly.

# Chapter-3 System Development

## 3.1 INTRODUCTION TO COLLEGE ERP SYSTEM

ERP as its name suggest is not only software but also a planning tool that will suggest you the best way to do the same task in a better way.

A College ERP system is an attempt to integrate functions across an institution by using An ERP for colleges is a single system to support rather than several small and different systems. Single application architecture with limited interfaces. Access to management information .Access to best practice systems and procedures. In this project, an attempt has been made to develop and deploy colleges ERP System on Android mobile phones.

An ERP for colleges is a single system to support rather than several small and different systems. Single application architecture with limited interfaces. Access to management information .Access to best practice systems and procedures. In this project, an attempt has been made to develop and deploy colleges ERP System on Android mobile phones.

The aim of this project is to create an android mobile application for ERP system that will integrate the various college activities like Faculty registration, student registration, upload student attendance, student marks and upload the information of placement drive in a single system with a common database that can be used by students, teachers and other staff members of educational institution

In other hand the rapid development in operating system of the mobile phones gives rise to the application development on large scale. The major reason behind the popularity of Android is that it is an open source system and a software development kit is provided which helps build and run android applications.

COLLEGE ERP System is the best of its type of ERP and the specialty in this is that it is best suited for the working of University, Colleges. It is made after extensive study of all the departments of Colleges and is provided with the extract of everything, a college requires for their department management and student/staff management.

ERP means performing college register functions by using portable electronic device. ERP has now become the backbone of every sphere of industries being in manufacturing unit, trading unit or the service arena. ERP as its name suggest is not only software to but also a planning tool that will suggest you the best way to do the same task in a better way.

## 3.2 PROBLEM DESCRIPTION

The software primarily uses Android client devices, such as mobile phones and tablets, to log in through an offline application. Instead of connecting through a wireless network, the application allows users to access and interact with a locally stored database. This offline setup ensures that users can view and update relevant information even without an internet connection.

In this offline college ERP system, the application retrieves needed information from a local database while allowing data entry based on actual situations. The local database acts as a bridge, storing essential campus-related information, which users can access directly through the app.

After installing the generated college ERP APK on an Android device, users can open the login page. By entering the student name and roll number as the password, students can log in. Since the app operates offline, it will authenticate the user against locally cached data. After a successful login, students can view attendance details, placement information, exam marks, and grade details—all stored in the local database.

## EXISTING SYSTEM

In the current system on most college campuses, information is typically shared through notices and websites, requiring students to frequently check physical notice boards or log into web portals for updates. Managing student details and academic information is often done manually, which can be time-consuming and prone to errors.

Today, there is a need to go beyond traditional forms of communication and leverage technology to streamline information sharing. However, existing systems lack a dedicated, efficient platform for accessing and managing academic details on mobile devices, especially in an offline environment. This limits the ability for real-time information access, which can be crucial for effective communication and updates among students and faculty.

## PROPOSED SYSTEM

The proposed college information system is a fully automated offline application developed using the Flutter framework with Dart programming and SQLite for data storage. This offline system ensures that students, faculty, and administrators can access and manage essential information directly on Android devices without relying on a constant internet connection.

In this system, the admin panel is designed within the app to manage faculty details and requirements, such as placement information and announcements, which are stored securely in the SQLite database. Admins can register faculty information and update important notices, which are then available offline to users across the campus.

Faculty members can register student details, input attendance, and add marks, all within the app. This information is stored in the local database, ensuring easy access and data security. Each student can log in using their roll number and password, enabling them to view attendance records, marks, placement details, and announcements.

This offline system utilizes a centralized SQLite database on each device, allowing for efficient data access and updates. The use of Flutter ensures a smooth, cross-platform user experience, and Dart programming enables seamless data processing and storage, making coordination within the app simple and effective even without internet connectivity.

## FEATURES

* **Login in College ERP**  
  A secure login feature allows students, faculty, and administrators to access the system using unique credentials. This helps ensure that each user’s data is protected and only accessible to authorized individuals.
* **Manage College Basic Data**  
  The system provides administrators with tools to manage essential college data, including updating general information, college announcements, and events, all within a centralized offline database.
* **Create Student Accounts**  
  Administrators and faculty can create and manage student accounts, allowing students to access their personalized homepage and view information specific to them, such as attendance, grades, and placement updates.
* **Create Faculty Accounts**  
  Faculty accounts can be created to enable teachers to log in, view their assigned duties, access student records, and update student-related data, such as marks and attendance records, directly within the app.
* **Manage Student Marks by Semester**  
  Faculty can add and update student marks on a semester basis, helping students track their academic performance over time. All records are stored locally, ensuring quick access without needing internet connectivity.
* **Manage Faculty Salary**  
  The system allows administrators to manage faculty salary information, keeping track of payment details and other payroll-related data securely within the app.
* **Easy-to-Use Interface**  
  The app is designed with a simple, user-friendly interface, making it easy for students, teachers, and administrators to navigate and use all features effectively.
* **Effective Management of Records**  
  The ERP system enables effective management of student, faculty, and administrative records, enhancing productivity and organization within the college environment.
* **Personalized Student Homepage**  
  Each student has a personalized homepage where they can view attendance, marks, placement information, and announcements, providing a tailored user experience.
* **Secure, Scalable, and Robust Platform**  
  Developed using Flutter, Dart, and SQLite, the app offers a secure, robust, and scalable offline platform that can be expanded as needed to accommodate additional features or users.

## BENEFITS

* **Fast and Efficient Information Access**  
  The offline system enables quick access to essential information, allowing students, faculty, and administrators to retrieve and update data seamlessly without needing internet connectivity.
* **Enhanced Process Efficiency**  
  Automating student and faculty data management simplifies previously manual tasks, streamlining everyday operations and saving time.
* **Greater Transparency and Accountability**  
  The system increases transparency by providing users with access to real-time data, ensuring that students, teachers, and administrators have a clear view of records and updates.
* **Better Decision-Making Support**  
  By consolidating data on attendance, grades, events, and faculty information in one place, the ERP system supports informed decision-making for both faculty and administrators.

## 3.3 SOFTWARE REQUIREMENTS

* **Operating System**
  + Windows 7/8/10
* **Front-End Tool**
  + Android (Using Flutter framework for app development)
* **Database**
  + SQLite (for offline data storage)
* **Client**
  + Any Android-enabled device for students, faculty, and administrators

**Hardware Requirements**

1. **Client (For Students, Faculty, Administrators)**
   * **Processor**: Intel Pentium 4 or equivalent processor with a minimum speed of 2.0 GHz
   * **RAM**: Minimum 512MB
   * **Hard Disk**: Minimum 30GB of free space
2. **Server (for Centralized Data Storage and Management, if applicable in future)**
   * **RAM**: Minimum 1GB
   * **Hard Disk**: Minimum 8GB
   * **Processor**: Any modern microprocessor
   * **Power Backup**: Required to ensure uninterrupted access to the system

### 3.4 DATA FLOW DIAGRAMS

The proposed college information system is a fully automated offline application developed using the Flutter framework with Dart programming and SQLite for data storage. This offline system ensures that students, faculty, and administrators can access and manage essential information directly on Android devices without relying on a constant internet connection.

In this system, the admin panel is designed within the app to manage faculty details and requirements, such as placement information and announcements, which are stored securely in the SQLite database. Admins can register faculty information and update important notices, which are then available offline to users across the campus.

Faculty members can register student details, input attendance, and add marks, all within the app. This information is stored in the local database, ensuring easy access and data security. Each student can log in using their roll number and password, enabling them to view attendance records, marks, placement details, and announcements.

The login system ensures that each user (admin, faculty, or student) has personalized access to the data. The SQLite database stores login credentials and authentication data locally, making the login process fast and secure.

## LOGIN TABLE

| **Field Name** | **Description** |
| --- | --- |
| **UserID** | Unique identifier for each user (student/faculty/admin). |
| **Username** | Roll number (for students) or ID (for faculty/admin). |
| **Password** | Password for user authentication. |
| **Role** | Specifies the user's role (student, faculty, or admin). |
| **LastLogin** | Timestamp of the last successful login. |

