I.P. COLLEGE, CAMPUS-II, BULANDSHAHR

स्थापना वर्ष : 1970

A

Project Report

Entitled

"College Management System"

As

Major Project

Submitted in the partial fulfillment the degree of Bachelor of Computer Application

Batch (2020–2023)

CH. CHARAN SINGH UNIVERSITY, MEERUT

Under The Distinguish Guidance of:

Mrs. Ginni Dua
Assistant professor
Dept. of Computer Science
I.P. College Campus-II, Bulandshahr

Submitted By:

Group No: - 21
Class:- BCA VIth Sem.

Name	Roll-No.
Yash Gupta	200955106196
Yash Gupta	200955106195
Sahil Gupta	200955106141
Navya	200955106101
Sakshi Kumari	200955106142

PROJECT CATEGORY

Web Application

ACKNOWLEDGEMENT

We take this opportunity to express our sincere thanks and deep gratitude to all those people who extended their whole hearted co-operation and have helped us in completing this project successfully.

We acknowledge the effort of those who have contributed significantly to our project. We express our sincere attitude and thank-fullness towards **Dr. T.N. MISHRA** (Principal of I.P. (P.G.) College Capmus-2 NH-91, Delhi Road, Bulandshahr) has provided such a talented faculty to us then we would like to thank **Mr**. **SANJAY KUMAR** (Head of Department) for their guidance throughout BCA.

We sincerely thank to our guide **"Mrs. Ginni Dua"** for her inspiring guidance and constant motivation. Every time we were lurking in the dark, she showed us the way out.

We should like to express our gratitude to other faculty members of our college for their reviews andmany helpful comments.

Finally, the project would not have been possible without confidence, endurance and support of group members.

Name of Group Member

Yash Gupta	[200955106196]
Yash Gupta	[200955106195]
Sahil Gupta	[200955106141]
Navya	[200955106101]
Sakshi Kumari	[200955106142]

STUDENT DECLARATION

We solemnly affirm and declare that the project **"College Management System"** under the guidance of "**Mrs. Ginni Dua"** our original work. No part of this work whether documentation or coding has not been copied or taken in any form and by any means.

We certify that our project work is original and it has been made as major project held in BCA VIth semester for the partial fulfillment the degree of "Bachelor of Computer Application"

Group members name with Roll no.		**Signature**
Yash Gupta	[200955106196]	
Yash Gupta	[200955106195]	
Sahil Gupta	[200955106141]	
Navya	[200955106101]	
Sakshi Kumari	[200955106142]	

CERTIFICATE FROM THE SUPERVISOR

This is to certify that the project report entitled, **“College Management System”** is a beneficed work done by:

1. Yash Gupta [200955106196]
2. Yash Gupta [200955106195]
3. Sahil Gupta [200955106141]
4. Navya [200955106101]
5. Sakshi Kumari [200955106142]

Student of BCA III Year Batch (2020-2023) as major project report carried out under my supervision and guidance.

To the best of my knowledge this project work is genuine.

…………………………

Signature of Guide

Mrs. Ginni Dua

Assistant professor

Dept. of Computer Science

I.P. College, Campus-II, Bulandshahr

Date: …………….

TABLE OF CONTENT

Phase I

Phase II

1. ABOUT THE PROJECT

I. INTRODUCTION

Management system is an important and essential part of any educational institute. The project defines all the arrangement of college system. College management system means it controls all management tasks and functions. It performs all the function which require to any educational institute such as information about each and every student, faculty & even staff members and information about any event of the college or performance records of each student etc.

Admin can view, insert or change all record. Faculty can also view their information and have been authorized to update results, attendance, performance on daily bases in their subject or class portals. Students can search for their records or needed information like result, time table, notice, date sheet etc. They can see more information about the college on the college management system software.

Every information about college will be provided there for students like rules and regulation, admission procedure, semester wise fee structure of courses, academic calendar, course evaluation forms at the end of each semester etc. Students can go through all those information and can give suggestions to make the program and educational environment better.

II. OBJECTIVES

- The main objective of this app is to save time and reduce paper work or paper usage as it is a digital platform.
- This app will help the management to control the educational system online more effectively.
- As everything is online in this app as resultant the coordinate between the staff will be strong.
- This app will help financially to the management as digital work save time and reduce staff.
- The work of hours can be done in minutes.
- We can add, update, search and delete data about faculty, students and other staff members of college.
- We can find out about any of the college member at only one place as it contains student's, faculty's and other staff's personal information.
- It contains student's as well as faculty's and other staff's attendance.
- Salary information will be easy to manage as it contains the data of faculty's and other staff's salary.
- Students can see their all fee package, admission details, the details of courses completed and pending courses or repeating courses.
- Students can fill their course evaluation form at the end of each semester.
- Students will be aware about their daily bases performance.
- The students can see all notices related to any event online so they don't need to rush at offices or in front of the staff rooms.

2. PRELIMINARY INVESTIGATION

I. TOOLS/PLATFORM USED

- **VS Code:-** Visual Studio Code is a source-code editor that can be used with a variety of programming languages. It is based on the Electron framework, which is used to develop Node.js web applications that run on the Blink layout engine. A notable feature is the ability to create extensions that add support for new languages, themes, debuggers, time travel debuggers, perform static code analysis, and add code linters using the Language Server Protocol.
  In our project vs code is used as the framework for dart language for the code editor which allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language.

- **Android Studio:-** Android Studio is the official integrated development environment (IDE) for Android app development. It is built on top of the IntelliJ IDEA platform and offers a comprehensive set of tools for developing Android apps. Some of the key features of Android Studio include: Project setup, Code editor, Layout editor, Emulator, Gradle build system, Integration with Firebase.
  We used android studio mainly for code editor to supports code completion, syntax highlighting, and refactoring and Android Virtual Device (Emulator) to run and debug app.

- **Firebase Console:-** Firebase Console is a web-based user interface provided by Google Firebase, which is a platform for developing mobile and web applications. The Firebase Console allows developers to manage their Firebase projects, configure settings, view analytics, set up notifications, and access other Firebase services. Some of the services and features available in the Firebase Console include: Authentication, Real-time Database, Cloud Firestore, Storage, Hosting, Functions, Analytics, Performance Monitoring, Remote Configuration.
  We used firebase console for backend to manage authentication, store and sync data and serve user-generated content, host static assets, and monitor app’s performance.

II. HARDWARE/SOFTWARE REQUIREMENT

➢ **Hardware Requirement: -**

- Android running OS 4.1 and newer
- iPhone running iOS and newer

➢ **Software Requirement: -**

- Microsoft Visual Studio
- Android studio
- Dart and Flutter
- Firebase Console

3. FEASIBILITY STUDY

I. OPERATION FEASIBILITY

In operational feasibility we will have to satisfy the requirements of customer who want to develop software. In other words, the developer has to give all the facilities and complete all the requirements given by user. The operation feasibility of a college management system depends on various factors such as the system's scope, requirements, complexity, and available resources.
In our program, there all facilities are present which have to be in a well college management software:-

1. Our app is practical to use and it meet the needs of the college like daily bases records, updates news about college events etc.
2. This is also user-friendly as it contains all the information needed by the user.
3. It has an easy to use interface so the user can operate it easily without any problem.
4. This is reliable as our system or process produce accurate and consistent results over time. This program can perform its intended functions consistently and without failure.
5. This is efficient as this system or process perform its intended functions in the most efficient way possible. In other words, it can accomplish its goals in the most optimal manner.
6. Our program is time saving as it achieve its objectives with minimal waste of time, effort, or resources.
7. The software is also nature loving because it can save the paper work, because it store information digitally and not on any paper.

II. TECHNICAL FEASIBILITY

In technical feasibility, whether we have required technical resources (hardware, software) to develop the project. Now, we used various well-defined softwares in our project. As a developer we know that good software helps for making a better project so we used popular and trustable software in our project like –

1. Android Studio
2. VS Code
3. Firebase Console

The members make the project after research on the language and we studied that language on which the project is based. The group members who make the project know everything and have good knowledge about the languages used in the project. The project is made up in a good way and we make it after so many researches and after gaining the knowledge about the language. So, our members who make the project have skills to make the project and have deep knowledge about the project.

Technically the project is fully modified in a good way to all the hardware and software which are used to make the project are the latest and the modified versions. In our project if the users wantany type of upgrade or change in the software so, ourdevelopers are also giving the option to update and add something new in the future. In other words, the software can be updated in the future according to the user requirements.

Our project has the facility to develop and maintain the software in the future and the software is made up in a very simple and it is also very easy to understand the software and the user can use the software/application easily in a very good way. It has various features which helps user to implement the software.

III. ECONOMICAL FEASIBILITY

The economic feasibility of a college management system depends on several factors such as the size of the institution, the scope of the system, and the budget available for implementation and maintenance.

In general, college management systems can improve the efficiency of administrative tasks, reduce paperwork, and provide valuable data for decision-making. These benefits can lead to cost savings in the long run, as well as increased productivity and effectiveness.

However, the initial investment in a college management system can be significant, as it may involve software licensing, hardware upgrades, and personnel training. Additionally, ongoing maintenance and updates may be required to ensure the system remains effective and up-to-date.

In order to determine the economic feasibility of a college management system, a cost-benefit analysis should be conducted. This analysis would consider the costs of implementation and maintenance, as well as the potential benefits such as increased efficiency, improved data management, and reduced workload for staff members. The analysis should also consider the return on investment over the long term, as well as any potential risks or drawbacks of implementing the system.

Overall, while the upfront costs of implementing a college management system can be significant, the potential benefits and cost savings over time may make it a worthwhile investment for many institutions.

Our software has many functions that give us features and facility to make the work easy in a very low cost. These are the various benefits of the software -

1. Drastic reduction in manual tasks.
2. Elimination of paper-based tasks.
3. Improved fee management.
4. Make the institute smart and ready for the future.
5. Improved institute reputation.
6. Better resource utilization.
7. Always ready information.

These are the various main benefits of the college management system in a very good way.

4. INFORMATION GATHERING

I. ON-SITE OBSERVATION

On site observation refers to the visit of system analyst in the organization in order to observe the working of the current system. Our project is totally based on the ground level system. The software is making after getting the deep knowledge about the project. In on-site observation, it checked the software at the low level and at the deep level of the software. For on-site observation we focused on the following areas:

1. Enrollment and registration processes: We observe how the system handles the enrollment and registration processes, including the collection and management of student information, course selection, and payment processing.
2. Academic management: We observe how the system handles academic-related tasks such as scheduling, attendance tracking, grading, and transcript management.
3. Financial management: We observe how the system handles financial-related tasks such as billing, payment processing, and budget management.
4. Human resources management: We observe how the system handles tasks related to human resources such as employee records management, payroll processing, and benefits administration.
5. Communication and collaboration: We observe how the system facilitates communication and collaboration among students, faculty, and staff, including the use of email, messaging, and other collaboration tools.
6. Security and privacy: We observe how the system handles security and privacy concerns, including user authentication, data encryption, and access controls.
7. User interface and usability: We observe the overall user interface and usability of the system, including ease of use, responsiveness, and user-friendliness.

By observing these areas, you can gain insight into the strengths and weaknesses of the college management system and identify areas for improvement.

II. QUESTIONNAIRE

A questionnaire is a research tool used to gather information or data from individuals or groups. It typically consists of a series of questions that respondents answer either in written or electronic form. Questionnaires can be used for various purposes, such as conducting surveys, collecting feedback, assessing customer satisfaction, or conducting research studies.

Questionnaires can be designed to collect quantitative data (numerical data) or qualitative data (descriptive or subjective data). The questions in a questionnaire can be open-ended (allowing respondents to provide detailed responses) or closed-ended (providing a set of predefined response options).

The design and structure of a questionnaire depend on the specific objectives and target audience of the research. Careful consideration should be given to the clarity and wording of the questions to ensure that respondents understand them and can provide accurate and meaningful answers.

Once the questionnaire is completed, the data collected can be analyzed to gain insights, identify patterns, or draw conclusions based on the research objectives.

Here is our performed questionnaire:-

1. Personal Information:
   a. Name:
   b. Student ID:
   c. Department:
   d. Year of Enrollment:
   e. Contact Number:
   f. Email Address:

2. User Experience:
   a. How satisfied are you with the college management system?
      - Very satisfied
      - Satisfied
      - Neutral
      - Dissatisfied
      - Very dissatisfied

   b. How often do you use the college management system?
      - Daily
      - Several times a week
      - Once a week
      - Rarely
      - Never

   c. What features do you find most useful in the college management system?
      - Course registration
      - Timetable management
      - Attendance tracking
      - Exam results
      - Fee payment
      - Library services
      - Other (please specify):

   d. Are there any features that you find difficult to use or navigate? If yes, please specify:

3. Communication and Notifications:
   a. How effective are the system's communication and notification features?
      - Very effective
      - Effective
      - Neutral
      - Ineffective
      - Very ineffective

   b. Are you receiving important notifications in a timely manner?
      - Yes, always
      - Yes, most of the time

- o Sometimes
- o Rarely
- o Never

c. Which communication channels do you prefer for receiving system notifications?

- o Email
- o SMS
- o Push notifications in the mobile app
- o In-app notifications
- o Other (please specify):

4. System Performance:

a. How would you rate the system's overall performance (speed, responsiveness, etc.)?

- o Excellent
- o Good
- o Average
- o Poor
- o Very poor

b. Have you experienced any system downtime or performance issues? If yes, please describe the problem.

6. Course Registration:

a. How satisfied are you with the course registration process in the college management system?

- o Very satisfied
- o Satisfied
- o Neutral
- o Dissatisfied
- o Very dissatisfied

b. Have you encountered any difficulties or issues during the course registration process? If yes, please describe:

7. Timetable Management:

a. How useful is the timetable management feature in the college management system?

- o Very useful
- o Useful
- o Neutral
- o Not very useful
- o Not useful at all

b. Are there any improvements you would like to see in the timetable management functionality?

8. Attendance Tracking:

a. How effectively does the system track your attendance?

- o Very effectively
- o Effectively
- o Neutral
- o Ineffectively

- Very ineffectively

b. Are there any challenges or limitations you have encountered while using the attendance tracking feature?

9. Exam Results:
a. How satisfied are you with the accuracy and timeliness of exam result publication in the college management system?
- Very satisfied
- Satisfied
- Neutral
- Dissatisfied
- Very dissatisfied

b. Is there any additional information or features related to exam results that you would like to see in the system?

10. Fee Payment:
a. How convenient and user-friendly is the fee payment process in the college management system?
- Very convenient and user-friendly
- Convenient and user-friendly
- Neutral
- Inconvenient and not user-friendly
- Very inconvenient and not user-friendly b. Have you faced any issues or encountered any difficulties while making fee payments through the system?

11. Library Services:
a. How satisfied are you with the library services provided through the college management system?
- Very satisfied
- Satisfied
- Neutral
- Dissatisfied
- Very dissatisfied

b. Are there any improvements or additional features you would like to see in the library services offered by the system?

12. Overall Satisfaction:
a. On a scale of 1 to 10, how likely are you to recommend the college management system to other students?
b. Is there anything else you would like to mention regarding your overall satisfaction with the college management system?

13. Suggestions for Improvement:
a. Are there any additional features or functionalities you would like to see in the college management system?
b. How can the system's user interface be improved to enhance the user experience?
c. Any other suggestions or feedback you would like to provide?

Thank you for your participation. Your feedback will contribute to enhancing the college management system for all users.

III. INTERVIEWING

An interview is a conversational interaction between two or more people, typically conducted for the purpose of gathering information, conducting research, or assessing a candidate's suitability for a job or position. It is a method of obtaining firsthand information through direct communication.

In the context of research or data collection, an interview involves a researcher or interviewer asking a series of questions to an individual or a group of participants. The questions can be open-ended, allowing for detailed responses and exploration of the topic, or they can be structured with predefined response options.

Interviews can be conducted in various formats, including face-to-face interviews, telephone interviews, video interviews, or online interviews. They can be structured, semi-structured, or unstructured, depending on the level of flexibility in the questioning and the desired depth of information.

Interviews are often used in qualitative research to gain insights into participants' experiences, perspectives, opinions, or behaviors. The interviewer guides the conversation, probes for more detailed information, and captures the responses in order to analyze them later.

In the context of job interviews, an employer or hiring manager conducts interviews with candidates to assess their qualifications, skills, experience, and fit for a particular job role.

Overall, interviews serve as a valuable method for gathering information, conducting research, making assessments, or engaging in meaningful conversations.