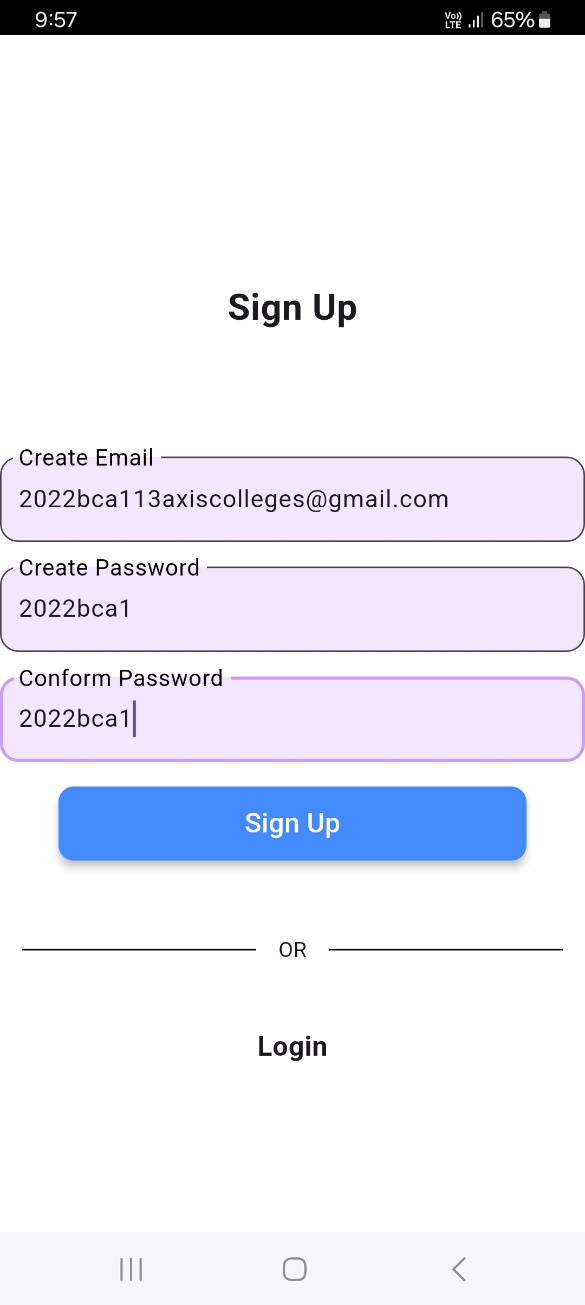
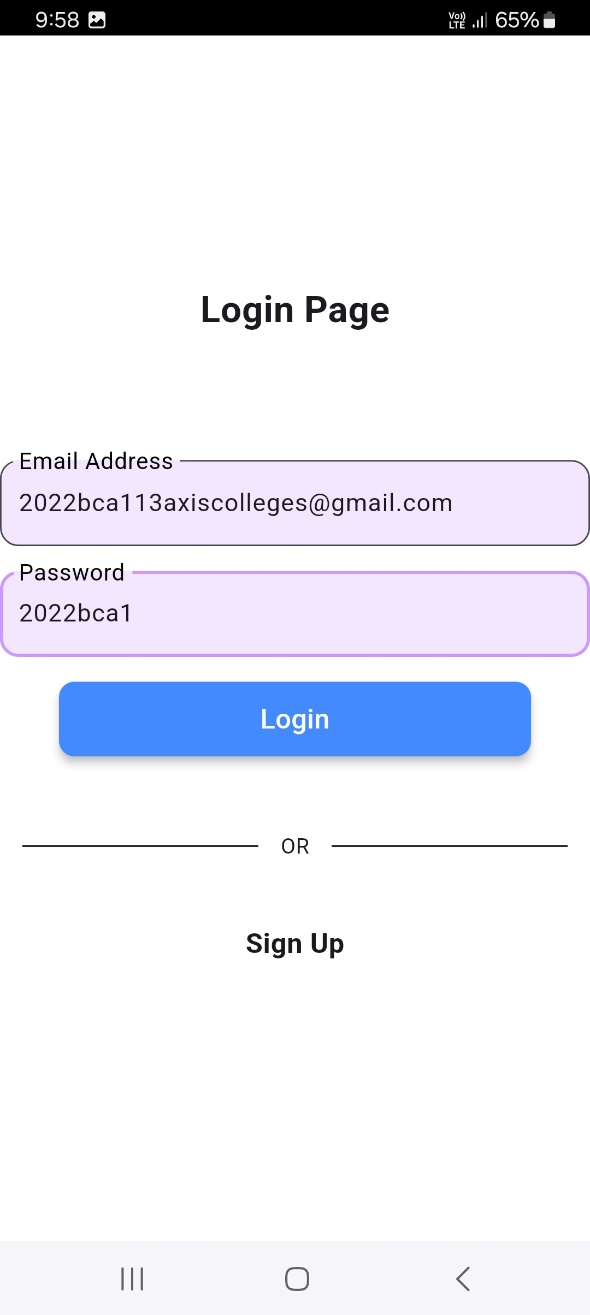
The offline system ensures that the user authentication process is handled efficiently, and all login data is stored in a local database, allowing users to access the app anytime, even without internet connectivity.

# Chapter-4

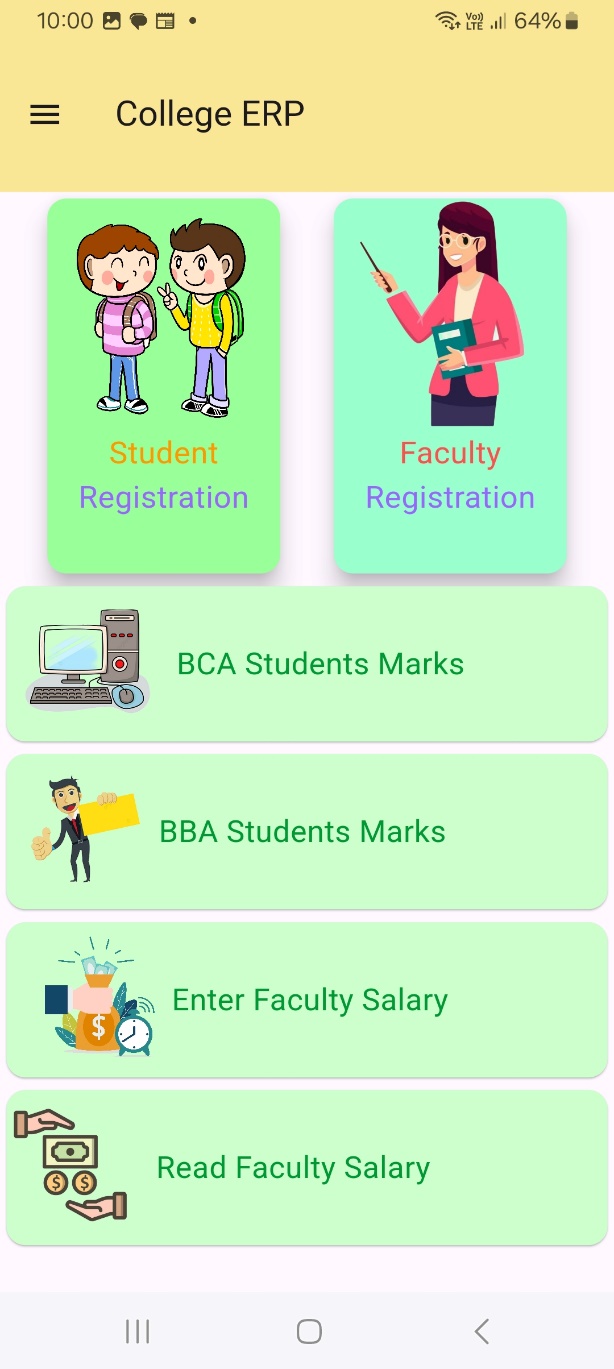
**Performance Analysis**

## 4.1 OUTPUT

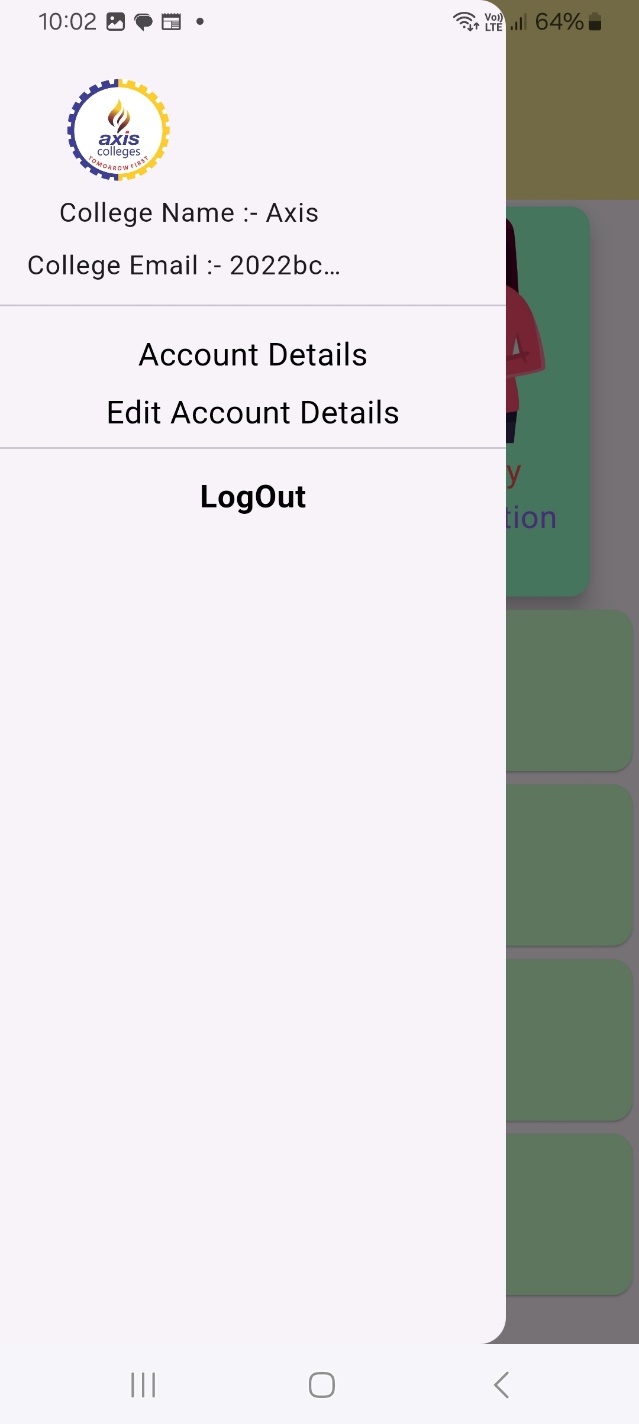
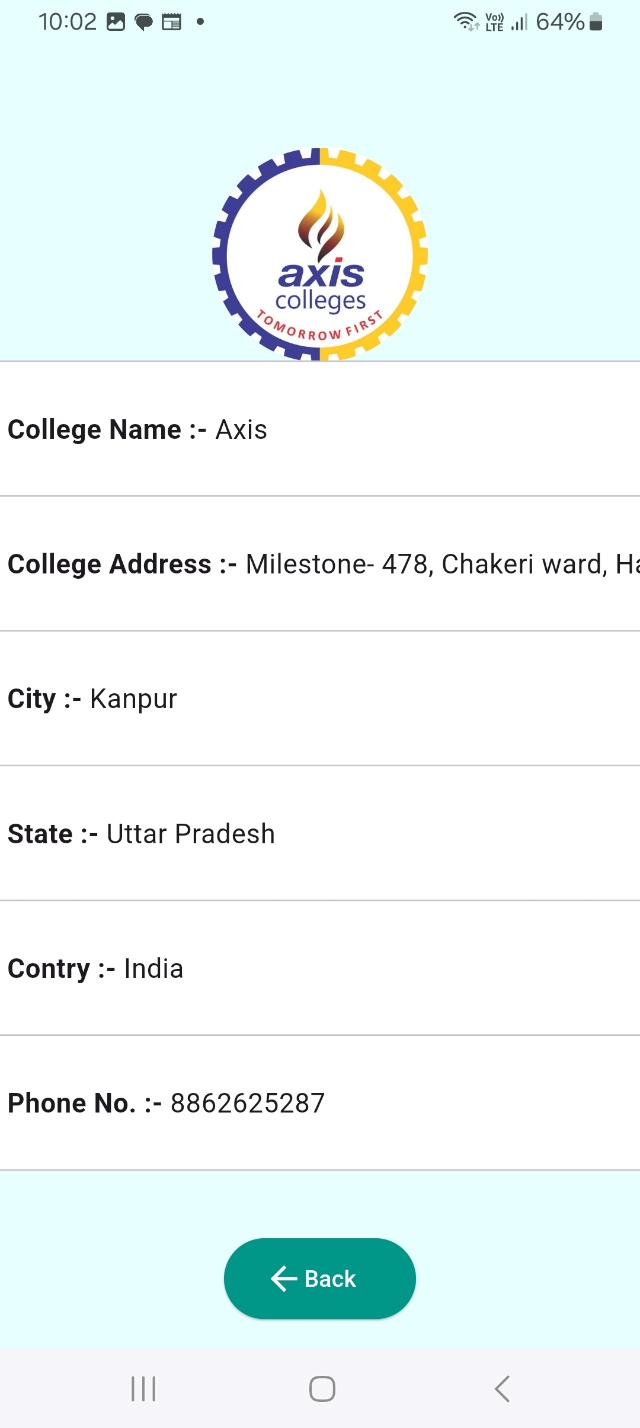
**Sign Up Page Login Page**