These are our some interview questions:-

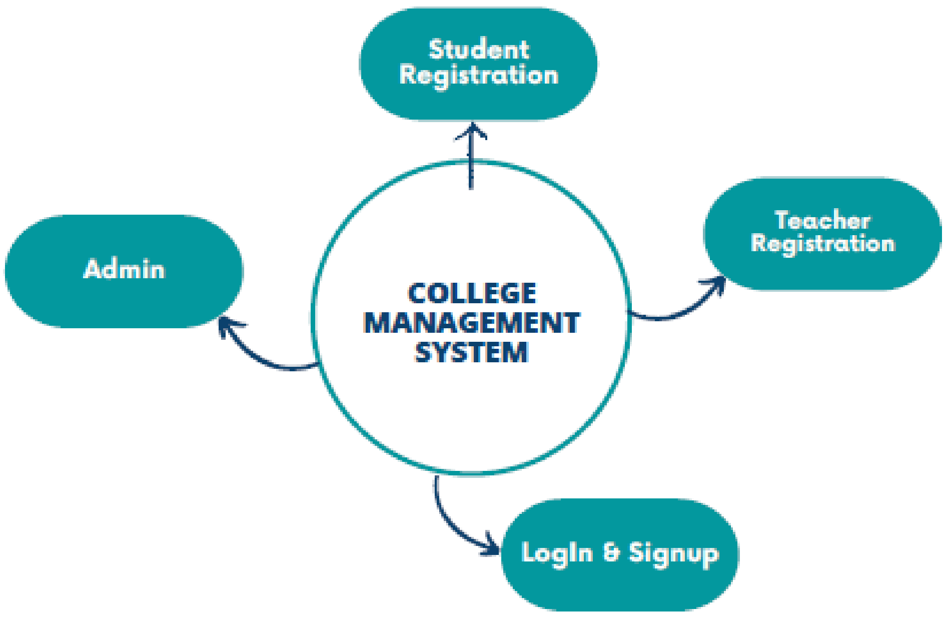
1. Can you briefly introduce yourself and your role at the college?
2. How long have you been using the college management system?
3. What are the key features of the college management system that you frequently use?
4. In your opinion, what are the strengths of the college management system?
5. Are there any areas of improvement or additional features you would like to see in the system?
6. How satisfied are you with the system's user interface and ease of navigation?
7. Have you encountered any challenges or difficulties while using the college management system? If so, could you provide some examples?
8. How effective are the system's communication and notification features in keeping you informed about important updates or announcements?
9. How well does the system handle course registration and timetable management?
10. What improvements would you suggest for the attendance tracking feature?
11. How satisfied are you with the accuracy and timeliness of exam result publication in the system?
12. Could you share your experience with the fee payment process in the college management system?

13. How useful are the library services provided through the system? Are there any areas that could be improved?
14. Overall, how satisfied are you with the college management system? Please rate on a scale of 1 to 10, with 1 being highly dissatisfied and 10 being highly satisfied.
15. Is there any additional feedback, suggestions, or comments you would like to provide regarding the college management system?
16. What are the considerations for scalability and future expansion when implementing a college management system?
17. How does a college management system assist in the management of faculty and staff, including payroll, leave management, and performance evaluation?
18. Can you provide examples of colleges or universities that have successfully implemented a management system and the benefits they have derived from it?
19. What is the implementation process for a college management system, and what are the key steps involved?
20. How does a college management system handle student records management and provide access to relevant stakeholders while ensuring data accuracy and privacy?
21. How does a college management system handle examination management, including scheduling, grading, and result processing?
22. What reporting and analytics capabilities should a college management system provide to assist in decision-making and performance monitoring?
23. How does a college management system handle financial management, including fee collection, budgeting, and expense tracking?
24. Can you discuss the integration possibilities of a college management system with other existing systems and tools used by the college?
25. How can a college management system support alumni management and engagement activities?
26. Can you describe the process of course scheduling and timetable management within a college management system?
27. What are the different modules or components that should be included in a comprehensive college management system?
28. How does a college management system handle student attendance tracking and management?
29. What are the security measures and data privacy considerations that should be taken into account while implementing a college management system?
30. Can you explain the role of a college management system in facilitating communication between faculty, students, and parents/guardians?

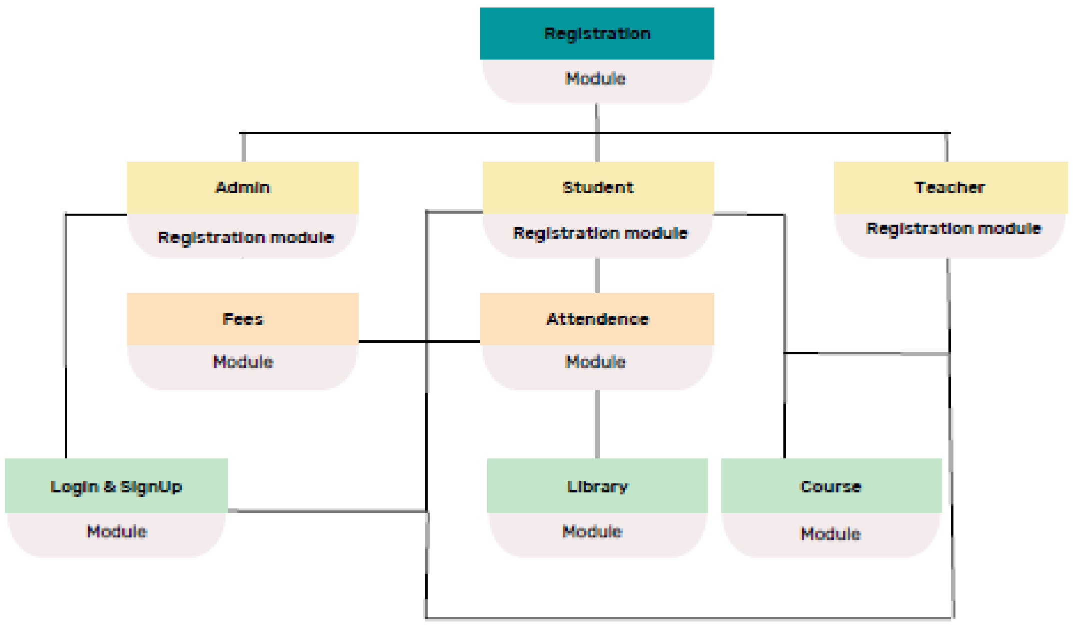
5. REQUIREMENT ANALYSIS & SPECIFICATIONS

I. DATAFLOW DIAGRAM (DFD)

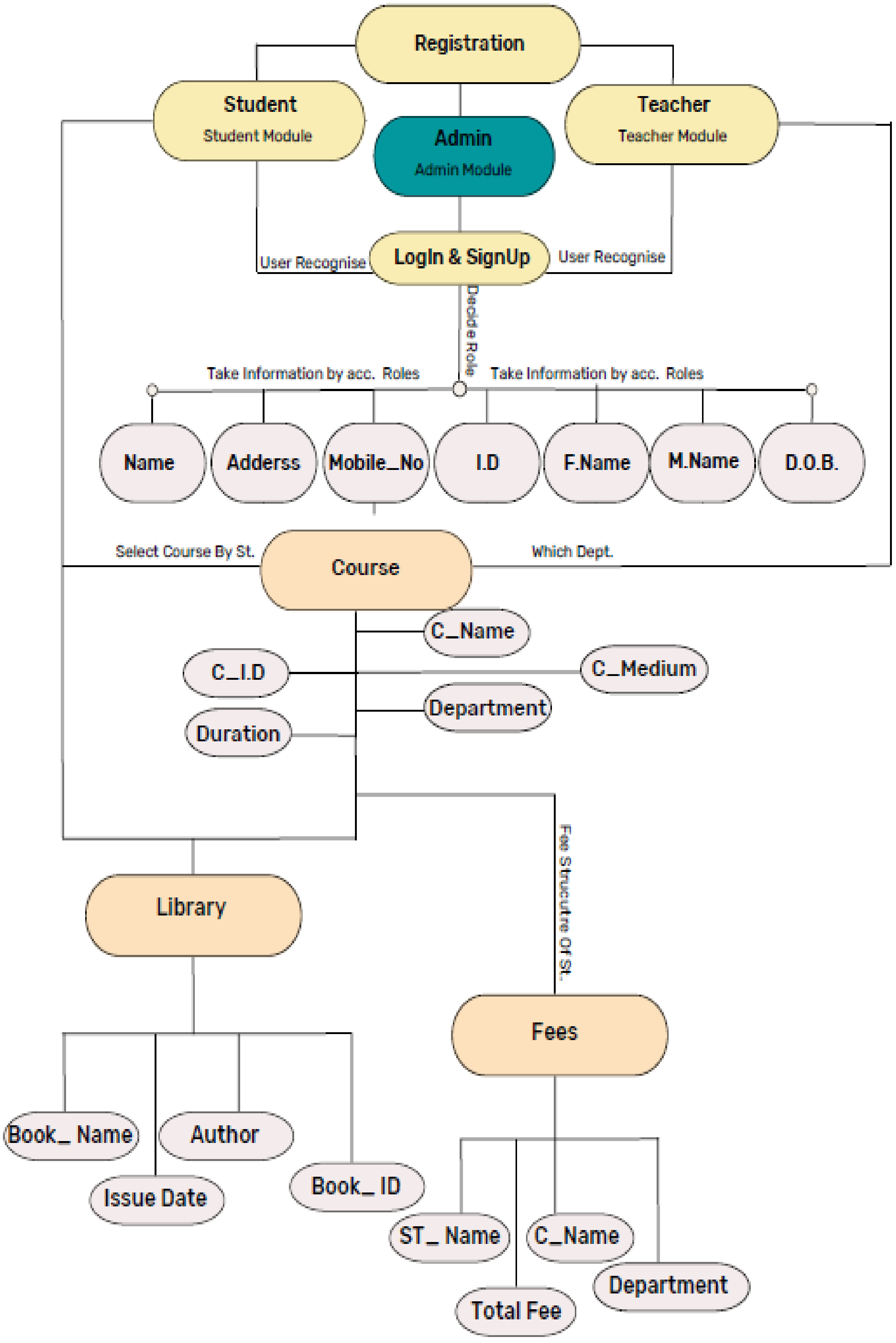
ZERO LEVEL DATA FLOW DIAGRAM



FIRST LEVEL DATA FLOW DIAGRAM

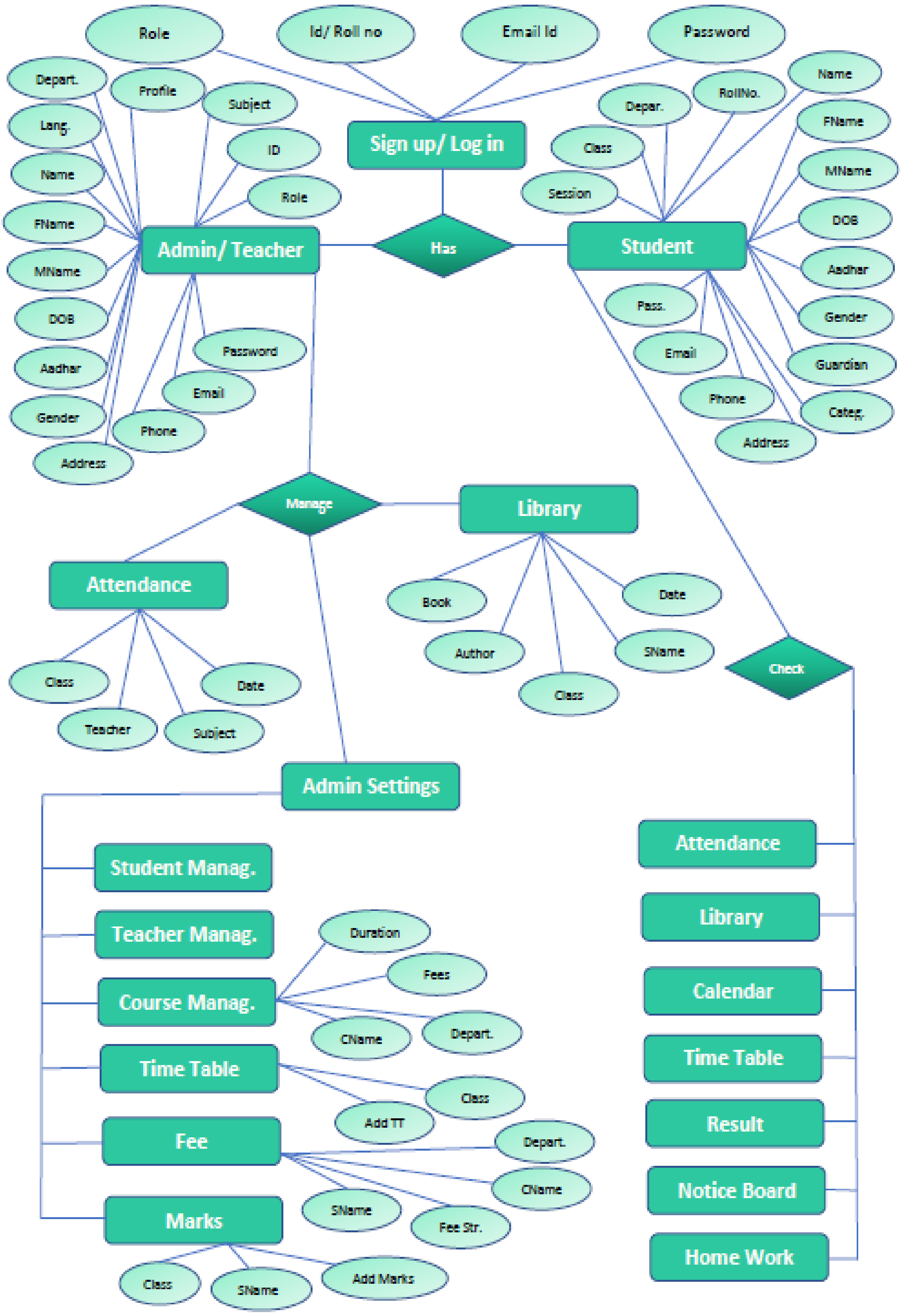


SECOND LEVEL DATA FLOW DIAGRAM

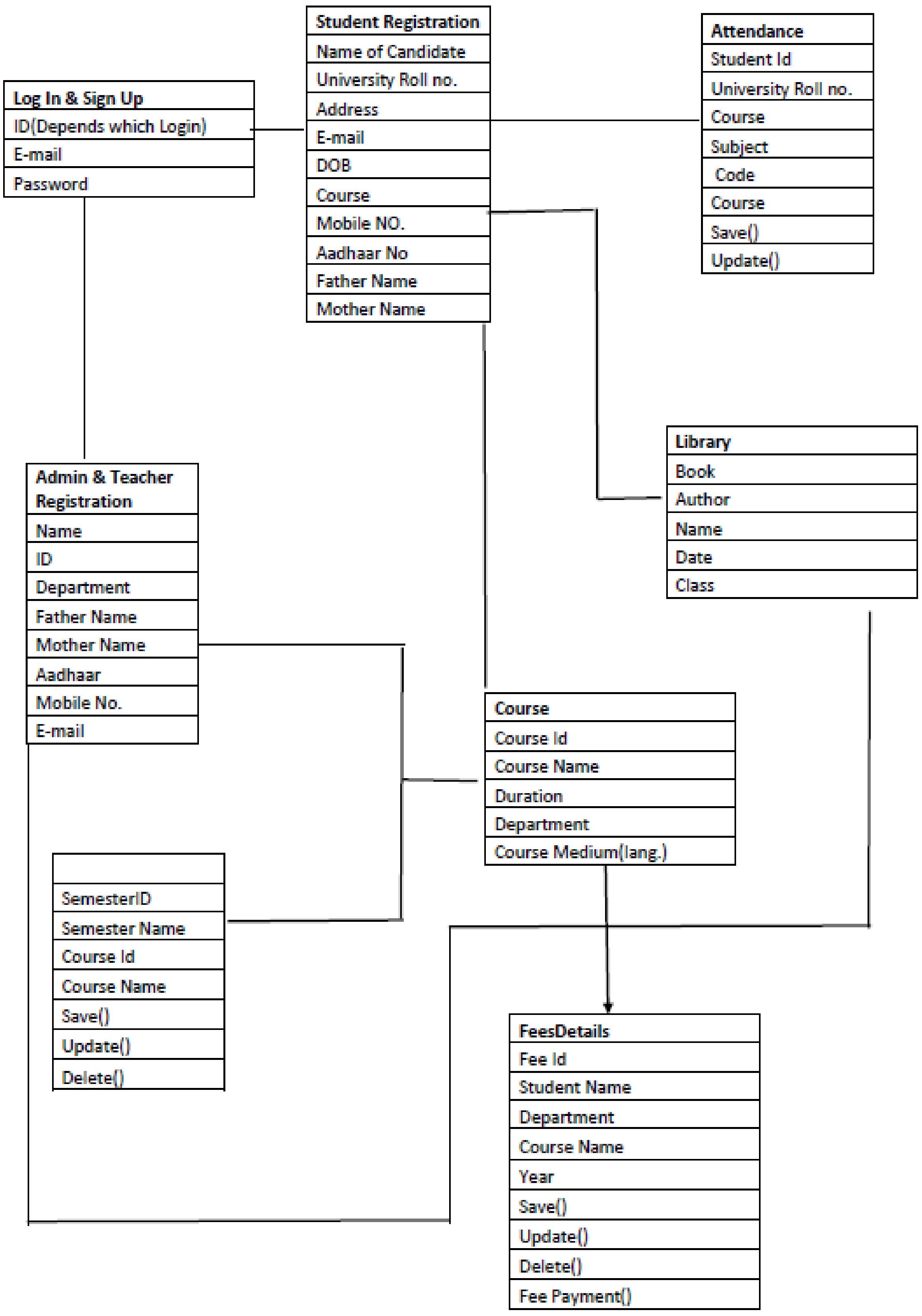


6. LOGICAL DESIGN

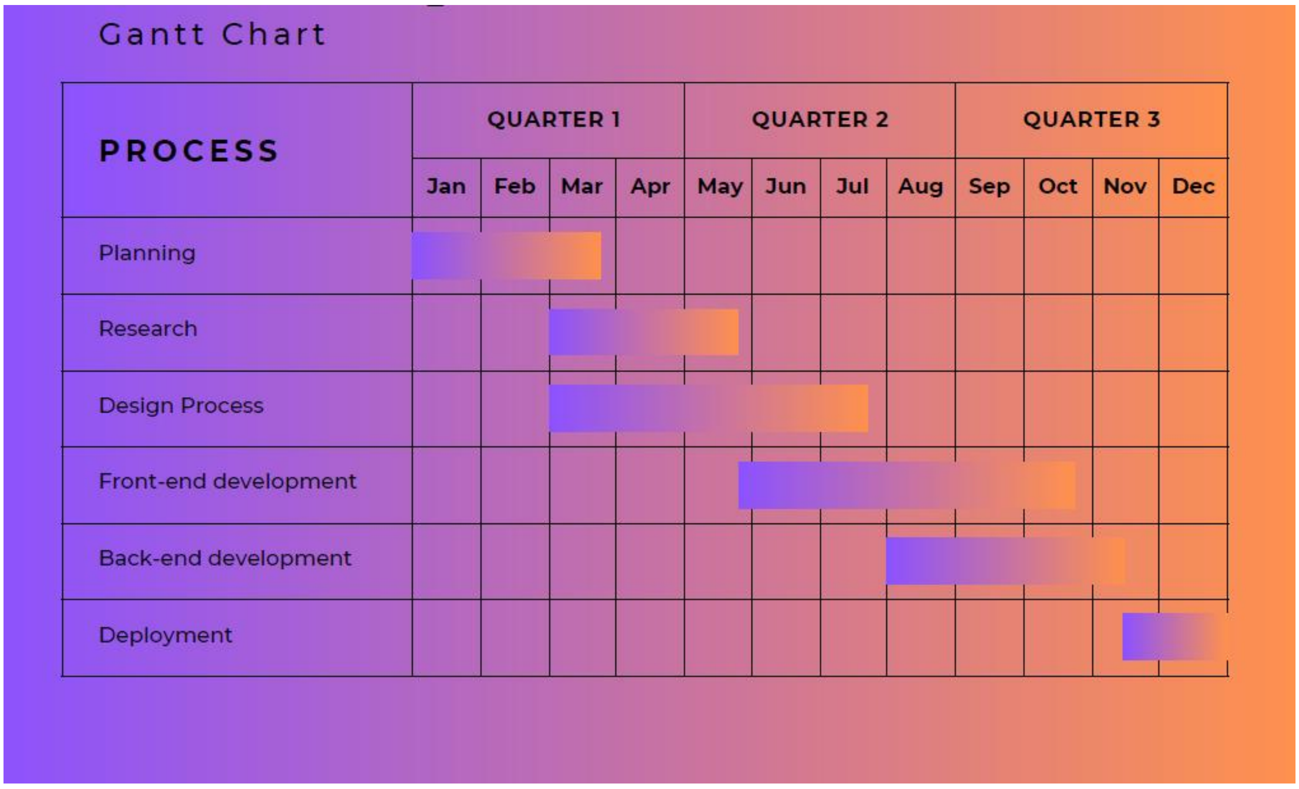
I. ENTITY RELATIONSHIP DIAGRAM (ERD)



II. DATABASE DESIGN



III. GANTT CHART



7. PHYSICAL DESIGN

Physical design, in the context of computer systems and integrated circuits, refers to the phase of the design process where the logical design of a system or chip is transformed into a physical representation that can be manufactured. It involves converting the abstract concepts and specifications into concrete layouts, placements, and connections of components, wires, and other physical elements.

I. OUTPUT REQUIREMENT

- Transport Detail
- Help
- Student & Faculty Information
- Class Scheduling
- Examination results
- Library Management etc

➢ User Interface design

User interface design focus on the user experience and interaction. The following characteristics are considered during the user interface design:

- Simplicity
- Dashboard
- Navigation
- Intuitive
- Consistent
- Efficient
- Error-free
- Functional
- Visual Design
- Feedback

➢ Output Report Design

- Report Overview: The College Management System is designed to streamline various administrative processes within a college or educational institution. This output report provides an overview of the system's key functionalities and data generated as a result of its operations. The report aims to present relevant information in a clear and concise manner to facilitate decision-making and enhance overall management efficiency.

1. Introduction: This section provides an introduction to the College Management System, including its purpose, scope, and the benefits it offers to the college administration, faculty, and students.

2. User Management:
  2.1. Registered Users:
  - Total number of registered users
  - Breakdown of user roles (e.g., administrators, faculty, students, staff)

- User profile information (name, email, contact details)

2.2. User Activity Logs:

- User login/logout activities
- User actions and system interactions
- Access logs for sensitive operations

3. <u>Student Management:</u>

3.1. Student Enrollment:

- Total number of enrolled students
- Student profiles (name, ID, contact details, program)
- Enrollment status (active, graduated, withdrawn)

3.2. Student Attendance:

- Attendance records for individual students
- Class attendance percentages
- Absence notifications and trends

3.3. Academic Performance:

- Grade records for individual students
- Course-wise performance analysis
- GPA calculations and trends

3.4. Student Reports:

- Individual student reports (enrollment, attendance, grades)
- Progress reports
- Student disciplinary records (if applicable)

4. <u>Faculty Management:</u>

4.1. Faculty Information:

- Faculty profiles (name, ID, contact details, qualifications)
- Teaching assignments and courses taught

4.2. Course Assignments:

- Faculty assigned to specific courses
- Course load distribution

4.3. Faculty Performance:

- Performance evaluation metrics
- Feedback from students and peers
- Professional development activities

4.4. Faculty Reports:

- Individual faculty reports (teaching assignments, performance)
- Faculty workload reports
- Research output and publications (if applicable)

5. <u>Course Management:</u>

5.1. Course Catalog:

- List of available courses
- Course descriptions and prerequisites

5.2. Course Enrollment:

- Enrollment statistics for each course
- Waitlist status for popular courses

5.3. Course Schedules:

- Timetables and class schedules
- Room allocations and availability

6. <u>Examination Management:</u>

6.1. Examination Schedules:

- Exam timetable
- Date, time, and location of each exam

6.2. Exam Results:

- Individual student exam results
- Class-wise performance analysis
- Grading statistics and trends

7. <u>Financial Management:</u>

7.1. Fee Collection:

- Fee payment records
- Pending fee balances
- Fee receipts and invoices

7.2. Financial Reports:

- Financial statements (income, expenses, balances)
- Budget allocations and expenditure tracking

8. <u>Library Management:</u>

8.1. Book Catalog:

- Available books and resources
- Book details (title, author, ISBN)

8.2. Book Issuance:

- Books issued to students and faculty
- Due dates and overdue books

8.3. Library Reports:

- Book circulation statistics
- Popular books and genres

Note: The report structure and content may vary based on the specific features and modules implemented in the College Management System.

II. <u>INPUT REQUIREMENTS</u>

- Login page get the input of user id and password.
- Create the new user id for your profile yourself.
- Compose mail.
- Attach the file with mail.
- View responses.
- Change password
- Search
- Chat

8. PROCESS DESIGN

I. CODING

- **Main.dart**

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:lottie/lottie.dart';
import 'package:yns_college_management/pages/College%20Web/aboutclg_page.dart';
import 'package:yns_college_management/pages/College%20Web/clgweb_page.dart';
import 'package:yns_college_management/pages/College%20Web/notification.dart';
import 'package:yns_college_management/pages/Splash%20Screen/splash_page.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:yns_college_management/Utils/routes.dart';
import 'package:yns_college_management/pages/try.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();

  // initialize app based on platform- web or mobile
  if (kIsWeb) {
    await Firebase.initializeApp(
        options: const FirebaseOptions(
            apiKey: 'AIzaSyB4qgtlot3GgmcyrO4YxoCzSLTMX3qaJIM',
            appId: '1:1022822098505:web:e6e12c5076c933fd708198',
            messagingSenderId: "1022822098505",
            projectId: "college-management-2bb43",
            storageBucket: "college-management-2bb43.appspot.com",
            authDomain: "college-management-2bb43.firebaseapp.com",
            measurementId: "G-ZB9BRK2FLQ"));
  } else {await Firebase.initializeApp();}
  runApp(const MyApp());
}

class MyApp extends StatefulWidget {
  const MyApp({super.key});
  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
        debugShowCheckedModeBanner: false,
        title: 'Clg Management',
        themeMode: ThemeMode.light,
        theme: ThemeData(
            primarySwatch: Colors.teal,
            backgroundColor: Colors.teal[300],
  textTheme: const TextTheme( headline1: TextStyle(fontSize: 21, fontWeight: FontWeight.bold),
     subtitle1: TextStyle(fontSize: 11, fontWeight: FontWeight.w500, fontStyle:FontStyle.italic)),
       fontFamily: GoogleFonts.lato().fontFamily),
    darkTheme: ThemeData(brightness: Brightness.dark),
    routes: {
      "/": (context) =>SplashPage(image: Lottie.asset('assets/images/img75.json')),
      MyRoutes.ClgWebpageRoute: (context) => const ClgWebPage(),
      MyRoutes.AboutClgpageRoute: (context) => const AboutClgPage(),
      MyRoutes.founderpageRoute: (context) => const AboutFounderPage(),
      MyRoutes.notiRoute: (context) => const NotificationPage()
    });}}
```

Home_Page.dart

```
import 'package:external_app_launcher/external_app_launcher.dart';
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:page_transition/page_transition.dart';
import'package:yns_college_management/pages/Admin/Admin%20Settings/addstudent_page.dart';
import 'package:yns_college_management/pages/College%20Web/my_drawer_header.dart';
import '../../Widgets/call_class_room_and_online_class.dart';
import '../../Widgets/home_page_widget.dart';
import 'Admin/Admin Settings/add_courses_page.dart';
import 'Admin/Admin Settings/add_teachers_or_admin_page.dart';

class HomePage extends StatefulWidget {
  String role;
  var username = "";
  var id = "";
  var profile = "";
  var department = "";
  var rollNo = "";
  var Class = "";
  HomePage(
 {super.key, required this.role, required this.Class, required this.rollNo, required this.department,
required this.id, required this.profile, required this.username});

  @override
  State<HomePage> createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(extendBody: true,
      backgroundColor: const Color.fromRGBO(100, 232, 222, 1.0),
      appBar: AppBar(title: Row(mainAxisAlignment: MainAxisAlignment.spaceBetween,
          children: [Text(widget.username),
            Row(children: [
              InkWell(onTap: () {},
                child: const Icon(FontAwesomeIcons.search, color: Colors.white)),
              const SizedBox(width: 10),
              const CallClasses(),
              const SizedBox(width: 20),
              InkWell( onTap: () async {
        await LaunchApp.openApp(androidPackageName: 'com.whatsapp',
      iosUrlScheme: 'https://www.apple.com/us/search/whatsapp?src=globalnav', openStore: true);
                },
                child: const Icon(FontAwesomeIcons.facebookMessenger, color: Colors.white))
            ])]),
        backgroundColor: Colors.transparent, elevation: 0),
      body: Container(decoration: const BoxDecoration(gradient: LinearGradient(colors: [
          Color.fromRGBO(165, 90, 255, 1.0),
          Color.fromRGBO(138, 100, 235, 1.0),
          Color.fromRGBO(100, 232, 222, 1.0),
        ], begin: Alignment.bottomLeft, end: Alignment.centerRight)),
        child: Column(children: [
          // Name Card...
          Padding(padding: const EdgeInsets.all(15.0),
            child: Container(height: 180,
              decoration: BoxDecoration(gradient: const LinearGradient(
                  colors: [
                    Color.fromRGBO(100, 232, 222, 1.0),
                    Color.fromRGBO(138, 120, 235, 1.0)
                  ], begin: Alignment.bottomLeft, end: Alignment.centerRight),
                boxShadow: [BoxShadow(offset: const Offset(5, 10), blurRadius: 20,
                    color: Colors.teal.shade900.withOpacity(0.2))],
                borderRadius: const BorderRadius.only(topLeft: Radius.circular(10), bottomLeft:
Radius.circular(10), bottomRight:Radius.circular(10), topRight: Radius.circular(80))),
              child: Padding(padding: const EdgeInsets.all(20.0),
                child: Row(mainAxisAlignment: MainAxisAlignment.spaceBetween,
                  children: [
                    const CircleAvatar(backgroundColor: Colors.white, radius: 50,
                      backgroundImage:   AssetImage("assets/images/img60.png")),
                    Expanded(child: Padding(padding: const EdgeInsets.all(15.0),
                child: FittedBox(child: Column(mainAxisAlignment: MainAxisAlignment.center,
                            children: [
                  RichText(softWrap: true, text: (widget.role =='admin' ||widget.role =='teacher')
                ? TextSpan(text: widget.username, style: const TextStyle(
                                      fontSize: 20,  fontWeight: FontWeight.bold),
                                    children: [const TextSpan(text: '\n'),
                                TextSpan(   text: widget.id, style: const TextStyle(fontSize: 15)),
                                      const TextSpan(text: '\n'),
                               TextSpan( text: widget.profile, style: const TextStyle(fontSize: 15)),
                                      const TextSpan(text: '\n'),
                          TextSpan(text: widget.department, style: const TextStyle(fontSize: 15)),])
            : TextSpan(text: widget.username, style: const TextStyle(fontSize: 20, fontWeight:
FontWeight.bold),
                                    children: [const TextSpan(text: '\n')
TextSpan(text: widget.rollNo, style: const TextStyle(fontSize: 15)),
                                      const TextSpan(text: '\n'),
                                TextSpan(text: widget.Class, style: const TextStyle(fontSize: 15)),
                                      const TextSpan(text: '\n'),
            TextSpan(text: widget.department, style: const TextStyle(fontSize: 15))]))]))))]))))),
          Expanded(
            child: Container(width: double.infinity, decoration: BoxDecoration(borderRadius: const
BorderRadius.only( topLeft: Radius.circular(40), topRight: Radius.circular(40)),
                gradient: LinearGradient(
                  colors: [
                    Colors.teal.shade700.withOpacity(0.1),
                    Colors.teal.shade500.withOpacity(0.5)
                  ],  begin: Alignment.bottomLeft, end: Alignment.centerRight)),
              // for Admin & teacher...
              child: (widget.role == 'admin' ||
                  widget.role == 'teacher')
                ? GridView.count(crossAxisCount: 2, children: [
                  const AttendanceTakerBtn(),
                  const CheckAttendanceBtn(),
                  LibraryBtn(role: widget.role),
                  CalendarBtn(role: widget.role),
                  const TimeTableBtn(),
                  const ResultBtn(),
                  const TransportBtn(),
                  NoticeBoardBtn(role: widget.role),
                  const IdCardBtn(),
                  HomeWorkBtn(role: widget.role),
                  const ApplyLeaveBtn(),
                  if (widget.role == 'admin')
                    // Admin Settings
                    ContainerWidget(
                      text: 'Admin\nSettings',
                      icon: Icons.settings,
                      ontap: () {
                        showDialog(
                            context: context,
                            builder: (ctx) => AlertDialog(
                                shape: RoundedRectangleBorder(
                               borderRadius:
                                   BorderRadius.circular(
                                    30.0)),
                                backgroundColor:
                               const Color.fromRGBO(
                                   100, 232, 222, 0.7),
                                content: SizedBox(
                               child: SingleChildScrollView(
                                   child: Column(
                                    crossAxisAlignment:
                                        CrossAxisAlignment
                                            .start,
                                    children: [
                                TextStyleWidget(
                                   text: 'Add Student',
                                   ontap: (() {
                                    Navigator.push(
      context,
      PageTransition(
          type:
              PageTransitionType
                  .fade,
          child:
              const SRegistrationPage()));
}),
icon: FontAwesomeIcons
.userPlus),
TextStyleWidget(
text:
'Add Teacher Or Admin',
ontap: (() {
  Navigator.push(
      context,
      PageTransition(
          type:
              PageTransitionType
                  .fade,
          child:
              const TRegistrationPage()));
}),
icon: FontAwesomeIcons
.userPlus),
TextStyleWidget(
text:
'Edit Student Details',
ontap: (() {}),
icon: FontAwesomeIcons
.userPen),
TextStyleWidget(
text:
'Edit Teacher Or Admin Details',
ontap: (() {}),
icon: FontAwesomeIcons
.userPen),
TextStyleWidget(
text: 'Delete Student',
ontap: (() {}),
icon: FontAwesomeIcons
.userXmark),
TextStyleWidget(
text:
'Delete Teacher Or Admin',
ontap: (() {}),
icon: FontAwesomeIcons
.userXmark),
TextStyleWidget(
text: 'Add Courses',
ontap: (() {
```