**** 

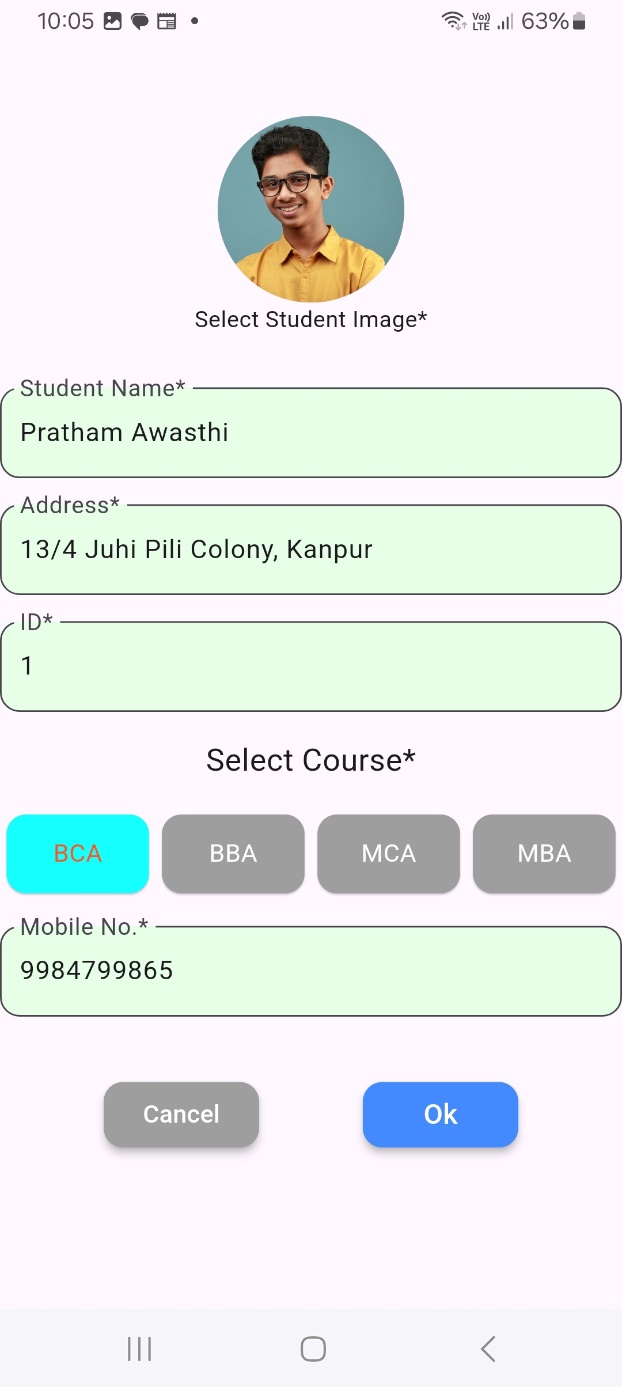
**Create College Account Home Page**

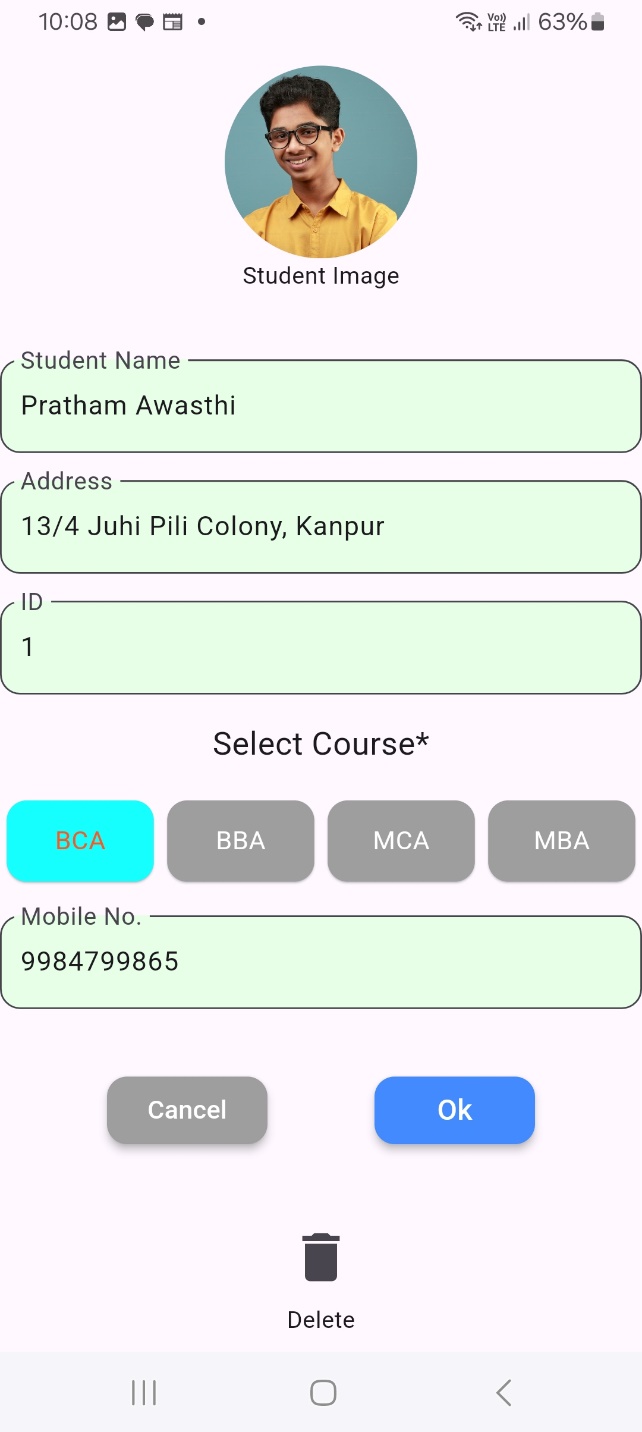
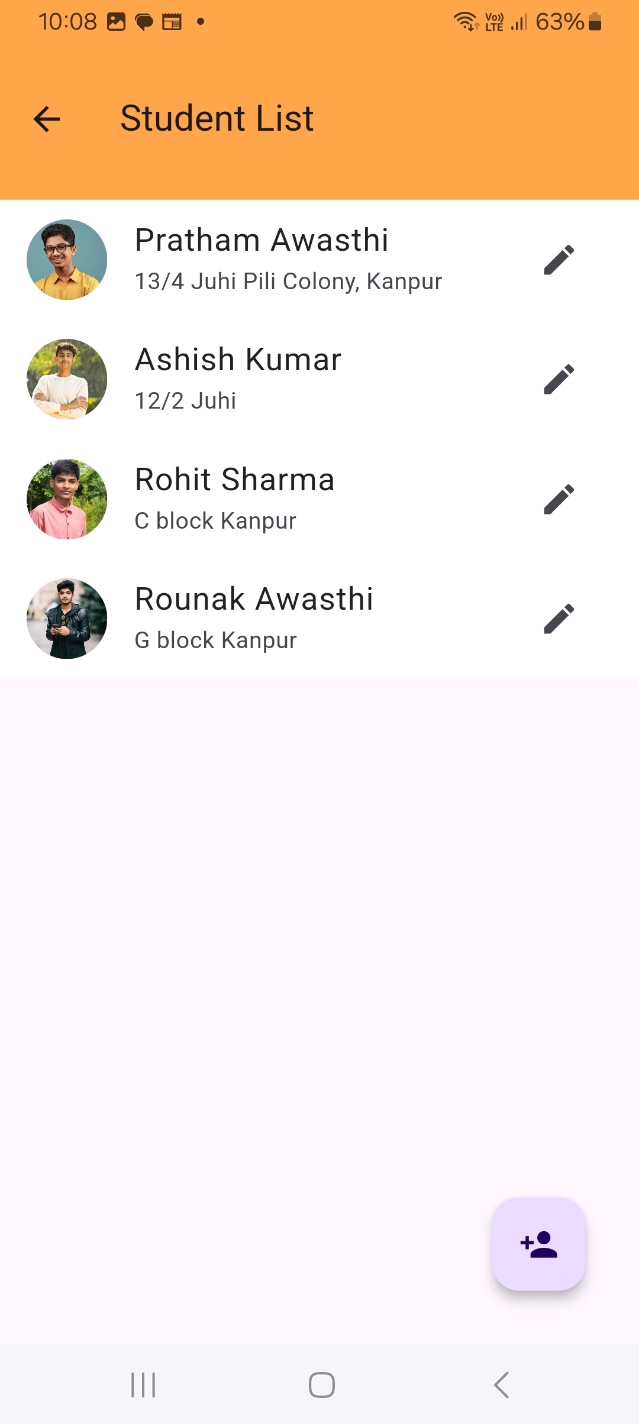
**Drawe Page Read College Detail’s Page**