```
                                          Navigator.push(
                                              context,
                                              PageTransition(
                                                  type:
                                                      PageTransitionType
                                                          .fade,
                                                  child:
                                                      const AddCoursesPage()));
                                        }),
                                        icon: FontAwesomeIcons
                                            .bookMedical),
                                    TextStyleWidget(
                                        text: 'Delete Courses',
                                        ontap: (() {}),
                                        icon: FontAwesomeIcons
                                            .trash),
                                    TextStyleWidget(
                                        text: 'Add Time Table',
                                        ontap: (() {}),
                                        icon: FontAwesomeIcons
                                            .calendarPlus),
                                    TextStyleWidget(
                                        text: 'Add Marks',
                                        ontap: (() {}),
                                        icon: FontAwesomeIcons
                                            .marker),
                                    TextStyleWidget(
                                        text:
                                            'Generate Reports',
                                        ontap: (() {}),
                                        icon: FontAwesomeIcons
                                            .file)
                                  ])))));
                        })
                    ])
                  // for Student...
                  : GridView.count(crossAxisCount: 2, children: [
                      const CheckAttendanceBtn(),
                      LibraryBtn(role: widget.role),
                      CalendarBtn(role: widget.role),
                      const TimeTableBtn(),
                      const ResultBtn(),
                      const TransportBtn(),
                      NoticeBoardBtn(role: widget.role),
                      const IdCardBtn(),
                      HomeWorkBtn(role: widget.role),
                      const ApplyLeaveBtn()
                    ]))
          ])),
      // App Drawer
      drawer: Drawer(
          child: SingleChildScrollView(
              child: Column(
                  children: const [MyHeaderDrawer(), MyDrawerlist()]))));
  }
}
```

Login_Page.dart

```
// ignore_for_file: prefer_const_constructors, use_build_context_synchronously
import 'package:flutter/material.dart';
import 'package:page_transition/page_transition.dart';
import 'package:yns_college_management/pages/signup_page.dart';
import '../../Resources/auth_method.dart';
import '../../Utils/utils.dart';
import '../../Widgets/siginup_and_login_page_widget.dart';
import 'navigation_bar.dart';

class LoginPage extends StatefulWidget {
  const LoginPage({super.key});
  @override
  State<LoginPage> createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  // String role = '';
  final TextEditingController email = TextEditingController();
  final TextEditingController password = TextEditingController();
  var isPasswordVisible = true;
  bool isLoading = false;
  final forrmkey = GlobalKey<FormState>();


  @override
  void dispose() {
    super.dispose();
    email.dispose();
    password.dispose();
  }

  void loginUser() async {
    setState(() {
      isLoading = true;
    });
    String res = await AuthMethods()
        .loginUser(email: email.text, password: password.text);
    if (res == 'success') {
      Navigator.pushReplacement(context,
          PageTransition(type: PageTransitionType.fade, child: BottomNavBar()));
      setState(() {
        isLoading = false;
      });
  } else {
    setState(() {
      isLoading = false;
    });
    showSnackBar(context, res);
  }
}

@override
Widget build(BuildContext context) {
  var mediaQuery = MediaQuery.of(context);
  return Scaffold(
      backgroundColor: Color.fromRGBO(165, 90, 255, 1.0),
      appBar: AppBar(
          elevation: 0,
          backgroundColor: Colors.transparent,
          toolbarHeight: 28),
      body: Container(
          height: double.infinity,
          decoration: const BoxDecoration(
              gradient: LinearGradient(colors: [
            Color.fromRGBO(165, 90, 255, 1.0),
            Color.fromRGBO(195, 77, 235, 1),
            Color.fromRGBO(100, 232, 222, 1.0),
            Color.fromRGBO(88, 199, 189, 1)
          ], begin: Alignment.topLeft, end: Alignment.centerLeft)),
          child: SingleChildScrollView(
              child: Column(children: [
            Column(crossAxisAlignment: CrossAxisAlignment.start, children: [
              Padding(
                  padding: const EdgeInsets.only(left: 25),
                  child: Text('Welcome Back',
                      style: TextStyle(
                          fontSize: 30,
                          color: Colors.teal[800],
                          fontWeight: FontWeight.bold,
                          fontFamily: 'Pacifico'))),
              Padding(
                  padding: const EdgeInsets.only(left: 25),
                  child: Text('Login back into your account',
                      style: TextStyle(
                          fontWeight: FontWeight.bold,
                          color: Colors.teal[700],
                          fontFamily: 'Pacifico'))),
              Padding(
                  padding:
                      const EdgeInsets.only(top: 15.0, left: 15, right: 15),
                  child: Container(
                      height: mediaQuery.size.height * 0.64,
                      width: double.infinity,
                      decoration: BoxDecoration(
    gradient: const LinearGradient(
        colors: [
          Color.fromRGBO(100, 232, 221, 0.500),
          Color.fromRGBO(137, 120, 235, 0.500)
        ],
        begin: Alignment.bottomLeft,
        end: Alignment.centerRight),
    boxShadow: [
      BoxShadow(
          offset: const Offset(5, 10),
          blurRadius: 20,
          color: Colors.teal.shade900.withOpacity(0.5))
    ],
    borderRadius: const BorderRadius.only(
        topLeft: Radius.circular(100),
        bottomLeft: Radius.circular(10),
        bottomRight: Radius.circular(100),
        topRight: Radius.circular(50))),
child: Padding(
    padding: const EdgeInsets.all(20.0),
    child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          // Title
          Padding(
              padding: const EdgeInsets.only(
                  top: 30, left: 35, bottom: 20),
              child: Text('Log In',
                  style: TextStyle(
                      fontSize: 40,
                      color: Colors.white,
                      fontWeight: FontWeight.bold,
                      fontFamily: 'Pacifico'))),
          SizedBox(height: 5),
          // Enter Email ID...
          SignUpPageTextField(
              textEditingController: email,
              hint: 'YNSManagement@gmail.com',
              label: 'Enter Your Email ID',
              icon: Icons.email),
          // Create Password...
          Padding(
              padding: const EdgeInsets.all(5.0),
              child: SizedBox(
                  height: mediaQuery.size.height * 0.09,
                  child: TextField(
                      controller: password,
                      style: const TextStyle(
                          fontSize: 14,
                          fontStyle: FontStyle.normal,
                          color: Colors.white),
                                    cursorColor: Colors.white,
                                    obscureText: isPasswordVisible,
                                    decoration: InputDecoration(
                                      hintText: 'v55tyr54@',
                                      hintStyle: TextStyle(
                                        fontStyle:
                                          FontStyle.normal,
                                        color: Color.fromARGB(
                                          117, 255, 255, 255)),
                                      labelText: "Create Password",
                                      labelStyle: TextStyle(
                                        fontSize: 15,
                                        fontStyle:
                                          FontStyle.italic,
                                        color: Colors.white),
                                      prefixIcon: const Icon(
                                        Icons.lock,
                                        color: Colors.white),
                                      suffixIcon: IconButton(
                                        color: Colors.white,
                                        icon: isPasswordVisible
                                          ? const Icon(
                                            Icons.visibility)
                                          : const Icon(Icons
                                            .visibility_off),
                                        onPressed: () {
                                         if (isPasswordVisible ==
                                           true) {
                                          isPasswordVisible =
                                            false;
                                         } else {
                                          isPasswordVisible =
                                            true;
                                         }
                                         setState(() {});
                                        }),
                                      focusedBorder: OutlineInputBorder(
                                        borderRadius:
                                          BorderRadius.circular(
                                            20),
                                        borderSide: const BorderSide(
                                          color: Colors.white,
                                          width: 2)),
                                      enabledBorder: OutlineInputBorder(
                                        borderRadius:
                                          BorderRadius.circular(20),
                                        borderSide: BorderSide(color: Color.fromARGB(117, 255, 255,
255), width: 1.5)),
                                      border: OutlineInputBorder(borderRadius:
BorderRadius.circular(20))))),
                              // Log in Button...
```

```
Padding(
    padding: const EdgeInsets.only(top: 10.0),
    child: Center(
        child: SizedBox(
            width:
                mediaQuery.size.width * 0.6,
            child: ElevatedButton(
                onPressed: () {
                  loginUser();
                },
                style: ElevatedButton.styleFrom(
                    elevation: 20,
                    backgroundColor:
                        Colors.teal[500],
                    shadowColor:
                        Colors.teal[400],
                    side: BorderSide(
                        color: Colors
                            .teal.shade500,
                        width: 2,
                        style: BorderStyle
                            .solid),
                    shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(
                            20.0)),
                    minimumSize:
                        const Size(20, 50)),
                child: !isLoading
                    ? const Text('Log In')
                    : const CircularProgressIndicator(
                        color: Color.fromARGB(
                            255, 137, 167, 172))))),
// Forgot Password...
Center(
    child: SignUpPageChangeButton(
        text1: 'Forgot Password?',
        text2: '',
        ontap: () {})),
// sign up link...
Center(
    child: SignUpPageChangeButton(
        text1: 'Don\'t have an account?  ',
        text2: 'Sign Up',
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType
                      .rightToLeft,
                  child: const SignUpPage()));
        }))
```

```
                ])))),
            // signup with...
            Center(
              child: SignUpPageChangeButton(
                text1: '--- Log In With ---',
                text2: '',
                ontap: () {
                  Navigator.push(
                      context,
                      PageTransition(
                          type: PageTransitionType.rightToLeft,
                          child: const LoginPage()));
                })),
            // logo
            Center(
              child: Padding(
                padding: const EdgeInsets.symmetric(horizontal: 70),
                child: Row(
                    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                    children: const [
                      CircleAvatar(
                          backgroundColor: Colors.white,
                          radius: 15,
                          backgroundImage:
                              AssetImage("assets/images/img67.png")),
                      CircleAvatar(
                          backgroundColor: Colors.white,
                          radius: 15,
                          backgroundImage:
                              AssetImage("assets/images/img68.jpg")),
                      CircleAvatar(
                          backgroundColor: Colors.white,
                          radius: 15,
                          backgroundImage:
                              AssetImage("assets/images/img69.png")),
                      CircleAvatar(
                          backgroundColor: Colors.white,
                          radius: 15,
                          backgroundImage:
                              AssetImage("assets/images/img70.jpg"))
                    ])))
          ])
        ])));
  }
}
```

Profile_page.dart

```
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:url_launcher/url_launcher.dart';
import 'package:yns_college_management/Widgets/log_out.dart';
import '../../Widgets/call_class_room_and_online_class.dart';
import '../../Widgets/profile_widget.dart';

class ProfilePage extends StatefulWidget {
  var role = "",
      username = "",
      session = "",
      id = "",
      rollNo = "",
      profile = "",
      Class = "",
      department = "",
      phoneNo = "",
      fName = "",
      mName = "",
      dob = "",
      subject = "",
      language = "",
      aadharNo = "",
      gender = "",
      category = "",
      occupation = "",
      email = "",
      income = "",
      address = "";
  ProfilePage(
      {super.key,
      required this.role,
      required this.Class,
      required this.aadharNo,
      required this.address,
      required this.category,
      required this.department,
      required this.dob,
      required this.email,
      required this.fName,
      required this.gender,
      required this.id,
      required this.income,
      required this.language,
      required this.mName,
      required this.occupation,
      required this.phoneNo,
      required this.profile,
      required this.rollNo,
      required this.session,
      required this.subject,
      required this.username});

  @override
  State<ProfilePage> createState() => _ProfilePageState();
```

```
}

class _ProfilePageState extends State<ProfilePage> {
  var present = "86", absent = "14";

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.teal[300],
      //aap bar
      appBar: AppBar(
          backgroundColor: Colors.teal[400],
          title:
              Row(mainAxisAlignment: MainAxisAlignment.spaceBetween, children: [
            // Id...
            (widget.role == 'admin' || widget.role == 'teacher')
                ? Text(widget.id)
                : (Text(widget.rollNo)),
            Row(children: const [
              // google classroom, zoom, and logout...
              CallClasses(),
              SizedBox(width: 20),
              LogOut()
            ])
          ]),
          elevation: 0),
      body: Column(crossAxisAlignment: CrossAxisAlignment.start, children: [
        Container(
            decoration: BoxDecoration(
                color: Colors.teal[400],
                borderRadius: const BorderRadius.only(
                    bottomRight: Radius.elliptical(120, 90))),
            child:
                Column(crossAxisAlignment: CrossAxisAlignment.start, children: [
              Padding(
                  padding: const EdgeInsets.only(top: 8, left: 15),
                  child: Row(
                      mainAxisAlignment: MainAxisAlignment.spaceAround,
                      children: [
                        const CircleAvatar(
                            backgroundColor: Colors.white,
                            radius: 50,
                            backgroundImage:
                                AssetImage("assets/images/img60.png")),
                        Expanded(
                            child: Row(
                                mainAxisAlignment:
                                    MainAxisAlignment.spaceEvenly,
                                children: [
                              Padding(
                                  padding: const EdgeInsets.only(
          left: 0, bottom: 15),
        child: SizedBox(
          width: 180,
          child: Column(
            mainAxisAlignment:
              MainAxisAlignment.start,
            crossAxisAlignment:
              CrossAxisAlignment.start,
            children: [
             Text(widget.username,
               style: TextStyle(
                 fontSize: 18,
                 fontWeight: FontWeight.bold,
                 color: Colors.teal[900])),
             Text(
               (widget.role == 'admin' ||
                   widget.role ==
                     'teacher')
                 ? widget.profile
                 : widget.Class,
               style: TextStyle(
                 fontSize: 15,
                 fontWeight: FontWeight.bold,
                 color: Colors.teal[900])),
             Text(widget.department,
               style: TextStyle(
                 fontSize: 15,
                 fontWeight: FontWeight.bold,
                 color: Colors.teal[900])),
             const SizedBox(height: 8),
             if (widget.role == 'student')
              Row(
                mainAxisAlignment:
                  MainAxisAlignment
                    .spaceBetween,
                children: [
                 profileWidget(
                   percent: '$present%',
                   name: 'PRESENTS'),
                 profileWidget(
                   percent: '$absent%',
                   name: 'ABSENTS')
                ])
            ]))
     ]))
  ])),
Padding(
  padding: const EdgeInsets.only(left: 60, right: 60, top: 20),
  child: Container(
    width: double.infinity,
    height: 35,
              decoration: BoxDecoration(
                boxShadow: [
                 BoxShadow(
                   offset: const Offset(5, 10),
                   blurRadius: 20,
                   color: Colors.teal.shade900.withOpacity(0.6))
                ],
                color: Colors.teal[600],
                borderRadius: const BorderRadius.only(
                  topLeft: Radius.circular(10),
                  topRight: Radius.circular(10),
                  bottomLeft: Radius.circular(10),
                  bottomRight: Radius.elliptical(120, 90))),
              child: InkWell(
                onTap: (() {
                 var number = widget.phoneNo;
                 launch('http://wa.me/+91$number');
                }),
                child: Row(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: const [
                   Icon(FontAwesomeIcons.facebookMessenger,
                     color: Colors.white),
                   SizedBox(width: 8),
                   Text('Message',
                     style: TextStyle(
                       fontSize: 15,
                       fontWeight: FontWeight.bold,
                       color: Colors.white))
                  ]))),
          const SizedBox(height: 20)
        ])),
      Expanded(
        child: SingleChildScrollView(
          child: Padding(
            padding: const EdgeInsets.all(15),
            child: Column(children: [
              // academic session
              if (widget.role == 'student')
               AdminRowWidget(
                 text1: 'Academic Session:', text2: widget.session),
              // Id or RollNo
              (widget.role == 'admin' || widget.role == 'teacher')
                ? AdminRowWidget(
                  text1: '${widget.role} Id:', text2: widget.id)
                : AdminRowWidget(
                  text1: 'Roll-No.:', text2: widget.rollNo),
              // profile or class
              (widget.role == 'admin' || widget.role == 'teacher')
                ? AdminRowWidget(
                  text1: 'Job Profile:', text2: widget.profile)
```

```
    : AdminRowWidget(
        text1: 'Class:', text2: widget.Class),
// Department
AdminRowWidget(
    text1: 'Department', text2: widget.department),
// Subjects for teachers
if (widget.role != 'student')
  AdminRowWidget(
      text1: 'Subjects:',
      text2: widget.subject), // display all subjects...
// language for teachers
if (widget.role != 'student')
  AdminRowWidget(
      text1: 'Language:',
      text2: widget.language), // display all language...
// Name
(widget.role == 'admin' || widget.role == 'teacher')
    ? AdminRowWidget(
        text1: '${widget.role} Name:',
        text2: widget.username)
    : AdminRowWidget(
        text1: 'Student Name:', text2: widget.username),
// Father Name
AdminRowWidget(
    text1: 'Father\'s Name:', text2: widget.fName),
// Mother Name
AdminRowWidget(
    text1: 'Mother\'s Name:', text2: widget.mName),
// Date fo Birth
AdminRowWidget(
    text1: 'Date Of Birth:', text2: widget.dob),
// Aadhar card number
AdminRowWidget(
    text1: 'Aadhar Card No.:', text2: widget.aadharNo),
// Gender
AdminRowWidget(text1: 'Gender:', text2: widget.gender),
// Category
if (widget.role == 'student')
  AdminRowWidget(
      text1: 'Category:', text2: widget.category),
// Guardian's Occupation
if (widget.role == 'student')
  AdminRowWidget(
      text1: 'Guardian\'s Occupation:',
      text2: widget.occupation),
// Guardian's Income
if (widget.role == 'student')
  AdminRowWidget(
      text1: 'Guardian\'s Income:', text2: widget.income),
// Address
AdminRowWidget(text1: 'Address:', text2: widget.address),
                    // Phone Number
                    AdminRowWidget(
                        text1: 'Phone Number:', text2: widget.phoneNo),
                    // Email Id
                    AdminRowWidget(text1: 'Email Id:', text2: widget.email),
                    const SizedBox(height: 10),
                    // social media icon
                    Row(
                        mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                        children: [
                          InkWell(
                              onTap: () {},
                              child: Icon(FontAwesomeIcons.facebook,
                                  color: Colors.teal[800])),
                          InkWell(
                              onTap: () {},
                              child: Icon(FontAwesomeIcons.instagram,
                                  color: Colors.teal[800])),
                          InkWell(
                              onTap: () {},
                              child: Icon(FontAwesomeIcons.twitter,
                                  color: Colors.teal[800])),
                          InkWell(
                              onTap: () {},
                              child:
                                  Icon(Icons.mail, color: Colors.teal[800]))
                        ]),
                    const SizedBox(height: 60)
                  ]))))
        ]),
      );
    }
  }
```

Signup_page.dart

```
import 'package:flutter/material.dart';
import 'package:page_transition/page_transition.dart';
import 'package:yns_college_management/Resources/auth_method.dart';
import 'package:yns_college_management/pages/login_page.dart';
import '../../Utils/utils.dart';
import '../../Widgets/siginup_and_login_page_widget.dart';
import 'navigation_bar.dart';

class SignUpPage extends StatefulWidget {
  const SignUpPage({super.key});
  @override
  State<SignUpPage> createState() => Controller();
}

class Controller extends State<SignUpPage> {
  var isPasswordVisible = true;
var isPasswordVisible2 = true;
bool _isLoading = false;
var role;
// Controller...
final TextEditingController Id = TextEditingController();
final TextEditingController Email = TextEditingController();
final TextEditingController Password = TextEditingController();
final forrmkey = GlobalKey<FormState>();

@override
void dispose() {
  super.dispose();
  Id.dispose();
  Email.dispose();
  Password.dispose();
}

void signUpUser() async {
  // set loading to true
  setState(() {
    _isLoading = true;
  });

  // sign up user using our authMethods
  String res = await AuthMethods().AddAdminOrTeacher(
      email: Email.text,
      password: Password.text,
      aadharNo: '',
      address: '',
      department: '',
      dob: '',
      id: Id.text,
      fName: '',
      gender: '',
      language: '',
      name: '',
      mName: '',
      phoneNo: '',
      profile: '',
      role: '',
      subject: '');
  if (res == "Success") {
    // navigate to the home screen
    Navigator.pushReplacement(context,
        PageTransition(type: PageTransitionType.fade, child: BottomNavBar()));
    setState(() {
    });
  } else {
    setState(() {
      _isLoading = false;
    });
```

```
 }
 // show the error
 showSnackBar(context, res);
}

@override
Widget build(BuildContext context) {
 var mediaQuery = MediaQuery.of(context);
 return Scaffold(
    backgroundColor: Color.fromRGBO(165, 90, 255, 1.0),
    appBar: AppBar(
       elevation: 0,
       backgroundColor: Colors.transparent,
       toolbarHeight: 28),
    body: Container(
       height: double.infinity,
       decoration: const BoxDecoration(
          gradient: LinearGradient(colors: [
        Color.fromRGBO(165, 90, 255, 1.0),
        Color.fromRGBO(195, 77, 235, 1),
        Color.fromRGBO(100, 232, 222, 1.0),
        Color.fromRGBO(88, 199, 189, 1)
       ], begin: Alignment.topLeft, end: Alignment.centerLeft)),
       child: SingleChildScrollView(
          child: Column(children: [
        Column(crossAxisAlignment: CrossAxisAlignment.start, children: [
          Padding(
             padding: const EdgeInsets.only(left: 25),
             child: Text('Welcome',
                style: TextStyle(
                   fontSize: 30,
                   color: Colors.teal[800],
                   fontWeight: FontWeight.bold,
                   fontFamily: 'Pacifico'))),
          Padding(
             padding: const EdgeInsets.only(left: 20),
             child: Text('Signup into your account',
                style: TextStyle(
                   fontWeight: FontWeight.bold,
                   color: Colors.teal[700],
                   fontFamily: 'Pacifico'))),
          Padding(
             padding:
                const EdgeInsets.only(top: 15.0, left: 15, right: 15),
             child: Container(
                height: mediaQuery.size.height * 0.68,
                width: double.infinity,
                decoration: BoxDecoration(
                   gradient: const LinearGradient(
                      colors: [
                        Color.fromRGBO(100, 232, 221, 0.500),
                        Color.fromRGBO(137, 120, 235, 0.500)
                      ],
                      begin: Alignment.bottomLeft,
                      end: Alignment.centerRight),
                   boxShadow: [
                     BoxShadow(
                        offset: const Offset(5, 10),
                        blurRadius: 20,
                        color: Colors.teal.shade900.withOpacity(0.5))
                   ],
                   borderRadius: const BorderRadius.only(
                      topLeft: Radius.circular(100),
                      bottomLeft: Radius.circular(10),
                      bottomRight: Radius.circular(100),
                      topRight: Radius.circular(50))),
                child: Padding(
                   padding: const EdgeInsets.all(20.0),
                   child: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        Padding(
                            padding: const EdgeInsets.only(
                                top: 20, left: 25),
                            child: Text('Sign Up',
                                style: TextStyle(
                                    fontSize: 30,
                                    color: Colors.white,
                                    fontWeight: FontWeight.bold,
                                    fontFamily: 'Pacifico'))),
                        SizedBox(height: 0),
                        // Role
                        Column(children: [
                          Center(
                              child: FittedBox(
                                  child: Row(
                                      mainAxisAlignment:
                                          MainAxisAlignment.start,
                                      children: [
                                const Text('Admin',
                                    style: TextStyle(
                                        color: Colors.white)),
                                Radio(
                                    value: 'admin',
                                    activeColor: Colors.white,
                                    fillColor: MaterialStateColor
                                        .resolveWith(
                                            (states) => Colors.white),
                                    groupValue: role,
                                    onChanged: (value) {
                                      setState(() {
                                        role = value.toString();
                              });
                            }),
                        const Text('Teacher',
                            style: TextStyle(
                                color: Colors.white)),
                        Radio(
                            value: 'teacher',
                            activeColor: Colors.white,
                            fillColor: MaterialStateColor
                                .resolveWith(
                                    (states) => Colors.white),
                            groupValue: role,
                            onChanged: (value) {
                              setState(() {
                                role = value.toString();
                              });
                            }),
                        const Text('Student',
                            style: TextStyle(
                                color: Colors.white)),
                        Radio(
                            value: 'student',
                            activeColor: Colors.white,
                            fillColor: MaterialStateColor
                                .resolveWith(
                                    (states) => Colors.white),
                            groupValue: role,
                            onChanged: (value) {
                              setState(() {
                                role = value.toString();
                              });
                            })
                      ]))
                ]),
                // Enter Your Admin ID....
                SignUpPageTextField(
                     textEditingController: Id,
                     hint: 'AD203458',
                     label: 'Create Your Admin ID',
                     icon: Icons.person),
                // Enter Email ID...
                SignUpPageTextField(
                     textEditingController: Email,
                     hint: 'YNSManagement@gmail.com',
                     label: 'Enter Your Email ID',
                     icon: Icons.email),
                // Create Password...
                Padding(
                     padding: const EdgeInsets.all(5.0),
                     child: SizedBox(
                        height: mediaQuery.size.height * 0.09,
                child: TextField(
                controller: Password,
                style: const TextStyle(
                   fontSize: 14,
                   fontStyle: FontStyle.normal,
                   color: Colors.white),
                cursorColor: Colors.white,
                obscureText: isPasswordVisible,
                decoration: InputDecoration(
                   hintText: 'v55tyr54@',
                   hintStyle: TextStyle(
                     fontStyle:
                      FontStyle.normal,
                     color: Color.fromARGB(
                      117, 255, 255, 255)),
                   labelText: "Create Password",
                   labelStyle: TextStyle(
                     fontSize: 15,
                     fontStyle:
                      FontStyle.italic,
                     color: Colors.white),
                   prefixIcon: const Icon(
                     Icons.lock,
                     color: Colors.white),
                   suffixIcon: IconButton(
                     color: Colors.white,
                     icon: isPasswordVisible
                      ? const Icon(
                        Icons.visibility)
                      : const Icon(Icons
                        .visibility_off),
                     onPressed: () {
                      if (isPasswordVisible ==
                        true) {
                      isPasswordVisible =
                        false;
                      } else {
                      isPasswordVisible =
                        true;
                      }
                      setState(() {});
                     }),
                   focusedBorder: OutlineInputBorder(
                     borderRadius:
                      BorderRadius.circular(
                        20),
                     borderSide: const BorderSide(
                      color: Colors.white,
                      width: 2)),
                   enabledBorder: OutlineInputBorder(
                     borderRadius:
```

```
                                                BorderRadius.circular(20),
                                            borderSide: BorderSide(color: Color.fromARGB(117, 255, 255,
255), width: 1.5)),
                                        border: OutlineInputBorder(borderRadius:
BorderRadius.circular(20)))))),
                              // Sign Up Button...
                              Padding(
                                padding: const EdgeInsets.all(10.0),
                                child: Center(
                                  child: SizedBox(
                                    width:
                                        mediaQuery.size.width * 0.6,
                                    child: ElevatedButton(
                                      onPressed: () {
                                        signUpUser();
                                      },
                                      style: ElevatedButton.styleFrom(
                                          elevation: 20,
                                          backgroundColor:
                                              Colors.teal[500],
                                          shadowColor:
                                              Colors.teal[400],
                                          side: BorderSide(
                                              color: Colors
                                                  .teal.shade500,
                                              width: 2,
                                              style: BorderStyle
                                                  .solid),
                                          shape: RoundedRectangleBorder(
                                              borderRadius: BorderRadius.circular(
                                                  20.0)),
                                          minimumSize:
                                              const Size(20, 50)),
                                      child: !_isLoading
                                          ? const Text('Sign up')
                                          : const CircularProgressIndicator(
                                              color: Color.fromARGB(
                                                  255, 137, 167, 172)))))),
                              // Login Button...
                              Center(
                                child: SignUpPageChangeButton(
                                  text1: 'Already Registered?  ',
                                  text2: 'Login.',
                                  ontap: () {
                                    Navigator.push(
                                        context,
                                        PageTransition(
                                            type: PageTransitionType
                                                .rightToLeft,
                                            child: const LoginPage()));
                                  }))
```

```
                ])))),
            // signup with...
            Center(
              child: SignUpPageChangeButton(
                text1: '--- Sign Up With ---',
                text2: '',
                ontap: () {
                  Navigator.push(
                      context,
                      PageTransition(
                          type: PageTransitionType.rightToLeft,
                          child: const LoginPage()));
                })),
            // logo
            Center(
              child: Padding(
                padding: const EdgeInsets.symmetric(horizontal: 70),
                child: Row(
                    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                    children: [
                      const CircleAvatar(
                          backgroundColor: Colors.white,
                          radius: 15,
                          backgroundImage:
                              AssetImage("assets/images/img67.png")),
                      const CircleAvatar(
                          backgroundColor: Colors.white,
                          radius: 15,
                          backgroundImage:
                              AssetImage("assets/images/img68.jpg")),
                      const CircleAvatar(
                          backgroundColor: Colors.white,
                          radius: 15,
                          backgroundImage:
                              AssetImage("assets/images/img69.png")),
                      const CircleAvatar(
                          backgroundColor: Colors.white,
                          radius: 15,
                          backgroundImage:
                              AssetImage("assets/images/img70.jpg"))
                    ])))
          ])
        ]))));
  }
}
```

Auth_method.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:yns_college_management/models/user.dart' as model;
```

```
class AuthMethods {
  final FirebaseFirestore _firestore = FirebaseFirestore.instance;
  final FirebaseAuth _auth = FirebaseAuth.instance;

// Teachers or Admin Registration
  Future<String> AddAdminOrTeacher(
      {required String role,
      required String id,
      required String subject,
      required String profile,
      required String department,
      required String language,
      required String name,
      required String fName,
      required String mName,
      required String dob,
      required String aadharNo,
      required String gender,
      required String address,
      required String phoneNo,
      required String email,
      required String password}) async {
    String res = "Some Error occurred";
    try {
      if (role.isNotEmpty ||
          id.isNotEmpty ||
          subject.isNotEmpty ||
          profile.isNotEmpty ||
          department.isNotEmpty ||
          language.isNotEmpty ||
          name.isNotEmpty ||
          fName.isNotEmpty ||
          mName.isNotEmpty ||
          dob.isNotEmpty ||
          aadharNo.isNotEmpty ||
          gender.isNotEmpty ||
          address.isNotEmpty ||
          phoneNo.isNotEmpty ||
          email.isNotEmpty ||
          password.isNotEmpty) {
        // register the user...
        UserCredential cred = await _auth.createUserWithEmailAndPassword(
            email: email, password: password);
        //user model...
        model.AdminAndTeachers user = model.AdminAndTeachers(
            role: role,
            id: id,
            subject: subject,
            profile: profile,
            department: department,
            language: language,
            name: name,
            fName: fName,
            mName: mName,
            dob: dob,
            aadharNo: aadharNo,
            gender: gender,
            address: address,
            phoneNo: phoneNo,
            email: email,
            uid: cred.user!.uid);
        // add user in database...
        await _firestore
            .collection('users')
            .doc(cred.user!.uid)
            .set(user.toJson());
        res = "Success";
      }
    } catch (err) {
      res = err.toString();
    }
    return res;
  }

  // Student Registration
  Future<String> AddStudent(
      {required String role,
      required String session,
      required String Class,
      required String department,
      required String rollNo,
      required String name,
      required String fName,
      required String mName,
      required String dob,
      required String aadharNo,
      required String gender,
      required String category,
      required String gOccupation,
      required String gIncome,
      required String address,
      required String phoneNo,
      required String email,
      required String password}) async {
    String res = "Some Error occurred";
    try {
      if (role.isNotEmpty ||
          session.isNotEmpty ||
          Class.isNotEmpty ||
          department.isNotEmpty ||
          rollNo.isNotEmpty ||
          name.isNotEmpty ||
          fName.isNotEmpty ||
          mName.isNotEmpty ||
          dob.isNotEmpty ||
          aadharNo.isNotEmpty ||
          gender.isNotEmpty ||
          category.isNotEmpty ||
          gOccupation.isNotEmpty ||
          gIncome.isNotEmpty ||
          address.isNotEmpty ||
          phoneNo.isNotEmpty ||
          email.isNotEmpty ||
          password.isNotEmpty) {
        // register the user...
        UserCredential cred = await _auth.createUserWithEmailAndPassword(
            email: email, password: password);
        model.Student user = model.Student(
            role: role,
            session: session,
            Class: Class,
            department: department,
            rollNo: rollNo,
            name: name,
            fName: fName,
            mName: mName,
            dob: dob,
            aadharNo: aadharNo,
            gender: gender,
            category: category,
            gOccupation: gOccupation,
            gIncome: gIncome,
            address: address,
            phoneNo: phoneNo,
            email: email,
            uid: cred.user!.uid);
        // add user in database...
        await _firestore
            .collection('users')
            .doc(cred.user!.uid)
            .set(user.toJson());
        res = "Success";
      }
    } catch (err) {
      res = err.toString();
    }
    return res;
  }

  // logging in user
  Future<String> loginUser({
    required String email,
    required String password,
  }) async {
    String res = "Some error Occurred";
    try {
      if (email.isNotEmpty || password.isNotEmpty) {
        // logging in user with email and password
        await _auth.signInWithEmailAndPassword(
            email: email, password: password);
        res = "success";
      } else {
        res = "Please enter all the fields";
      }
    } catch (err) {
      return err.toString();
    }
    return res;
  }

  // log out
  Future<void> signOut() async {
    await _auth.signOut();
  }
}
```