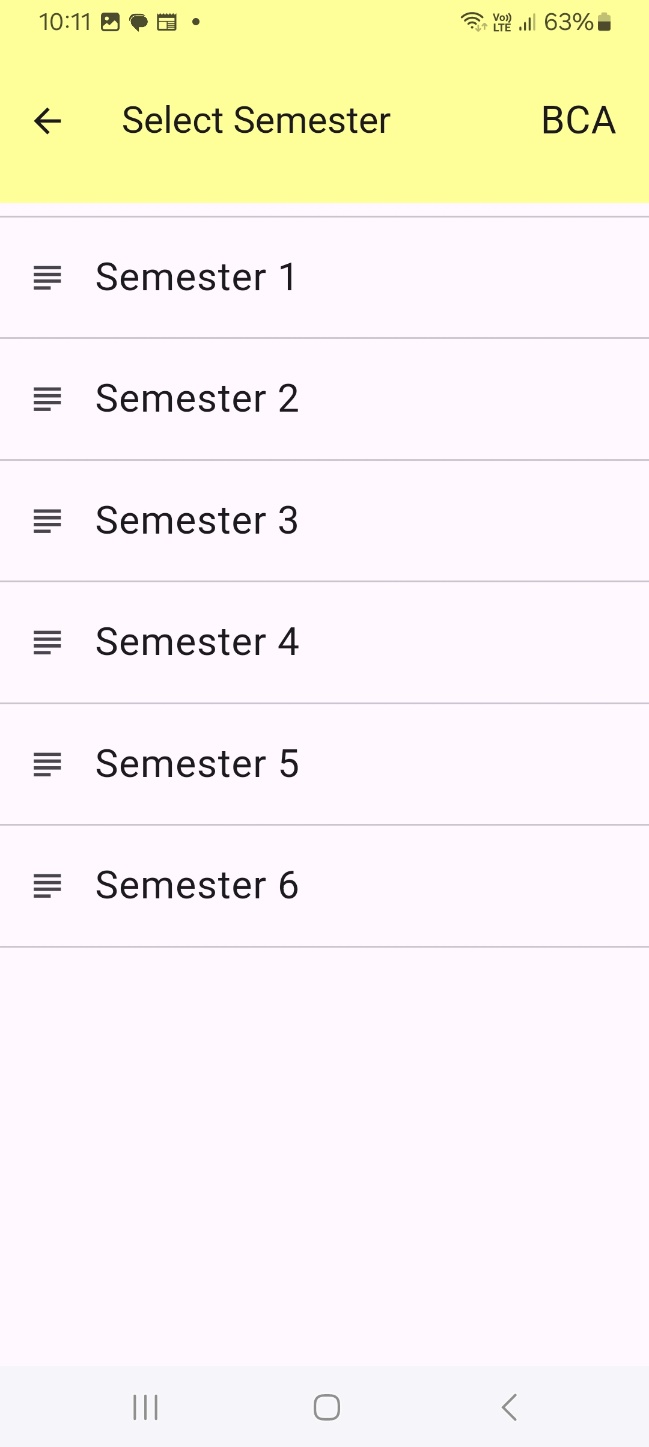
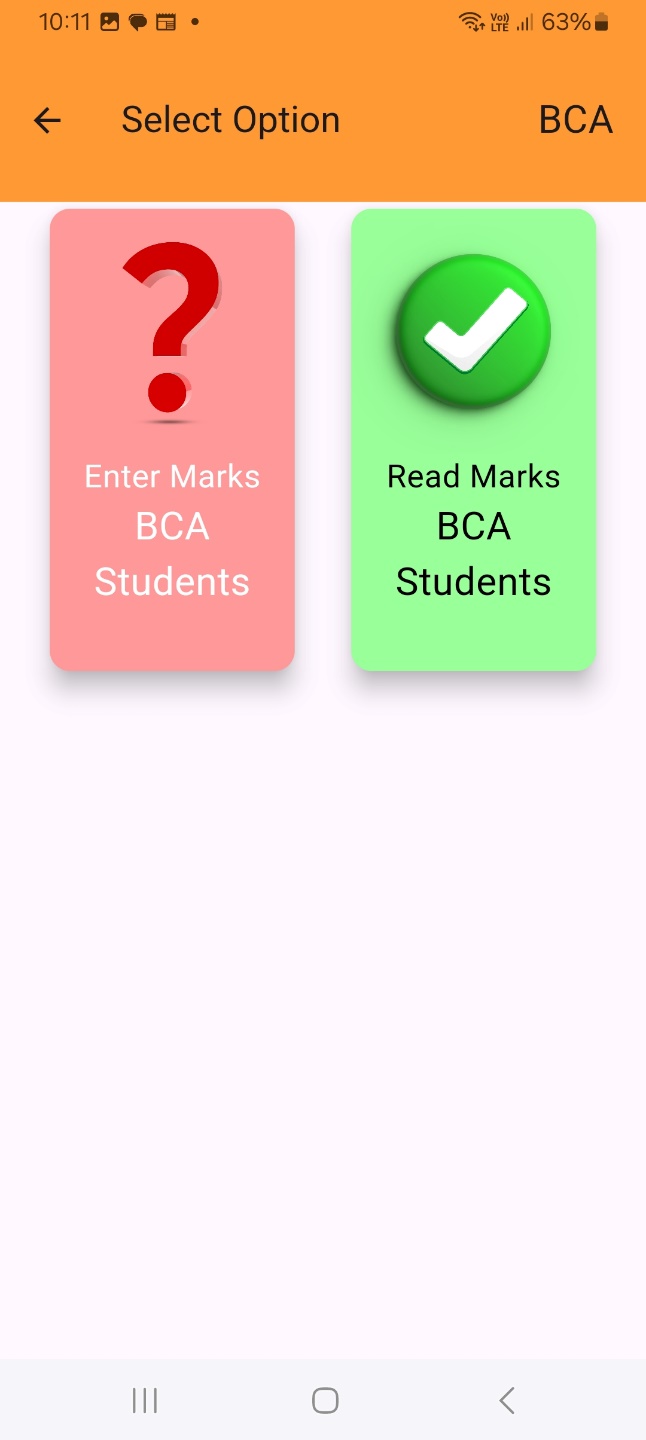
**Edit College Detail’s Page Create Student Account Page**

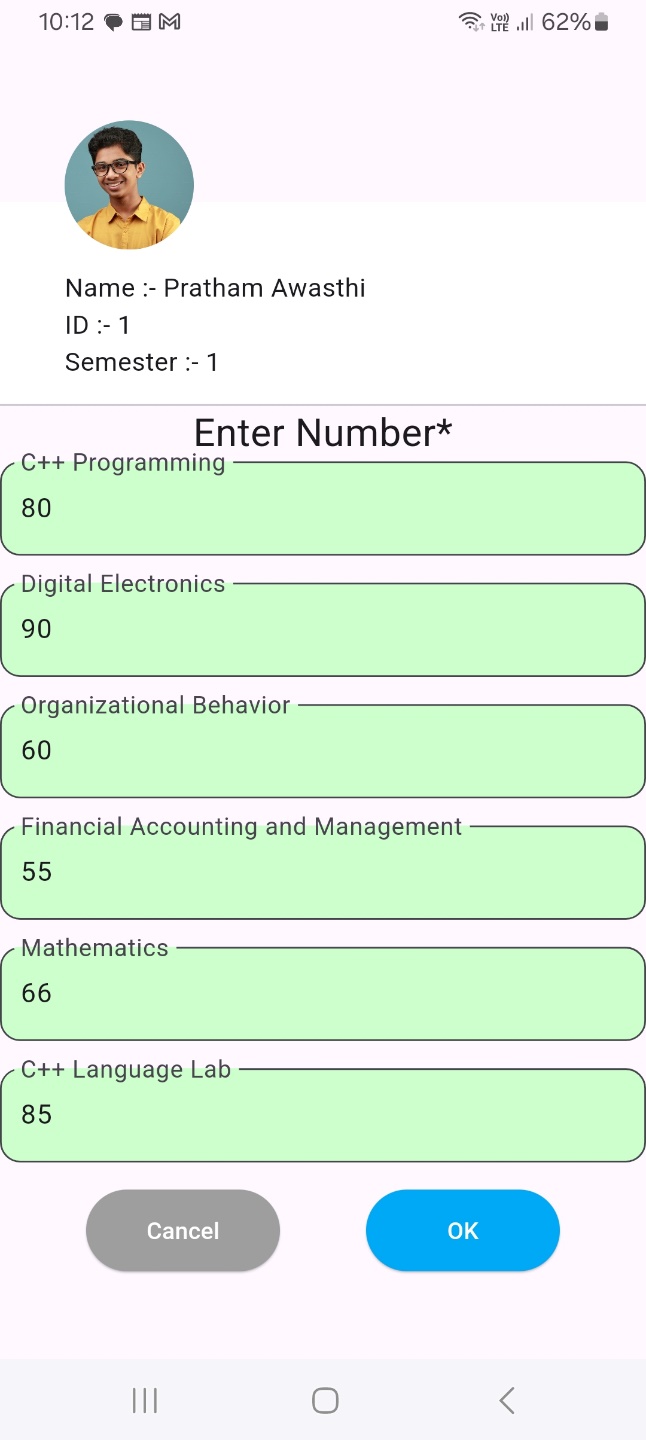
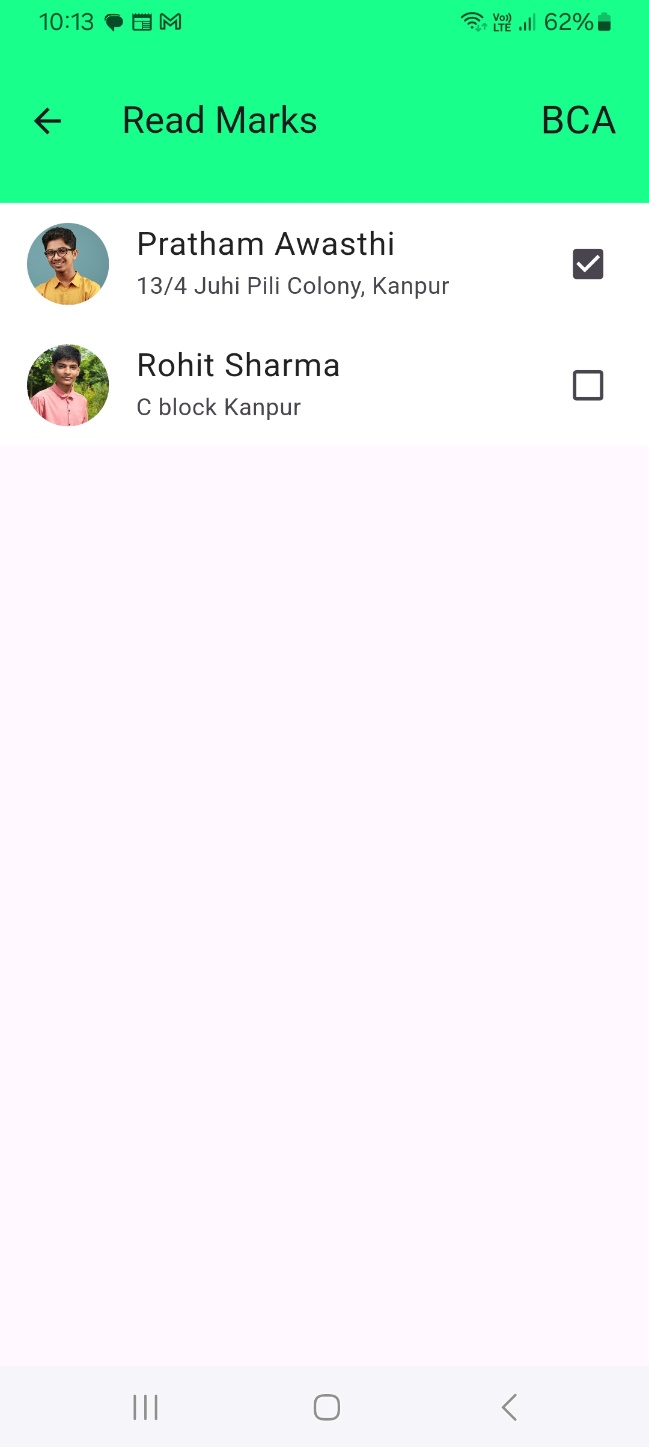
**Edit Student Detail’s Page Student List Page**