Home_page_widget.dart

```
// ignore_for_file: deprecated_member_use, must_be_immutable
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:page_transition/page_transition.dart';
import 'package:yns_college_management/pages/Attendance/attendance_form.dart';
import 'package:yns_college_management/pages/Library/issue_book.dart';
import '../pages/Apply Leave/apply_leave.dart';
import '../pages/Calender/calender.dart';
import '../pages/Check Attendance/check_attendance.dart';
import '../pages/Home Work/home_work.dart';
import '../pages/Id Card/id_card.dart';
import '../pages/Library/return_book.dart';
import '../pages/Notice Board/notice_board.dart';
import '../pages/Result/result.dart';
import '../pages/Time Table/time_table.dart';
import '../pages/Transport/transport.dart';

// Container Widget....
class ContainerWidget extends StatelessWidget {
  final String text;
  final IconData icon;
  final VoidCallback ontap;
  const ContainerWidget(
      {Key? key, required this.text, required this.icon, required this.ontap})
      : super(key: key);

  @override
```

```
Widget build(BuildContext context) {
  var mediaQuery = MediaQuery.of(context);
  return Padding(
      padding: const EdgeInsets.all(30),
      child: Container(
          height: mediaQuery.size.width * 0.35,
          width: mediaQuery.size.width * 0.35,
          decoration: BoxDecoration(
              gradient: LinearGradient(colors: [
                Colors.teal.shade100,
                Colors.teal.shade300,
                Colors.teal.shade500,
              ], begin: Alignment.bottomLeft, end: Alignment.centerRight),
              borderRadius: const BorderRadius.all(Radius.circular(40))),
          child: ElevatedButton(
              onPressed: ontap,
              style: ElevatedButton.styleFrom(
                  elevation: 20,
                  backgroundColor: Colors.transparent,
                  shadowColor: Colors.teal[800],
                  side: BorderSide(
                      color: Colors.teal.shade500,
                      width: 2,
                      style: BorderStyle.solid),
                  shape: RoundedRectangleBorder(
                      borderRadius: BorderRadius.circular(35.0)),
                  minimumSize: const Size(200, 160)),
              child: FittedBox(
                  child: Padding(
                      padding: const EdgeInsets.all(40.0),
                      child: Column(
                          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                          children: [
                            Icon(icon,
                                color: Colors.white,
                                size: 300,
                                shadows: <Shadow>[
                                  Shadow(
                                      color: Colors.teal.shade800,
                                      blurRadius: 6.0,
                                      offset: const Offset(2, 2))
                                ]),
                            Text(text,
                                textAlign: TextAlign.center,
                                style: TextStyle(shadows: [
                                  Shadow(
                                      color: Colors.teal.shade900,
                                      blurRadius: 5,
                                      offset: const Offset(2, 2))
                                ], fontSize: 80, fontWeight: FontWeight.bold))
                          ]))))));
  }
}

// AttendanceTaker for Admin, Teachers Home Page
class AttendanceTakerBtn extends StatelessWidget {
  const AttendanceTakerBtn({super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Attendance\nTaker',
        icon: Icons.edit_calendar,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade,
                  child: const AttendanceForm()));
        });
  }
}

// AttendanceTaker for Admin, Teachers and student Home Page
class CheckAttendanceBtn extends StatelessWidget {
  const CheckAttendanceBtn({super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Check\nAttendance',
        icon: FontAwesomeIcons.calendarCheck,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade,
                  child: const CheckAttendance()));
        });
  }
}

// Library for Admin, Teachers and student Home Page
class LibraryBtn extends StatelessWidget {
  String role;
  LibraryBtn({required this.role, super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Library    ',
        icon: Icons.local_library,
        ontap: () {
          showDialog(
              context: context,
              builder: (ctx) => AlertDialog(
                  shape: RoundedRectangleBorder(
                      borderRadius: BorderRadius.circular(30.0)),
                  backgroundColor: const Color.fromRGBO(100, 232, 222, 0.7),
                  content: SizedBox(
                      child: SingleChildScrollView(
                          child: Column(
                              crossAxisAlignment: CrossAxisAlignment.start,
                              children: [
                        if (role != 'Students')
                          TextStyleWidget(
                              text: 'Issue Book',
                              ontap: (() {
                                Navigator.of(ctx).pop();
                                Navigator.push(
                                    context,
                                    PageTransition(
                                        type: PageTransitionType.fade,
                                        child: IssueBook(role: role)));
                              }),
                              icon: FontAwesomeIcons.book),
                        TextStyleWidget(
                            text: 'Return Book',
                            ontap: (() {
                              Navigator.of(ctx).pop();
                              Navigator.push(
                                  context,
                                  PageTransition(
                                      type: PageTransitionType.fade,
                                      child: ReturnBook(role: role)));
                            }),
                            icon: FontAwesomeIcons.book)
                      ])))));
        });
  }
}

// Calendar for Admin, Teachers and student Home Page
class CalendarBtn extends StatelessWidget {
  String role;
  CalendarBtn({required this.role, super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Calendar',
        icon: FontAwesomeIcons.calendar,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade, child: Calendar(role: role)));
        });
  }
}

// TimeTable for Admin, Teachers and student Home Page
class TimeTableBtn extends StatelessWidget {
  const TimeTableBtn({super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Time Table',
        icon: FontAwesomeIcons.clock,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade, child: const TimeTable()));
        });
  }
}

// Result for Admin, Teachers and student Home Page
class ResultBtn extends StatelessWidget {
  const ResultBtn({super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Result        ',
        icon: FontAwesomeIcons.chartLine,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade, child: const Result()));
        });
  }
}

// Transport for Admin, Teachers and student Home Page
class TransportBtn extends StatelessWidget {
  const TransportBtn({super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Transport',
        icon: FontAwesomeIcons.bus,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade, child: const Transport()));
        });
  }
}

// NoticeBoard for Admin, Teachers and student Home Page
class NoticeBoardBtn extends StatelessWidget {
  String role;
  NoticeBoardBtn({required this.role, super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Notice\nBoard',
        icon: FontAwesomeIcons.volumeHigh,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade,
                  child: NoticeBoard(role: role)));
        });
  }
}

// IdCard for Admin, Teachers and student Home Page
class IdCardBtn extends StatelessWidget {
  const IdCardBtn({super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'ID Card',
        icon: FontAwesomeIcons.idCard,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade, child: const IdCard()));
        });
  }
}

// HomeWork for Admin, Teachers and student Home Page
class HomeWorkBtn extends StatelessWidget {
  String role;
  HomeWorkBtn({required this.role, super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Home\nWork',
        icon: FontAwesomeIcons.pen,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade, child: HomeWork(role: role)));
        });
  }
}

// ApplyLeave for Admin, Teachers and student Home Page
class ApplyLeaveBtn extends StatelessWidget {
  const ApplyLeaveBtn({super.key});

  @override
  Widget build(BuildContext context) {
    return ContainerWidget(
        text: 'Apply\nLeave',
        icon: FontAwesomeIcons.personWalking,
        ontap: () {
          Navigator.push(
              context,
              PageTransition(
                  type: PageTransitionType.fade, child: const ApplyLeave()));
        });
  }
}

//Text Style For Admin Settings
class TextStyleWidget extends StatelessWidget {
  final String text;
  final VoidCallback ontap;
  final IconData icon;
  const TextStyleWidget(
      {Key? key, required this.text, required this.ontap, required this.icon})
      : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Row(children: [
      Icon(icon, size: 18, color: Colors.teal[900]),
      TextButton(
          onPressed: ontap,
          child: Text(text,
              style: TextStyle(
                  fontSize: 16,
                  color: Colors.teal[900],
                  fontStyle: FontStyle.normal)))
    ]);
  }
}
```

User.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';

// for Admin or Teacher...
class AdminAndTeachers {
  final String role;
  final String id;
  final String subject;
  final String profile;
  final String department;
  final String language;
  final String name;
  final String fName;
  final String mName;
  final String dob;
  final String aadharNo;
  final String gender;
  final String address;
  final String phoneNo;
  final String email;
  final String uid;

  const AdminAndTeachers(
      {required this.role,
      required this.id,
      required this.subject,
      required this.profile,
      required this.department,
      required this.language,
      required this.name,
      required this.fName,
      required this.mName,
      required this.dob,
      required this.aadharNo,
      required this.gender,
      required this.address,
      required this.phoneNo,
      required this.email,
      required this.uid});

  static AdminAndTeachers fromSnap(DocumentSnapshot snap) {
    var snapshot = snap.data() as Map<String, dynamic>;
    return AdminAndTeachers(
        role: snapshot["role"],
        id: snapshot["id"],
        subject: snapshot["subject"],
        profile: snapshot["profile"],
        department: snapshot["department"],
        language: snapshot["language"],
        name: snapshot["name"],
        fName: snapshot["fName"],
        mName: snapshot["mName"],
        dob: snapshot["dob"],
        aadharNo: snapshot["aadharNo"],
        gender: snapshot["gender"],
        address: snapshot["address"],
        phoneNo: snapshot[" phoneNo"],
        email: snapshot["email"],
        uid: snapshot["uid"]);
  }

  Map<String, dynamic> toJson() => {
        'role': role,
        'id': id,
        'subject': subject,
        'profile': profile,
        'department': department,
        'language': language,
        'name': name,
        'fName': fName,
        'mName': mName,
        'dob': dob,
        'aadharNo.': aadharNo,
        'gender': gender,
        'address': address,
        'phoneNo': phoneNo,
        'email': email,
        'uid': uid
      };
}

// for Student...
class Student {
  final String role;
  final String session;
  final String Class;
  final String department;
  final String rollNo;
  final String name;
  final String fName;
  final String mName;
  final String dob;
  final String aadharNo;
  final String gender;
  final String category;
  final String gOccupation;
  final String gIncome;
  final String address;
  final String phoneNo;
  final String email;
  final String uid;

  const Student(
      {required this.role,
      required this.session,
      required this.Class,
      required this.department,
      required this.rollNo,
      required this.name,
      required this.fName,
      required this.mName,
      required this.dob,
      required this.aadharNo,
      required this.gender,
      required this.category,
      required this.gOccupation,
      required this.gIncome,
      required this.address,
      required this.phoneNo,
      required this.email,
      required this.uid});

  static Student fromSnap(DocumentSnapshot snap) {
    var snapshot = snap.data() as Map<String, dynamic>;

    return Student(
        role: snapshot["role"],
        session: snapshot["session"],
        Class: snapshot["Class"],
        department: snapshot["department"],
        rollNo: snapshot["rollNo"],
        name: snapshot["name"],
        fName: snapshot["fName"],
        mName: snapshot["mName"],
        dob: snapshot["dob"],
        aadharNo: snapshot["aadharNo"],
        gender: snapshot["gender"],
        category: snapshot["category"],
        gOccupation: snapshot["gOccupation"],
        gIncome: snapshot["gIncome"],
        address: snapshot["address"],
        phoneNo: snapshot["phoneNo"],
        email: snapshot["email"],
        uid: snapshot["uid"]);
```

```
 }

 Map<String, dynamic> toJson() => {
       'role': role,
       'session': session,
       'Class': Class,
       'department': department,
       'rollNo.': rollNo,
       'name': name,
       'fName': fName,
       'mName': mName,
       'dob': dob,
       'aadharNo.': aadharNo,
       'gender': gender,
       'category': category,
       'gOccupation': gOccupation,
       'gIncome': gIncome,
       'address': address,
       'phoneNo.': phoneNo,
       'email': email,
       'uid': uid
     };
}
```

User.dart

```
import 'dart:typed_data';
import 'package:flutter/material.dart';
import 'package:image_picker/image_picker.dart';
import 'package:intl/intl.dart';
import 'package:yns_college_management/widgets/input_field_student_registration.dart';
import '../../../Resources/auth_method.dart';
import '../../../Utils/utils.dart';

class SRegistrationPage extends StatefulWidget {
  const SRegistrationPage({super.key});
  @override
  State<SRegistrationPage> createState() => _SRegistrationPageState();
}

class _SRegistrationPageState extends State<SRegistrationPage> {
  var role = 'student';
  var gender = '';
  var transport = '';
  var Category = '';
  //controller
  bool _isLoading = false;
  final TextEditingController sessionController = TextEditingController();
  final TextEditingController idController = TextEditingController();
  final TextEditingController nameController = TextEditingController();
  final TextEditingController fatherController = TextEditingController();
  final TextEditingController motherController = TextEditingController();
final TextEditingController dateController = TextEditingController();
final TextEditingController aadharController = TextEditingController();
final TextEditingController occupationController = TextEditingController();
final TextEditingController IncomeController = TextEditingController();
final TextEditingController AddressController = TextEditingController();
final TextEditingController MobileController = TextEditingController();
final TextEditingController EmailController = TextEditingController();
final TextEditingController PasswordController = TextEditingController();
Uint8List? _image;

var dropdownclass;
var dropdowndepartment;
var department = [
  'Computer Science Dep.',
  'Commerce & Business Dep.',
  'Teacher Education Dep.',
  'Biotechnology Dep.',
  'B.Sc(Home Science) Dep.',
  'B.Sc Department'
];
var classes = [
  'BCA',
  'MCA',
  'BBA',
  'MBA',
  'Bcom.',
  'MCom.',
  'BA',
  'MA',
  'B.Ed',
  'M.Ed',
  'D.EI.Ed',
  'B.Sc(Biotechnology)',
  'M.Sc(Biotechnology)',
  'B.Sc(HomeScience)',
  'M.Sc(HomeScience)',
  'B.Sc(Bio)-BCZ',
  'B.Sc(Math)-PCM'
];
@override
void dispose() {
  super.dispose();
  sessionController.dispose();
  // classController.dispose();
  idController.dispose();
  nameController.dispose();
  fatherController.dispose();
  motherController.dispose();
  dateController.dispose();
  aadharController.dispose();
  occupationController.dispose();
  IncomeController.dispose();
  AddressController.dispose();
  MobileController.dispose();
  EmailController.dispose();
  PasswordController.dispose();
  // departmentController.dispose();
}

void AddStudent() async {
  // set loading to true
  setState(() {
    _isLoading = true;
  });

  // Add Student  using our authMethods
  String res = await AuthMethods().AddStudent(
      role: role,
      email: EmailController.text,
      password: PasswordController.text,
      id: idController.text,
      Class: dropdownclass,
      aadharNo: aadharController.text,
      address: AddressController.text,
      category: Category,
      dob: dateController.text,
      fName: fatherController.text,
      gIncome: IncomeController.text,
      gOccupation: occupationController.text,
      department: dropdowndepartment,
      gender: gender,
      transport: transport,
      mName: motherController.text,
      name: nameController.text,
      phoneNo: MobileController.text,
      session: sessionController.text,
      file: _image!);
  // if string returned is success, user has been created
  if (res == "Success") {
    // navigate to the home screen
    Navigator.pop(context);
    Navigator.pop(context);
    setState(() {
      _isLoading = false;
    });
  } else {
    setState(() {
      _isLoading = false;
    });
  }
  // show the error
  showSnackBar(context, res);
  }

// for picking up image from gallery
  pickImage(ImageSource source) async {
    final ImagePicker imagePicker = ImagePicker();
    XFile? file = await imagePicker.pickImage(source: source);

    if (file != null) {
      return await file.readAsBytes();
    }
    // print('No Image Selected');
  }

  //select image
  selectImage() async {
    Uint8List im = await pickImage(ImageSource.gallery);

    // set state because we need to display the image we selected on the circle avatar
    setState(() {
      _image = im;
    });
  }

  @override
  Widget build(BuildContext context) {
    var mediaQuery = MediaQuery.of(context);
    return Scaffold(
        backgroundColor: Colors.teal[300],
        appBar: AppBar(
            title: (const Text("Student Registration ")),
            backgroundColor: Colors.transparent,
            elevation: 0),
        body: SingleChildScrollView(
            child:
                Column(crossAxisAlignment: CrossAxisAlignment.start, children: [
          // title...
          Center(
              child: SizedBox(
                  width: mediaQuery.size.width * 0.7,
                  height: mediaQuery.size.height * 0.15,
                  child: FittedBox(
                      child: Text("Student\n Registration",
                          textAlign: TextAlign.center,
                          style: TextStyle(
                              shadows: [
                                Shadow(
                                    color: Colors.teal.shade900,
                                    blurRadius: 5,
                                    offset: const Offset(2, 2))
                              ],
                              // fontSize: 50,
                fontWeight: FontWeight.bold,
                color: Colors.white))))),
Padding(
  padding: const EdgeInsets.only(
    left: 20, right: 20, top: 20, bottom: 2),
  child: Container(
    width: double.infinity,
    decoration: BoxDecoration(
      borderRadius: BorderRadius.circular(20),
      color: Colors.teal[400]),
    child: Column(children: [
     Padding(
       padding: const EdgeInsets.all(12),
       child: Column(
         crossAxisAlignment: CrossAxisAlignment.start,
         children: [
          // Academic Session...
          Row(children: [
           const Text('Academic Session:'),
           Expanded(
             child: InputFieldStudentRegistration(
               textEditingController:
                 sessionController,
               keyboard: const TextInputType
                 .numberWithOptions(signed: true)))
          ]),

          // Department
          Row(children: [
           const Text('Department:'),
           Expanded(
             child: Padding(
               padding:
                 const EdgeInsets.only(left: 20),
               child: DropdownButton(
                 dropdownColor: Colors.teal[400],
                 hint:
                   const Text('Select Department'),
                 menuMaxHeight: 300,
                 isExpanded: true,
                 underline: Container(
                  color: Colors.teal[800],
                  height: 1,
                 ),
                 iconEnabledColor: Colors.teal[800],
                 style: const TextStyle(
                   color: Color.fromARGB(
                     255, 13, 71, 161),
                   fontSize: 13),
                 value: dropdowndepartment,
                 icon: const Icon(
                  Icons.keyboard_arrow_down),
              items:
                  department.map((String items) {
                return DropdownMenuItem(
                    value: items,
                    child: Text(items));
              }).toList(),
              onChanged: (newValue) {
                setState(() {
                  dropdowndepartment = newValue!;
                });
              }))
]),

// class...
Row(children: [
  const Text('Class:'),
  Expanded(
      child: Padding(
          padding:
              const EdgeInsets.only(left: 20),
          child: DropdownButton(
              dropdownColor: Colors.teal[400],
              hint: const Text('Select Class'),
              menuMaxHeight: 300,
              isExpanded: true,
              underline: Container(
                color: Colors.teal[800],
                height: 1,
              ),
              iconEnabledColor: Colors.teal[800],
              style: const TextStyle(
                  color: Color.fromARGB(
                      255, 13, 71, 161),
                  fontSize: 13),
              value: dropdownclass,
              icon: const Icon(
                  Icons.keyboard_arrow_down),
              items: classes.map((String items) {
                return DropdownMenuItem(
                    value: items,
                    child: Text(items));
              }).toList(),
              onChanged: (newValue) {
                setState(() {
                  dropdownclass = newValue!;
                });
              }))
]),

// rollno...
```

```
          Row(children: [
            const Text('University Roll No.:'),
            Expanded(
                child: InputFieldStudentRegistration(
                    textEditingController: idController,
                    keyboard: TextInputType.text))
          ])
        ]))
    ]))),
//image
Padding(
    padding: const EdgeInsets.only(left: 20, right: 20, bottom: 2),
    child: Container(
        width: double.infinity,
        decoration: BoxDecoration(
            borderRadius: BorderRadius.circular(20),
            color: Colors.teal[400]),
        child: Center(
          child: Padding(
              padding: const EdgeInsets.all(12),
              child: Stack(
                children: [
                  _image != null
                      ? CircleAvatar(
                          radius: 64,
                          backgroundImage: MemoryImage(_image!),
                          backgroundColor: Colors.teal,
                        )
                      : const CircleAvatar(
                          radius: 64,
                          backgroundImage: NetworkImage(
                              'https://i.stack.imgur.com/l60Hf.png'),
                          backgroundColor: Colors.teal,
                        ),
                  Positioned(
                    bottom: -10,
                    left: 80,
                    child: IconButton(
                      onPressed: selectImage,
                      icon: const Icon(
                        Icons.add_a_photo,
                        color: Color.fromARGB(255, 0, 73, 65),
                      ),
                    ),
                  )
                ],
              )),
        ))),
Padding(
    padding: const EdgeInsets.only(left: 20, right: 20, bottom: 2),
    child: Container(
width: double.infinity,
decoration: BoxDecoration(
    borderRadius: BorderRadius.circular(20),
    color: Colors.teal[400]),
child: Column(children: [
  Padding(
      padding: const EdgeInsets.all(12),
      child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            // student name...
            Row(children: [
              const Text('Name of Candidate:'),
              Expanded(
                  child: InputFieldStudentRegistration(
                      textEditingController: nameController,
                      keyboard: TextInputType.name))
            ]),
            // father name...
            Row(children: [
              const Text("Father's Name:"),
              Expanded(
                  child: InputFieldStudentRegistration(
                      textEditingController: fatherController,
                      keyboard: TextInputType.name))
            ]),
            // mother name...
            Row(children: [
              const Text("Mother's Name:"),
              Expanded(
                  child: InputFieldStudentRegistration(
                      textEditingController: motherController,
                      keyboard: TextInputType.name))
            ]),
            // student dob...
            Row(children: [
              const Text("Date of Birth:"),
              Expanded(
                  child: Padding(
                      padding: const EdgeInsets.all(20.0),
                      child: TextField(
                        style: TextStyle(
                            fontSize: 13,
                            fontStyle: FontStyle.normal,
                            color: Colors.blue[900]),
                        controller: dateController,
                        decoration: const InputDecoration(
                            contentPadding: EdgeInsets.all(8),
                            icon: Icon(
                              Icons.calendar_today_rounded,
                              // color: Colors.teal[800],
                                ),
                                hintText: "Select Date"),
                              onTap: () async {
                                DateTime? pickedDate =
                                    await showDatePicker(
                                        context: context,
                                        initialDate: DateTime.now(),
                                        firstDate: DateTime(1900),
                                        lastDate: DateTime.now());
                                if (pickedDate != null) {
                                  setState(() {
                                    dateController.text =
                                        DateFormat('dd-MM-yyyy')
                                            .format(pickedDate);
                                  });
                                }
                              },
                            )))
                      ]),
                      // aadhar number...
                      Row(children: [
                        const Text("Aadhar No:"),
                        Expanded(
                            child: InputFieldStudentRegistration(
                                textEditingController: aadharController,
                                keyboard: TextInputType.datetime))
                      ])
                    ]))
              ]))),
          Padding(
            padding: const EdgeInsets.only(left: 20, right: 20, bottom: 2),
            child: Container(
              width: double.infinity,
              decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(20),
                color: Colors.teal[400]),
              child: Padding(
                padding: const EdgeInsets.all(12.0),
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.start,
                  children: [
                    // gender...
                    const Text("GENDER"),
                    FittedBox(
                      child: Row(
                        mainAxisAlignment: MainAxisAlignment.start,
                        children: [
                      const Text('Male'),
                      Radio(
                        value: 'male',
                        groupValue: gender,
          onChanged: (value) {
            setState(() {
              gender = value.toString();
            });
          }),
      SizedBox(width: mediaQuery.size.width * 0.1),
      const Text('female'),
      Radio(
          value: 'female',
          groupValue: gender,
          onChanged: (value) {
            setState(() {
              gender = value.toString();
            });
          }),
      SizedBox(width: mediaQuery.size.width * 0.1),
      const Text('Other'),
      Radio(
          value: 'other',
          groupValue: gender,
          onChanged: (value) {
            setState(() {
              gender = value.toString();
            });
          })
    ])),
const Divider(
    color: Colors.teal,
    height: 25,
    thickness: 1,
    indent: 5,
    endIndent: 5),
// category...
const Text("CATEGORY"),
FittedBox(
    child: Row(
        mainAxisAlignment: MainAxisAlignment.start,
        children: [
      const Text('Gen'),
      Radio(
          value: 'gen',
          groupValue: Category,
          onChanged: (value) {
            setState(() {
              Category = value.toString();
            });
          }),
      SizedBox(width: mediaQuery.size.width * 0.1),
      const Text('OBC'),
      Radio(
          value: 'OBC',
          groupValue: Category,
          onChanged: (value) {
            setState(() {
              Category = value.toString();
            });
          }),
        SizedBox(width: mediaQuery.size.width * 0.1),
        const Text('SC'),
        Radio(
          value: 'SC',
          groupValue: Category,
          onChanged: (value) {
            setState(() {
              Category = value.toString();
            });
          }),
        SizedBox(width: mediaQuery.size.width * 0.1),
        const Text('ST'),
        Radio(
          value: 'ST',
          groupValue: Category,
          onChanged: (value) {
            setState(() {
              Category = value.toString();
            });
          })
      ])),
  const Divider(
      color: Colors.teal,
      height: 25,
      thickness: 1,
      indent: 5,
      endIndent: 5),
  // occupation and income...
  const Text("Guardian's"),
  Row(children: [
    const Text("Occupation:"),
    Expanded(
        child: InputFieldStudentRegistration(
            textEditingController:
                occupationController,
            keyboard: TextInputType.text))
  ]),
  Row(children: [
    const Text("Income(Per Annum):"),
    Expanded(
        child: InputFieldStudentRegistration(
            textEditingController: IncomeController,
            keyboard: TextInputType.datetime))
  ])
]))),
Padding(
   padding: const EdgeInsets.only(left: 20, right: 20, bottom: 2),
   child: Container(
      width: double.infinity,
      decoration: BoxDecoration(
         borderRadius: BorderRadius.circular(20),
         color: Colors.teal[400]),
      child: Padding(
         padding: const EdgeInsets.all(12.0),
         child: Column(
            mainAxisAlignment: MainAxisAlignment.start,
            children: [
              // College Transport...
              const Text("Use College Transport"),
              FittedBox(
                 child: Row(
                    mainAxisAlignment: MainAxisAlignment.start,
                    children: [
                  const Text('Yes'),
                  Radio(
                     value: 'Yes',
                     groupValue: transport,
                     onChanged: (value) {
                       setState(() {
                         transport = value.toString();
                       });
                     }),
                  SizedBox(width: mediaQuery.size.width * 0.1),
                  const Text('No'),
                  Radio(
                     value: 'No',
                     groupValue: transport,
                     onChanged: (value) {
                       setState(() {
                         transport = value.toString();
                       });
                     }),
                 ])),
            ])))),