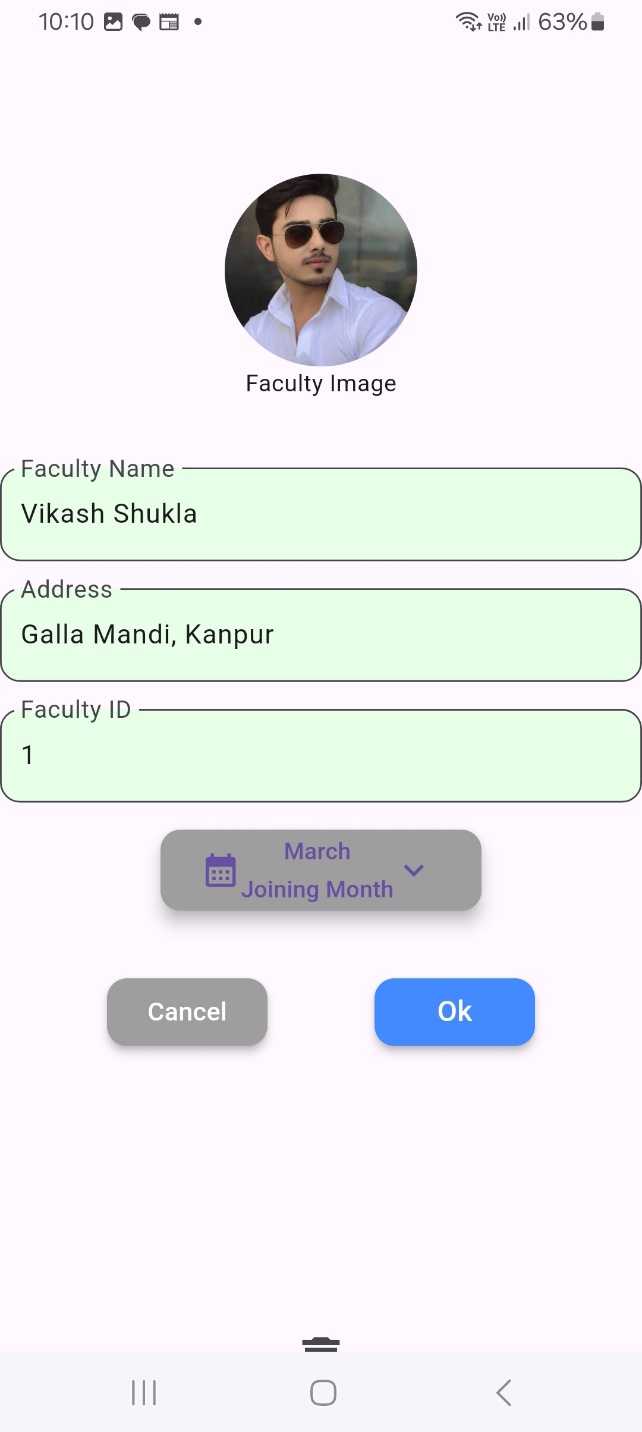
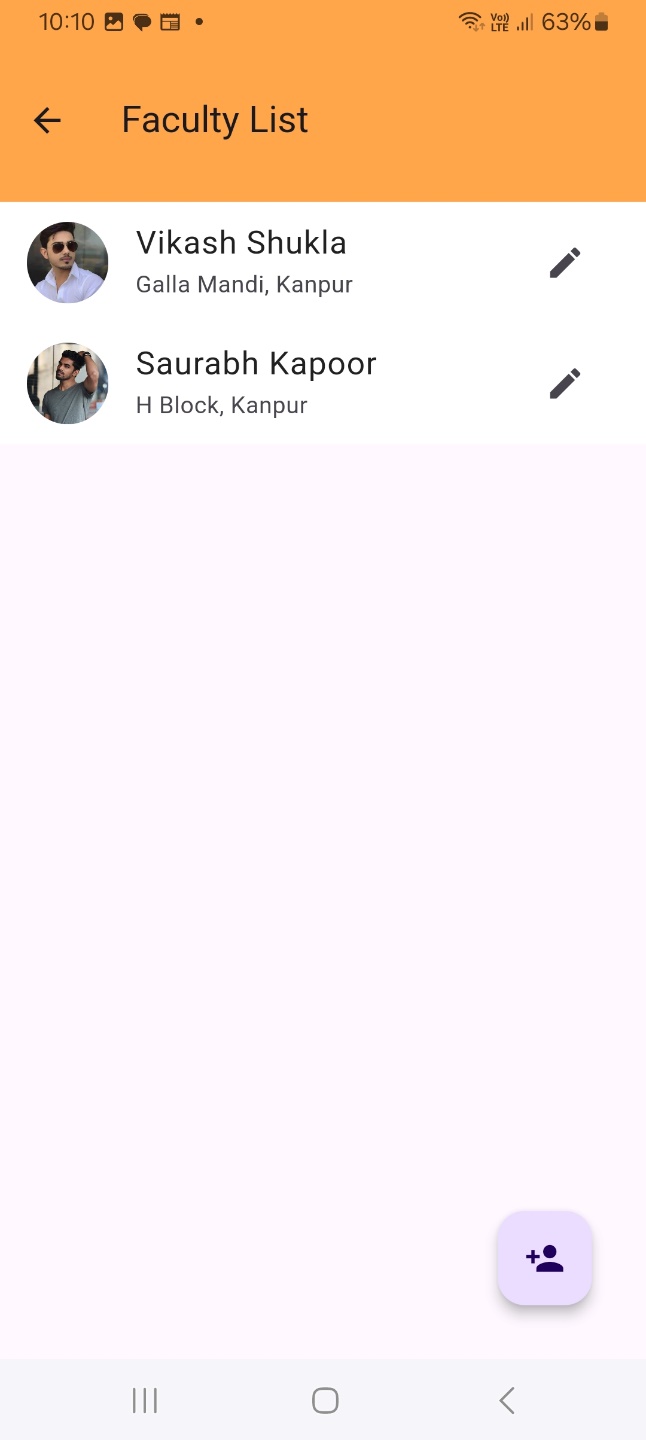
**Select Semester Page Enter Marks or Read Marks Page**

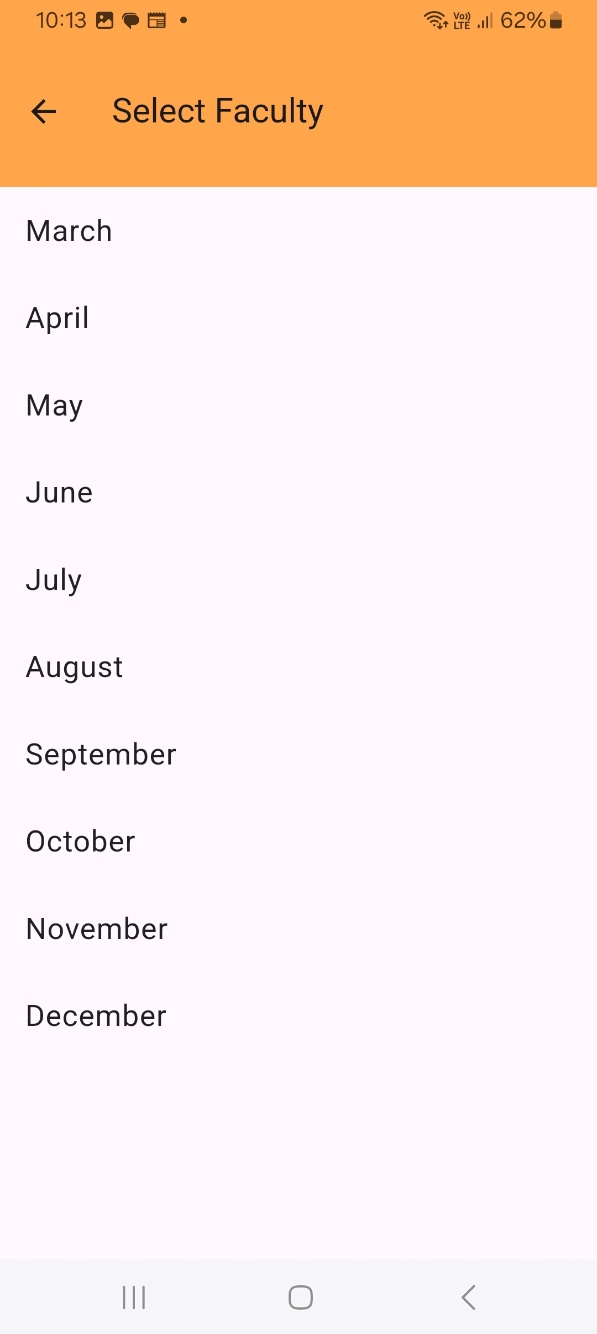
**Enter Student Marks Page Student Marks List Page**

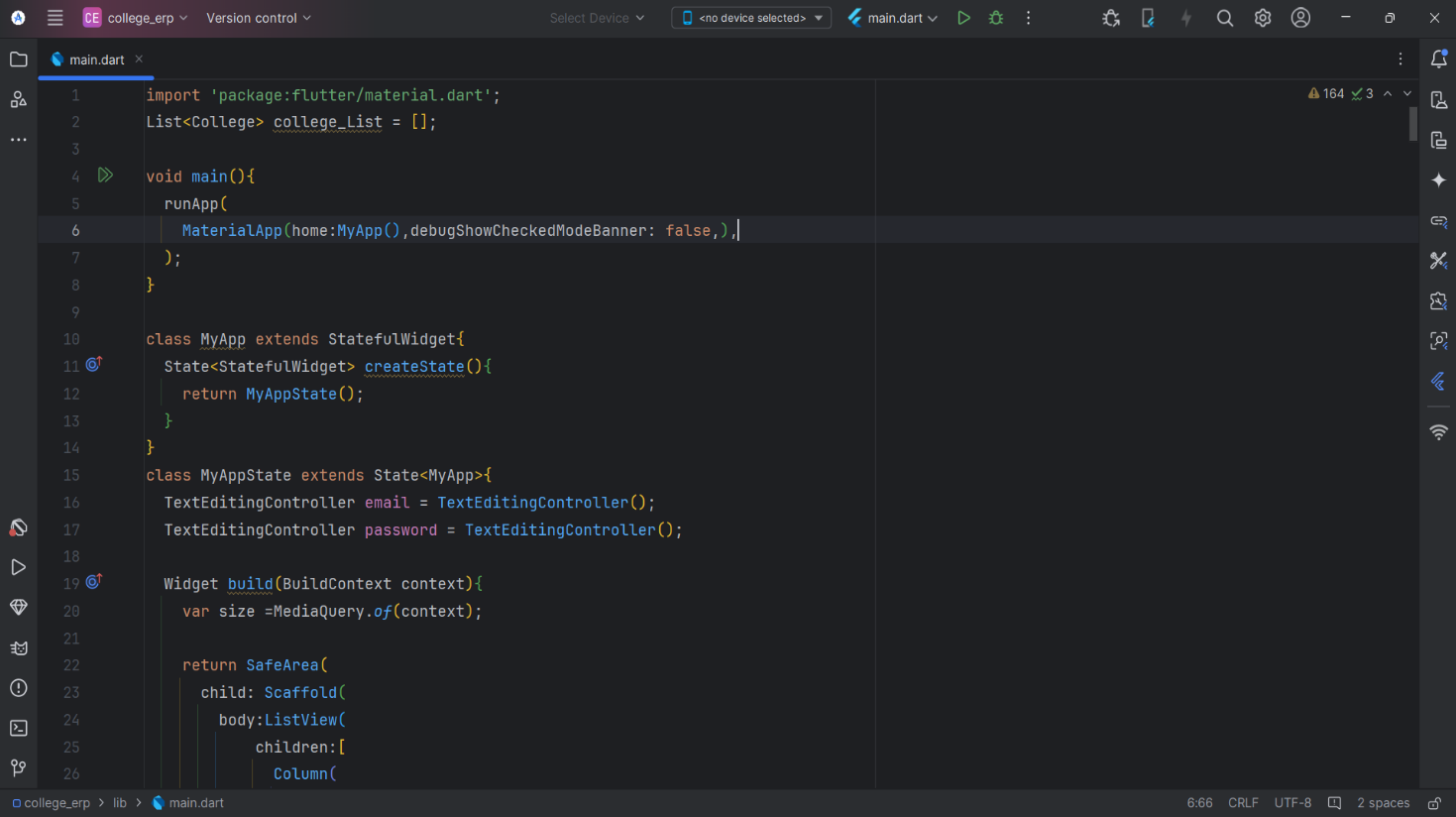
**Create Faculty Account Page Faculty List Page**

**Select Faculty Joining Month Page Enter Salary Page**

# IDE Screenshot



# 4.2 Code

# import 'package:flutter/material.dart';

# import 'package:college\_erp/UX.dart';

# import 'ERP\_Home\_Page.dart';

# List<College> college\_List = [];

# void main(){

# runApp(

# MaterialApp(home:MyApp(),debugShowCheckedModeBanner: false,),

# );

# }

# class MyApp extends StatefulWidget{

# State<StatefulWidget> createState(){

# return MyAppState();

# }

# }

# class MyAppState extends State<MyApp>{

# TextEditingController email = TextEditingController();

# TextEditingController password = TextEditingController();

# Widget build(BuildContext context){

# var size =MediaQuery.of(context);

# return SafeArea(

# child: Scaffold(

# body:ListView(

# children:[

# Column(

# children:[

# Padding(

# padding: EdgeInsets.fromLTRB(0,size.size.height\*0.2,0,0),

# child: Text("Login Page",style:TextStyle(fontSize:24,fontWeight:FontWeight.bold),),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0,size.size.height\*0.1,0,0),

# child: TextField(

# controller:email,

# decoration:InputDecoration(

# filled:true,

# fillColor:Color(0xfff2e6ff),

# labelText:"Email Address",

# labelStyle:TextStyle(fontSize:20,color:Colors.black),

# border:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# ),

# focusedBorder:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# borderSide:BorderSide(

# color:Color(0xffcc99ff),

# width:2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0,size.size.height\*0.02,0,0),

# child: TextField(

# controller:password,

# decoration:InputDecoration(

# filled:true,

# fillColor:Color(0xfff2e6ff),

# labelText:"Password",

# labelStyle:TextStyle(fontSize:20,color:Colors.black),

# border:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# ),

# focusedBorder:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# borderSide:BorderSide(

# color:Color(0xffcc99ff),

# width:2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0,size.size.height\*0.02,0,0),

# child: SizedBox(

# child:ElevatedButton(

# onPressed:(){

# String? em = email.text.trim().toString();

# String? ps = password.text.trim().toString();

# if(em==""||ps==""){

# return;

# }

# for( int i=0;i<college\_List.length;i++){

# if(college\_List[i].email==em){

# if(college\_List[i].password==ps){

# Navigator.pushReplacement(context, MaterialPageRoute(builder: (context)=>HomePage(college\_List[i])));

# break;

# }

# break;

# }

# }

# },

# child:Text("Login",style:TextStyle(color:Colors.white,fontSize:18)),

# style:ElevatedButton.styleFrom(

# backgroundColor:Colors.blueAccent,

# elevation:5,

# shape:RoundedRectangleBorder(

# borderRadius:BorderRadius.all(Radius.circular(10)),

# ),

# ),

# ),

# height:size.size.height\*0.06,

# width:size.size.width\*0.8,

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0,size.size.height\*0.06,0,0),

# child: Row(

# children:[

# Container(

# height:1,

# width:size.size.width\*0.4,

# color:Colors.black,

# ),

# Text("OR"),

# Container(

# height:1,

# width:size.size.width\*0.4,

# color:Colors.black,

# ),

# ],

# mainAxisAlignment:MainAxisAlignment.spaceEvenly,

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0,size.size.height\*0.05,0,0),

# child: InkWell(

# onTap:(){

# Navigator.pushReplacement(context,MaterialPageRoute(builder:(context)=>SignUp()));

# },

# child: Text("Sign Up",style:TextStyle(fontSize:18,fontWeight:FontWeight.bold),)),

# ),

# ],

# ),

# ]

# ),

# backgroundColor:Colors.white,

# ),

# );

# }

# }

# class SignUp extends StatefulWidget{

# State<StatefulWidget> createState(){

# return SignUpState();

# }

# }

# class SignUpState extends State<SignUp> {

# TextEditingController email = TextEditingController();

# TextEditingController password = TextEditingController();

# TextEditingController conformpassword = TextEditingController();

# TextEditingController college\_Name = TextEditingController();

# TextEditingController address = TextEditingController();

# TextEditingController city = TextEditingController();

# TextEditingController state = TextEditingController();

# TextEditingController contry = TextEditingController();

# TextEditingController phone = TextEditingController();

# Widget build(BuildContext context) {

# var size = MediaQuery.of(context);

# return SafeArea(

# child: Scaffold(

# body: ListView(

# children: [

# Column(

# children: [

# Padding(

# padding: EdgeInsets.fromLTRB(

# 0, size.size.height \* 0.2, 0, 0),

# child: Text("Sign Up", style: TextStyle(

# fontSize: 24, fontWeight: FontWeight.bold),),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.1, 0, 0),

# child: TextField(

# controller: email,

# decoration: InputDecoration(

# filled: true,

# fillColor: Color(0xfff2e6ff),

# labelText: "Create Email",

# labelStyle: TextStyle(

# fontSize: 20, color: Colors.black),

# border: OutlineInputBorder(

# borderRadius: BorderRadius.all(Radius.circular(12)),

# ),

# focusedBorder: OutlineInputBorder(

# borderRadius: BorderRadius.all(Radius.circular(12)),

# borderSide: BorderSide(

# color: Color(0xffcc99ff),

# width: 2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(

# 0, size.size.height \* 0.02, 0, 0),

# child: TextField(

# controller: password,

# decoration: InputDecoration(

# filled: true,

# fillColor: Color(0xfff2e6ff),

# labelText: "Create Password",

# labelStyle: TextStyle(

# fontSize: 20, color: Colors.black),

# border: OutlineInputBorder(

# borderRadius: BorderRadius.all(Radius.circular(12)),

# ),

# focusedBorder: OutlineInputBorder(

# borderRadius: BorderRadius.all(Radius.circular(12)),

# borderSide: BorderSide(

# color: Color(0xffcc99ff),

# width: 2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

# child: TextField(

# controller:conformpassword,

# decoration: InputDecoration(

# filled: true,

# fillColor: Color(0xfff2e6ff),

# labelText: "Conform Password",

# labelStyle: TextStyle(

# fontSize: 20, color: Colors.black),

# border: OutlineInputBorder(

# borderRadius: BorderRadius.all(Radius.circular(12)),

# ),

# focusedBorder: OutlineInputBorder(

# borderRadius: BorderRadius.all(Radius.circular(12)),

# borderSide: BorderSide(

# color: Color(0xffcc99ff),

# width: 2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

# child: SizedBox(

# child: ElevatedButton(

# onPressed: () {

# String? e = email.text.trim().toString();

# String? p = password.text.trim().toString();

# String? cp = conformpassword.text.trim().toString();

# if(e=="" || p=="" || cp==""){

# return;

# }

# if(p!=cp){

# return;

# }

# College ob = new College(e,p);

# college\_List.add(ob);

# Navigator.pushReplacement(

# context,

# MaterialPageRoute(builder: (context)=>create\_Account(context,ob)),

# );

# },

# child: Text("Sign Up", style: TextStyle(

# color: Colors.white, fontSize: 18)),

# style: ElevatedButton.styleFrom(

# backgroundColor: Colors.blueAccent,

# elevation: 5,

# shape: RoundedRectangleBorder(

# borderRadius: BorderRadius.all(Radius.circular(10)),

# ),

# ),

# ),

# height: size.size.height \* 0.06,

# width: size.size.width \* 0.8,

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(

# 0, size.size.height \* 0.06, 0, 0),

# child: Row(

# children: [

# Container(

# height: 1,

# width: size.size.width \* 0.4,

# color: Colors.black,

# ),

# Text("OR"),

# Container(

# height: 1,

# width: size.size.width \* 0.4,

# color: Colors.black,

# ),

# ],

# mainAxisAlignment: MainAxisAlignment.spaceEvenly,

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(

# 0, size.size.height \* 0.05, 0, 0),

# child: InkWell(

# onTap: () {

# Navigator.pushReplacement(context, MaterialPageRoute(

# builder: (context) => MyApp()));

# },

# child: Text("Login", style: TextStyle(

# fontSize: 18, fontWeight: FontWeight.bold),)),

# ),

# ],

# ),

# ]

# ),

# backgroundColor:Colors.white,

# ),

# );

# }

# Widget create\_Account(BuildContext context, College ob){

# var size = MediaQuery.of(context);

# return SafeArea(

# child: Scaffold(

# body:ListView(

# children:[

# Column(

# children: [

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.1, 0, 0),

# child: Text("Create Account", style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.1, 0, 0),

# child: TextField(

# controller:college\_Name,

# decoration:InputDecoration(

# label:Text("Collage Name"),

# labelStyle:TextStyle(fontSize:18,color:Colors.black),

# filled:true,

# fillColor:Color(0xfff2e6ff),

# border:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12))

# ),

# focusedBorder:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# borderSide:BorderSide(

# color:Colors.purple,

# width:2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

# child: TextField(

# controller:address,

# decoration:InputDecoration(

# label:Text("Address"),

# labelStyle:TextStyle(fontSize:18,color:Colors.black),

# filled:true,

# fillColor:Color(0xfff2e6ff),

# border:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12))

# ),

# focusedBorder:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# borderSide:BorderSide(

# color:Colors.purple,

# width:2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

# child: TextField(

# controller:city,

# decoration:InputDecoration(

# label:Text("City"),

# labelStyle:TextStyle(fontSize:18,color:Colors.black),

# filled:true,

# fillColor:Color(0xfff2e6ff),

# border:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12))

# ),

# focusedBorder:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# borderSide:BorderSide(

# color:Colors.purple,

# width:2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

# child: TextField(

# controller:state,

# decoration:InputDecoration(

# label:Text("State"),

# labelStyle:TextStyle(fontSize:18,color:Colors.black),

# filled:true,

# fillColor:Color(0xfff2e6ff),

# border:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12))

# ),

# focusedBorder:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# borderSide:BorderSide(

# color:Colors.purple,

# width:2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

# child: TextField(

# controller:contry,

# decoration:InputDecoration(

# label:Text("Contry"),

# labelStyle:TextStyle(fontSize:18,color:Colors.black),

# filled:true,

# fillColor:Color(0xfff2e6ff),

# border:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12))

# ),

# focusedBorder:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# borderSide:BorderSide(

# color:Colors.purple,

# width:2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

# child: TextField(

# controller:phone,

# decoration:InputDecoration(

# label:Text("Phone No."),

# labelStyle:TextStyle(fontSize:18,color:Colors.black),

# filled:true,

# fillColor:Color(0xfff2e6ff),

# border:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12))

# ),

# focusedBorder:OutlineInputBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# borderSide:BorderSide(

# color:Colors.purple,

# width:2,

# ),

# ),

# ),

# ),

# ),

# Padding(

# padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

# child: SizedBox(

# height:size.size.height\*0.06,

# width:size.size.width\*0.5,

# child: ElevatedButton(

# onPressed:(){

# String cm = college\_Name.text.trim().toString();

# String ad = address.text.trim().toString();

# String cy = city.text.trim().toString();

# String st = state.text.trim().toString();

# String co = contry.text.trim().toString();

# String ph = phone.text.trim().toString();

# if(cm=="" || ad=="" || cy=="" || st=="" || co=="" || ph==""){

# return;

# }

# ob.input\_Data(cm,ad,cy,st,co,ph);

# Navigator.pushReplacement(context,MaterialPageRoute(builder:(context)=>HomePage(ob)));

# },

# child:Text("Create",style:TextStyle(color:Colors.white,fontSize:18,fontWeight:FontWeight.bold)),

# style:ElevatedButton.styleFrom(

# shape:RoundedRectangleBorder(

# borderRadius:BorderRadius.all(Radius.circular(12)),

# ),

# backgroundColor:Colors.blue,

# ),

# ),

# ),

# ),

# ],

# ),

# ],

# ),

# backgroundColor:Colors.white,

# ),

# );

# }

# }

import 'dart:io';

import 'package:college\_erp/BBA\_marks.dart';

import 'package:college\_erp/BCA\_marks.dart';

import 'package:college\_erp/Faculty.dart';

import 'package:college\_erp/Faculty\_Salary.dart';

import 'package:college\_erp/Student\_Page.dart';

import 'package:college\_erp/faculty\_Salary\_Read.dart';

import 'package:college\_erp/main.dart';

import 'package:flutter/material.dart';

import 'package:image\_picker/image\_picker.dart';

import 'package:provider/provider.dart';

import 'Home\_Povider.dart';

import 'UX.dart';

class HomePage extends StatefulWidget{

College? ob;

HomePage(this.ob);

State<StatefulWidget> createState(){

return HomePageState(ob);

}

}

class HomePageState extends State<HomePage>{

TextEditingController email = TextEditingController();

TextEditingController password = TextEditingController();

TextEditingController conformpassword = TextEditingController();

TextEditingController college\_Name = TextEditingController();

TextEditingController address = TextEditingController();

TextEditingController city = TextEditingController();

TextEditingController state = TextEditingController();

TextEditingController contry = TextEditingController();

TextEditingController phone = TextEditingController();

final pick\_image =ImagePicker();

College? ob;

HomePageState(this.ob){

college\_Name.text=ob!.college\_Name!;

address.text=ob!.address!;

city.text=ob!.city!;

state.text=ob!.state!;

contry.text=ob!.contry!;

phone.text=ob!.phone!;

}

Future getImager(State\_Provider provider) async{

final picked\_image = await pick\_image.pickImage(source: ImageSource.gallery);

if(picked\_image!=null){

ob!.college\_Image = File(picked\_image.path);

provider.change\_State();

}

}

Widget build(BuildContext context){

var size = MediaQuery.of(context);

return Scaffold(

appBar:AppBar(

title:Text("College ERP"),

backgroundColor:Color(0xffF9E795),

toolbarHeight:size.size.height\*0.12,

),

body:ListView(

children: [

Row(

children: [

SizedBox(

child: InkWell(

child: Card(

color:Color(0xff99ff99),

elevation: 12,

child: Column(

children: [

Image.asset('assets/image/—Pngtree—cartoon school season happy pupils\_4575715.png',height:size.size.height\*0.18),

Text("Student",style: TextStyle(fontSize: size.size.height\*0.024,color:Color(0xffff9900)),),

Text("Registration",style: TextStyle(fontSize: size.size.height\*0.024,color:Color(0xff9966ff)),),

],

),

),

onTap: (){

Navigator.push(context,MaterialPageRoute(builder: (context)=>StudentPage(ob)));

},

),

height: size.size.height\*0.3,

width: size.size.width\*0.4,

),

SizedBox(

child: InkWell(

child: Card(

color:Color(0xff99ffcc),

elevation: 12,

child: Column(

children: [

Image.asset('assets/image/1725075965714.png',height:size.size.height\*0.18),

Text("Faculty",style: TextStyle(fontSize: size.size.height\*0.024,color:Color(0xffff5050)),),

Text("Registration",style: TextStyle(fontSize: size.size.height\*0.024,color:Color(0xff9966ff)),),

],

),

),

onTap: (){

Navigator.push(context, MaterialPageRoute(builder: (context)=>Faculty\_Page(ob)));

},

),

height: size.size.height\*0.3,

width: size.size.width\*0.4,

),

],

mainAxisAlignment: MainAxisAlignment.spaceEvenly,

),

SizedBox(

height: size.size.height\*0.13,

child: InkWell(

child: Card(

color: Color(0xffccffcc),

child: Row(

children: [

Image.asset('assets/image/1725115887833.png',height: size.size.height\*0.1,),

Text("BCA Students Marks",style: TextStyle(fontSize: size.size.width\*0.05,color: Color(0xff009933)),),

],

),

),

onTap: (){

Navigator.push(context, MaterialPageRoute(builder: (context)=>BcaMarks(ob)));

},

),

),

SizedBox(

height: size.size.height\*0.13,

child: InkWell(

child: Card(

color: Color(0xffccffcc),

child: Row(

children: [

Image.asset('assets/image/1725354545621.png',height: size.size.height\*0.1,),

Text("BBA Students Marks",style: TextStyle(fontSize: size.size.width\*0.05,color: Color(0xff009933)),),

],

),

),

onTap: (){

Navigator.push(context, MaterialPageRoute(builder: (context)=>BbaMarks(ob)));

},

),

),

SizedBox(

height: size.size.height\*0.13,

child: InkWell(

child: Card(

color: Color(0xffccffcc),

child: Row(

children: [

Padding(

padding: EdgeInsets.fromLTRB(size.size.width\*0.06, 0, 0, 0),

child: Image.asset('assets/image/—Pngtree—business finance salary\_5424419.png',height: size.size.height\*0.1,),

),

Text(" Enter Faculty Salary",style: TextStyle(fontSize: size.size.width\*0.05,color: Color(0xff009933)),),

],

),

),

onTap: (){

Navigator.push(context, MaterialPageRoute(builder: (context)=>Faculty\_Salary(ob)));

},

),

),

SizedBox(

height: size.size.height\*0.13,

child: InkWell(

child: Card(

color: Color(0xffccffcc),

child: Row(

children: [

Image.asset('assets/image/1725354521424.png',height: size.size.height\*0.1,),

Text(" Read Faculty Salary",style: TextStyle(fontSize: size.size.width\*0.05,color: Color(0xff009933)),),

],

),

),

onTap: (){

Navigator.push(context, MaterialPageRoute(builder: (context)=>Faculty\_Salary\_Read(ob)));

},

),

),

],

),

drawer:Drawer(

child: ListView(

children: [

DrawerHeader(

child: Column(

children: [

ChangeNotifierProvider<State\_Provider>(

create:(context)=>State\_Provider(),

child:Consumer<State\_Provider>(

builder: (context,provider,child){

return Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height\*0.01, size.size.height\*0.2, 0),

child: InkWell(

child: CircleAvatar(

radius: size.size.width\*0.08,

backgroundImage:ob!.college\_Image!=null?FileImage(ob!.college\_Image!):NetworkImage(''),

),

onTap: (){

getImager(provider);

},

),

);

},

),

),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.width\*0.02, size.size.width\*0.2, 0),

child: Text("College Name :- ${ob!.college\_Name}",overflow: TextOverflow.ellipsis,),

),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.width\*0.02, size.size.width\*0.2, 0,),

child: Text("College Email :- ${ob!.email}",overflow: TextOverflow.ellipsis,),

),

],

),

),

Center(

child: Padding(

padding: EdgeInsets.fromLTRB(0, size.size.width\*0.02,0, 0,),

child: InkWell(

child: Text("Account Details",style: TextStyle(fontSize: size.size.width\*0.05,color: Colors.black),),

onTap: (){

Navigator.push(context, MaterialPageRoute(builder: (context)=>college\_Detail(context)));

},

),

),

),

Center(

child: Padding(

padding: EdgeInsets.fromLTRB(0, size.size.width\*0.02,0, 0,),

child: InkWell(

child: Text("Edit Account Details",style: TextStyle(fontSize: size.size.width\*0.05,color: Colors.black),),

onTap: (){

Navigator.push(context, MaterialPageRoute(builder: (context)=>edit\_College\_Detail(context)));

},

),

),

),

Divider(),

Center(

child: Padding(

padding: EdgeInsets.fromLTRB(0, size.size.width\*0.02,0, 0,),

child: InkWell(

child: Text("LogOut",style: TextStyle(fontSize: size.size.width\*0.05,color: Colors.black,fontWeight: FontWeight.bold),),

onTap: (){

Navigator.pushReplacement(context, MaterialPageRoute(builder: (context)=>MyApp()));

},

),

),

),

],

),

),

);

}

Widget college\_Detail(BuildContext context){

var size = MediaQuery.of(context);

return Scaffold(

body: SafeArea(

child: ListView(

children: [

Column(

children: [

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height\*0.08, 0, 0),

child: CircleAvatar(

backgroundImage: ob!.college\_Image != null ? FileImage(ob!.college\_Image!) : NetworkImage(""),

radius: size.size.height\*0.08,

),

),

Divider(height: 0,),

Container(

color: Colors.white,

height: size.size.height\*0.1,

width: size.size.width\*1,

child: Row(

children: [

Text(" College Name :- ",style: TextStyle(fontSize: size.size.height\*0.02,fontWeight: FontWeight.bold),),

Text(ob!.college\_Name!,style: TextStyle(fontSize: size.size.height\*0.02),softWrap: true,),

],

),

),

Divider(height: 0,),

Container(

color: Colors.white,

height: size.size.height\*0.1,

width: size.size.width\*1,

child: Row(

children: [

Text(" College Address :- ",style: TextStyle(fontSize: size.size.height\*0.02,fontWeight: FontWeight.bold),),

Text(ob!.address!,style: TextStyle(fontSize: size.size.height\*0.02),softWrap: true,),

],

),

),

Divider(height: 0,),

Container(

color: Colors.white,

height: size.size.height\*0.1,

width: size.size.width\*1,

child: Row(

children: [

Text(" City :- ",style: TextStyle(fontSize: size.size.height\*0.02,fontWeight: FontWeight.bold),),

Text(ob!.city!,style: TextStyle(fontSize: size.size.height\*0.02),softWrap: true,),

],

),

),

Divider(height: 0,),

Container(

color: Colors.white,

height: size.size.height\*0.1,

width: size.size.width\*1,

child: Row(

children: [

Text(" State :- ",style: TextStyle(fontSize: size.size.height\*0.02,fontWeight: FontWeight.bold),),

Text(ob!.state!,style: TextStyle(fontSize: size.size.height\*0.02),softWrap: true,),

],

),

),

Divider(height: 0,),

Container(

color: Colors.white,

height: size.size.height\*0.1,

width: size.size.width\*1,

child: Row(

children: [

Text(" Contry :- ",style: TextStyle(fontSize: size.size.height\*0.02,fontWeight: FontWeight.bold),),

Text(ob!.contry!,style: TextStyle(fontSize: size.size.height\*0.02),softWrap: true,),

],

),

),

Divider(height: 0,),

Container(

color: Colors.white,

height: size.size.height\*0.1,

width: size.size.width\*1,

child: Row(

children: [

Text(" Phone No. :- ",style: TextStyle(fontSize: size.size.height\*0.02,fontWeight: FontWeight.bold),),

Text(ob!.phone!,style: TextStyle(fontSize: size.size.height\*0.02),softWrap: true,),

],

),

),

Divider(height: 0,),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height\*0.05, 0, 0),

child: SizedBox(

height: size.size.height\*0.06,

width: size.size.width\*0.3,

child: ElevatedButton(

onPressed: (){

Navigator.pop(context);

},

style: ElevatedButton.styleFrom(

backgroundColor: Colors.teal,

),

child: Row(

children: [

Icon(Icons.arrow\_back,color: Colors.white,),

Text("Back",style: TextStyle(color: Colors.white),),

],

),

),

),

),

],

),

],

),

),

backgroundColor: Color(0xffe6ffff),

);

}

Widget edit\_College\_Detail(BuildContext context){

var size = MediaQuery.of(context);

return Scaffold(

appBar: AppBar(

title: Text("Edit College Detail's"),

backgroundColor: Color(0xffffff99),

toolbarHeight: size.size.height\*0.1,

actions: [

Padding(

padding: EdgeInsets.all(size.size.width\*0.05),

child: Text("BCA",style: TextStyle(fontSize: size.size.width\*0.06),),

),

],

),

body: ListView(

children: [

Column(

children: [

ChangeNotifierProvider<State\_Provider>(

create:(context)=>State\_Provider(),

child:Consumer<State\_Provider>(

builder: (context,provider,child){

return Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height\*0.01, 0, 0),

child: InkWell(

child: CircleAvatar(

radius: size.size.width\*0.15,

backgroundImage:ob!.college\_Image!=null?FileImage(ob!.college\_Image!):NetworkImage(''),

),

onTap: (){

getImager(provider);

},

),

);

},

),

),

Text("College Image"),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.04, 0, 0),

child: TextField(

controller:college\_Name,

decoration:InputDecoration(

label:Text("Collage Name"),

labelStyle:TextStyle(fontSize:18,color:Colors.black),

filled:true,

fillColor:Color(0xfff2e6ff),

border:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12))

),

focusedBorder:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12)),

borderSide:BorderSide(

color:Colors.purple,

width:2,

),

),

),

),

),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

child: TextField(

controller:address,

decoration:InputDecoration(

label:Text("Address"),

labelStyle:TextStyle(fontSize:18,color:Colors.black),

filled:true,

fillColor:Color(0xfff2e6ff),

border:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12))

),

focusedBorder:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12)),

borderSide:BorderSide(

color:Colors.purple,

width:2,

),

),

),

),

),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

child: TextField(

controller:city,

decoration:InputDecoration(

label:Text("City"),

labelStyle:TextStyle(fontSize:18,color:Colors.black),

filled:true,

fillColor:Color(0xfff2e6ff),

border:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12))

),

focusedBorder:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12)),

borderSide:BorderSide(

color:Colors.purple,

width:2,

),

),

),

),

),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

child: TextField(

controller:state,

decoration:InputDecoration(

label:Text("State"),

labelStyle:TextStyle(fontSize:18,color:Colors.black),

filled:true,

fillColor:Color(0xfff2e6ff),

border:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12))

),

focusedBorder:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12)),

borderSide:BorderSide(

color:Colors.purple,

width:2,

),

),

),

),

),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

child: TextField(

controller:contry,

decoration:InputDecoration(

label:Text("Contry"),

labelStyle:TextStyle(fontSize:18,color:Colors.black),

filled:true,

fillColor:Color(0xfff2e6ff),

border:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12))

),

focusedBorder:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12)),

borderSide:BorderSide(

color:Colors.purple,

width:2,

),

),

),

),

),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

child: TextField(

controller:phone,

decoration:InputDecoration(

label:Text("Phone No."),

labelStyle:TextStyle(fontSize:18,color:Colors.black),

filled:true,

fillColor:Color(0xfff2e6ff),

border:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12))

),

focusedBorder:OutlineInputBorder(

borderRadius:BorderRadius.all(Radius.circular(12)),

borderSide:BorderSide(

color:Colors.purple,

width:2,

),

),

),

),

),

Padding(

padding: EdgeInsets.fromLTRB(0, size.size.height \* 0.02, 0, 0),

child: Row(

children: [

SizedBox(

height:size.size.height\*0.06,

width:size.size.width\*0.3,

child: ElevatedButton(

onPressed:(){

Navigator.pop(context);

},

child:Text("Cancel",style:TextStyle(color:Colors.white,fontSize:18,fontWeight:FontWeight.bold)),

style:ElevatedButton.styleFrom(

shape:RoundedRectangleBorder(

borderRadius:BorderRadius.all(Radius.circular(12)),

),

backgroundColor:Colors.grey,

),

),

),

SizedBox(

height:size.size.height\*0.06,

width:size.size.width\*0.3,

child: ElevatedButton(

onPressed:(){

String cm = college\_Name.text.trim().toString();

String ad = address.text.trim().toString();

String cy = city.text.trim().toString();

String st = state.text.trim().toString();

String co = contry.text.trim().toString();

String ph = phone.text.trim().toString();

if(cm=="" || ad=="" || cy=="" || st=="" || co=="" || ph==""){

return;

}

ob!.input\_Data(cm,ad,cy,st,co,ph);

setState(() {

});

Navigator.pop(context);

},

child:Text("Edit",style:TextStyle(color:Colors.white,fontSize:18,fontWeight:FontWeight.bold)),

style:ElevatedButton.styleFrom(

shape:RoundedRectangleBorder(

borderRadius:BorderRadius.all(Radius.circular(12)),

),

backgroundColor:Colors.blue,

),

),

),

],

mainAxisAlignment: MainAxisAlignment.spaceEvenly,

)

),

],

)

],

),

);

}

}

# Chapter-5 Conclusions

The College ERP System has been developed using the Flutter framework with Dart programming for better functionality in college administration. For backend data storage, SQLite has been used, enabling offline access and management of information without requiring an internet connection.

The application provides essential functionalities for administrators, faculty, and students. Administrators can add new faculty members, update faculty details, and manage general college information. Faculty members can add new students, record attendance, and enter marks, all of which are stored locally for efficient offline access.

Students benefit from easy access to their personal details, attendance records, and academic marks through a secure login. This offline ERP system simplifies campus operations by offering a user-friendly, reliable, and accessible platform, ensuring that college data is available anytime, even without internet connectivity.

# References

# -: GitHub Link :- <https://github.com/PrathamAwasthi/College_ERP>

# -: Apk Link :-https://github.com/PrathamAwasthi/College\_ERP/blob/main/app-release.apk

# 