Padding(
   padding: const EdgeInsets.only(left: 20, right: 20, bottom: 2),
   child: Container(
      width: double.infinity,
      decoration: BoxDecoration(
         borderRadius: BorderRadius.circular(20),
         color: Colors.teal[400]),
      child: Column(children: [
        Padding(
           padding: const EdgeInsets.all(12),
child: Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      Row(children: [
        // address...
        const Text('Address:'),
        Expanded(
            child: InputFieldStudentRegistration(
                textEditingController:
                    AddressController,
                keyboard: TextInputType.text))
      ]),
      // phone number...
      Row(children: [
        const Text("Mobile NO:"),
        Expanded(
            child: InputFieldStudentRegistration(
                textEditingController: MobileController,
                keyboard: TextInputType.datetime))
      ]),
      // email id...
      Row(children: [
        const Text("E-mail ID:"),
        Expanded(
            child: InputFieldStudentRegistration(
                textEditingController: EmailController,
                keyboard: TextInputType.emailAddress))
      ]),
      // create password...
      Row(children: [
        const Text("Create Password:"),
        Expanded(
            child: InputFieldStudentRegistration(
                textEditingController:
                    PasswordController,
                keyboard: TextInputType.text))
      ]),
      const SizedBox(height: 20.0),
      // submit button...
      Padding(
          padding: const EdgeInsets.only(
              right: 20, bottom: 15, top: 5),
          child: Row(
              mainAxisAlignment: MainAxisAlignment.end,
              children: [
                ElevatedButton(
                    onPressed: () {
                      AddStudent();
                    },
                    style: ElevatedButton.styleFrom(
                        elevation: 20,
                                                  backgroundColor:
                                                      Colors.teal[600],
                                                  shadowColor: Colors.teal[600],
                                                  side: BorderSide(
                                                      color: Colors.teal.shade600,
                                                      width: 2,
                                                      style: BorderStyle.solid),
                                                  shape: RoundedRectangleBorder(
                                                      borderRadius:
                                                          BorderRadius.circular(
                                                              12.0)),
                                                  minimumSize:
                                                      const Size(130, 50)),
                                              child: !_isLoading
                                                  ? const Text('Submit')
                                                  : const CircularProgressIndicator(
                                                      color: Color.fromARGB(
                                                          255, 115, 162, 170)))
                                        ]))
                                ]))
                        ]))),
                const SizedBox(height: 20.0)
              ])));
    }
  }
```

9. TESTING

Testing refers to the process of evaluating a system, product, or application to ensure that it meets the specified requirements and functions as intended. It is an essential part of the software development lifecycle and is conducted to identify defects, errors, or inconsistencies in the system.

The main objectives of testing are to:

1. Identify defects: Testing aims to uncover errors, bugs, or issues in the system. By identifying and addressing these defects, the overall quality and reliability of the system can be improved.
2. Ensure functionality: Testing verifies that the system functions correctly according to the defined requirements and specifications. It ensures that all the intended features and functionalities are working as expected.
3. Validate requirements: Testing helps in validating whether the system meets the specified requirements and performs as intended. It ensures that the developed software aligns with the customer's needs and expectations.
4. Enhance user experience: Through testing, usability aspects of the system can be evaluated to ensure that it is user-friendly, intuitive, and provides a seamless experience to the end-users.
5. Increase reliability and performance: Testing helps in assessing the stability, reliability, and performance of the system under different conditions. It helps in identifying bottlenecks, performance issues, or vulnerabilities that may impact the system's functionality.

Testing can encompass various levels and types, including:

1. Unit Testing: Testing individual units or components of the software to ensure they work correctly.
2. Integration Testing: Testing the interaction and integration between different components or modules of the software.
3. System Testing: Testing the entire system as a whole to ensure it meets the specified requirements.
4. Acceptance Testing: Testing conducted to determine whether the system is ready for deployment and meets the customer's requirements.
5. Performance Testing: Testing to assess the performance and scalability of the system under different load conditions.
6. Security Testing: Testing to identify vulnerabilities and ensure the system is secure against potential threats.

Testing is an iterative process, and it involves designing test cases, executing tests, analyzing results, and making necessary modifications or improvements based on the findings. The goal is to ensure that the system functions reliably, meets user expectations, and delivers the desired outcomes.

➢ **Test Cases:-**

1. Test Case: Student Registration
   - Preconditions: System is accessible, valid user credentials provided.
   - Inputs: Student details (name, email, contact number), course selection.
   - Expected Results: Student successfully registered, student details stored in the system, course enrollment recorded.

2. Test Case: Course Enrollment
   - Preconditions: Student is registered and logged into the system, available courses are populated.
   - Inputs: Select a course from the available options.

- Expected Results: Student is successfully enrolled in the selected course, enrollment details updated in the system.

3. Test Case: Grade Calculation
    - Preconditions: Student is enrolled in a course, instructor has entered grades.
    - Inputs: Student ID, course ID.
    - Expected Results: System calculates the final grade based on the entered grades, grade is recorded for the student in the system.

4. Test Case: Generate Student Report
    - Preconditions: Student is registered, enrolled in courses, grades are entered.
    - Inputs: Student ID.
    - Expected Results: System generates a report containing student details, enrolled courses, and respective grades.

5. Test Case: Faculty Management
    - Preconditions: Faculty member is registered and logged into the system.
    - Inputs: Add/update faculty details, assign courses to faculty.
    - Expected Results: Faculty details are stored in the system, courses assigned to faculty are reflected correctly.

6. Test Case: Financial Management
    - Preconditions: System has financial management functionality enabled.
    - Inputs: Add/update student fee payment details, generate fee receipts.
    - Expected Results: Student fee payment details are stored accurately, fee receipts are generated correctly.

7. Test Case: Security and Access Control
    - Preconditions: Different user roles exist in the system (admin, faculty, student), permissions are assigned.
    - Inputs: Login with different user roles, attempt to access restricted areas.
    - Expected Results: User with the appropriate role can access permitted areas, unauthorized access is denied.

8. Test Case: Performance and Scalability
    - Preconditions: The system is under a load of concurrent users.
    - Inputs: Simulate multiple users performing actions simultaneously.
    - Expected Results: The system maintains acceptable response times, handles concurrent requests without performance degradation.

9. Test Case: Integration with External Systems
    - Preconditions: The system is integrated with external systems (e.g., library management system, student information system).
    - Inputs: Perform actions that involve interactions with external systems.
    - Expected Results: Data exchange and synchronization with external systems occur accurately.

10. Test Case: Usability Testing
    - Preconditions: Users are not familiar with the system.
    - Inputs: Perform common tasks like registration, enrollment, grade viewing.
    - Expected Results: Users can easily navigate the system, understand instructions, and complete tasks without confusion.

10. MAINTENANCE

Maintenance refers to the activities and processes undertaken to keep a system, product, equipment, or infrastructure in a good working condition or to restore it to an operational state if issues or defects arise. It involves regular upkeep, repair, servicing, and monitoring to ensure the system continues to function effectively and efficiently over its intended lifespan.

Maintenance can be categorized into different types:

1. Corrective Maintenance: This type of maintenance involves fixing issues or defects that are discovered during the operation of the system. It aims to restore the system to its desired functionality and resolve any problems that may arise.
2. Preventive Maintenance: Also known as planned or scheduled maintenance, preventive maintenance involves proactive measures taken to prevent potential issues or breakdowns. It includes routine inspections, servicing, cleaning, and replacing components before they fail, with the goal of reducing the likelihood of unexpected failures.
3. Adaptive Maintenance: Adaptive maintenance refers to making modifications or adjustments to a system to accommodate changes in the environment, technology, or user requirements. It ensures that the system remains compatible, relevant, and functional in evolving circumstances.
4. Perfective Maintenance: Perfective maintenance focuses on enhancing or improving the system's performance, efficiency, or usability. It involves making changes to the system's code, design, or functionality to optimize its operation or to add new features that enhance its value.

Maintenance activities can include tasks such as software updates, hardware repairs, system backups, security patches, performance optimization, data cleaning, and regular inspections. It is crucial to have a well-defined maintenance plan or schedule to ensure the system's reliability, longevity, and continued operational effectiveness.

Effective maintenance practices help minimize downtime, extend the lifespan of systems, prevent major failures or breakdowns, and ensure optimal performance. It is an ongoing process that requires attention, resources, and expertise to keep the system in a reliable and functional state.

Maintaining a college management system is essential to ensure its smooth and efficient operation over time. Regular maintenance activities help to address issues, improve performance, and enhance the overall user experience. Here are some key aspects to consider when it comes to the maintenance of a college management system:

1. <u>Bug Fixes and Issue Resolution</u>: Monitor the system for any reported bugs, errors, or issues. Prioritize and resolve them promptly to minimize disruption to system functionality. This includes addressing user-reported problems, conducting root cause analysis, and applying necessary patches or updates.

2. <u>System Upgrades and Enhancements</u>: Stay updated with the latest technology trends and advancements relevant to college management systems. Regularly evaluate new features, functionalities, and modules that can enhance the system's performance, security, or user

experience. Plan and execute system upgrades or enhancements in a structured manner, ensuring proper testing and backup procedures.

3. Security Updates and Vulnerability Patching: Keep the college management system secure by promptly applying security updates and patches. Regularly monitor security advisories and address any identified vulnerabilities promptly to prevent data breaches, unauthorized access, or other security incidents. Implement robust authentication mechanisms, encryption protocols, and access controls to safeguard sensitive information.

4. Database Maintenance and Optimization: Regularly maintain and optimize the database used by the system. Perform routine backups, database integrity checks, and performance optimizations to ensure data reliability, efficient query execution, and fast response times. Regularly purge outdated or irrelevant data to optimize storage and improve system performance.

5. User Support and Training: Provide ongoing support and training to system users, including college administrators, faculty, and students. Offer resources such as user guides, FAQs, and training sessions to assist users in understanding and utilizing the system effectively. Address user queries, provide timely assistance, and encourage feedback to continuously improve user satisfaction.

6. Monitoring and Performance Tuning: Implement monitoring tools and procedures to track system performance metrics, such as response time, resource utilization, and error rates. Proactively identify performance bottlenecks and fine-tune the system configuration, hardware, or software components as needed. Regularly review system logs and performance data to identify any potential issues or areas for improvement.

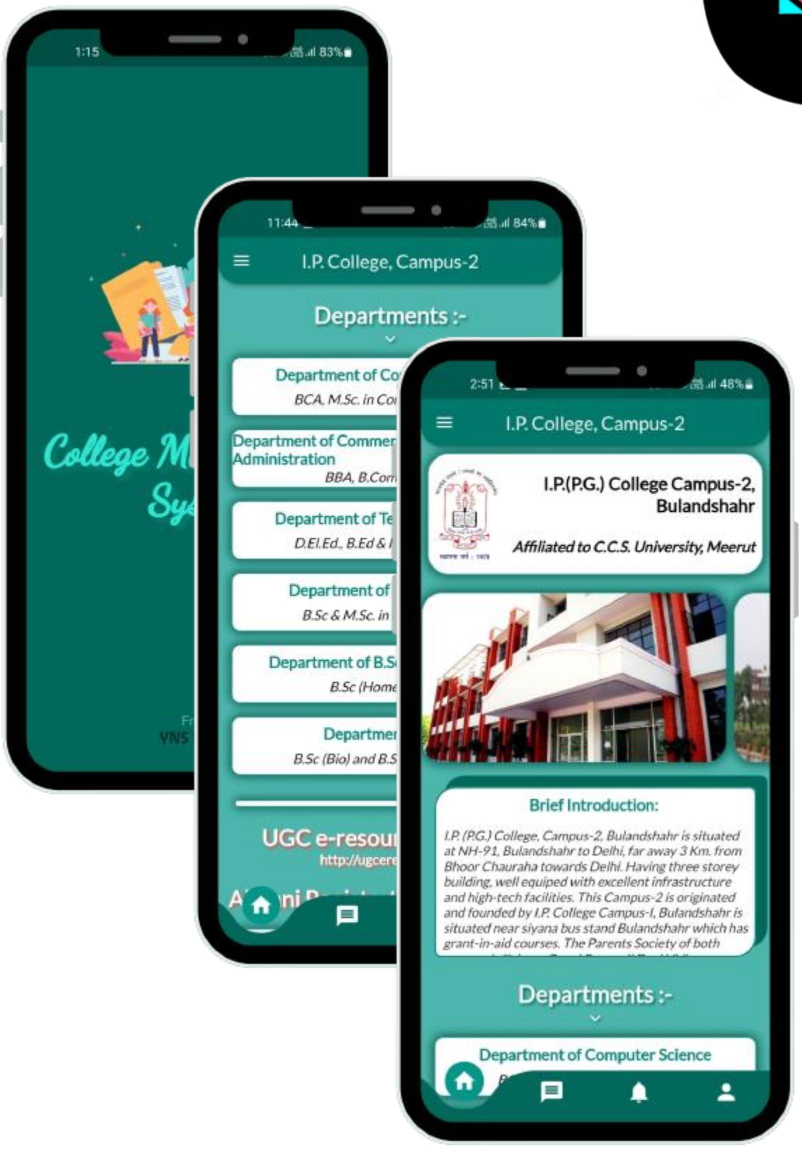
7. Compliance with Regulations and Standards: Ensure that the college management system complies with relevant regulations and standards, such as data privacy laws (e.g., GDPR, CCPA), accessibility guidelines (e.g., WCAG), and educational sector regulations. Regularly review and update system policies, data protection measures, and consent mechanisms to maintain compliance and mitigate legal and reputational risks.

8. User Feedback and Continuous Improvement: Encourage users to provide feedback on their experience with the college management system. Actively seek user input through surveys, focus groups, or feedback forms. Analyze and incorporate user feedback into the system's roadmap, prioritizing improvements that enhance usability, address pain points, and align with user needs and expectations.

9. Documentation and Knowledge Management: Maintain up-to-date documentation of the college management system, including user manuals, system architecture, configurations, and integration details. Document known issues, troubleshooting steps, and resolutions for efficient problem-solving. Promote knowledge sharing among the maintenance team to ensure smooth handovers and enable continuous improvement.

10. Disaster Recovery and Business Continuity: Implement robust disaster recovery and business continuity plans to mitigate the impact of potential system failures, natural disasters, or data breaches. Regularly test and update these plans to ensure they are effective and aligned with the evolving needs of the college

11. SCREENSHOTS



I.P.(P.G.) College Campus-2, Bulandshahr

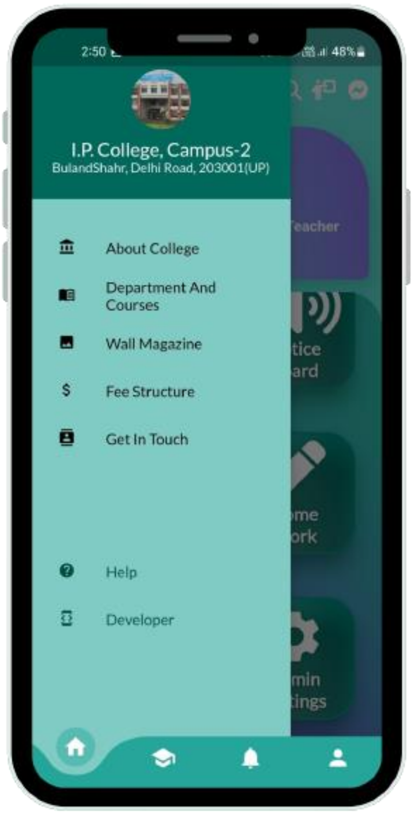
Affiliated to C.C.S. University, Meerut

Brief Introduction:

I.P. (P.G.) College, Campus-2, Bulandshahr is situated at NH-91, Bulandshahr to Delhi, far away 3 Km. from Bhoor Chauraha towards Delhi. Having three storey building, well equiped with excellent infrastructure and high-tech facilities. This Campus-2 is originated and founded by I.P. College Campus-I, Bulandshahr is situated near siyana bus stand Bulandshahr which has grant-in-aid courses. The Parents Society of both

Departments :-

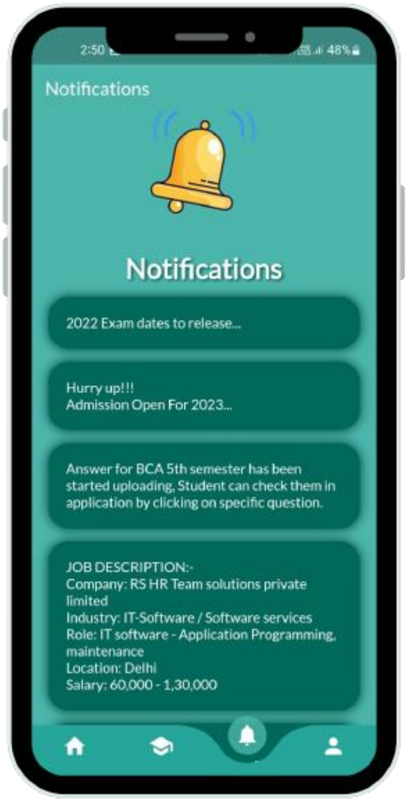
Department of Computer Science


2:50 48%

I.P. College, Campus-2
BulandShahr, Delhi Road, 203001(UP)

About College
Department And Courses
Wall Magazine
Fee Structure
Get In Touch

Help
Developer


2:50 48%

Notifications

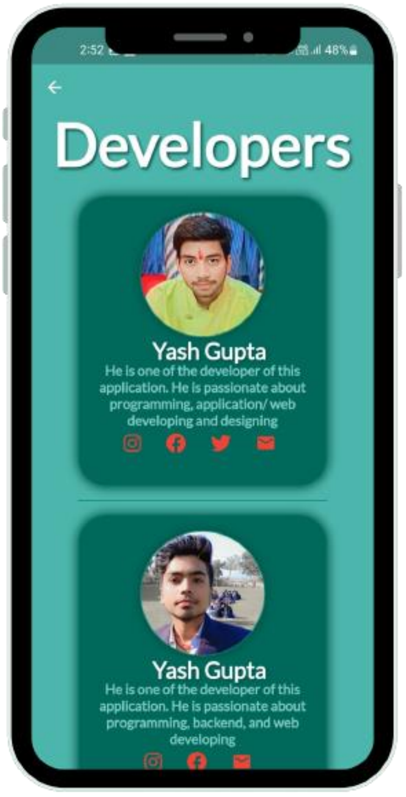
Notifications

2022 Exam dates to release...

Hurry up!!!
Admission Open For 2023...

Answer for BCA 5th semester has been started uploading, Student can check them in application by clicking on specific question.

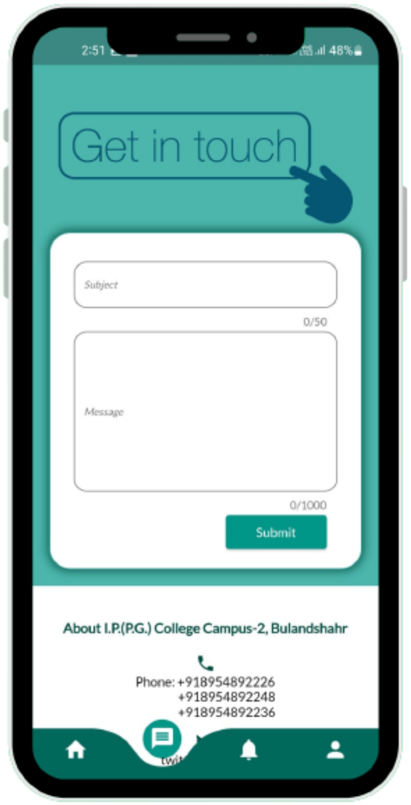
JOB DESCRIPTION:-
Company: RS HR Team solutions private limited
Industry: IT-Software / Software services
Role: IT software - Application Programming, maintenance
Location: Delhi
Salary: 60,000 - 1,30,000


2:52 48%

Developers

Yash Gupta
He is one of the developer of this application. He is passionate about programming, application/ web developing and designing

Yash Gupta
He is one of the developer of this application. He is passionate about programming, backend, and web developing

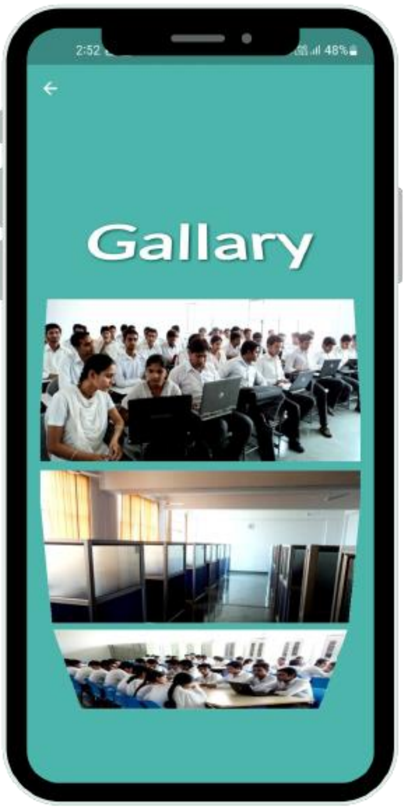

2:51 48%

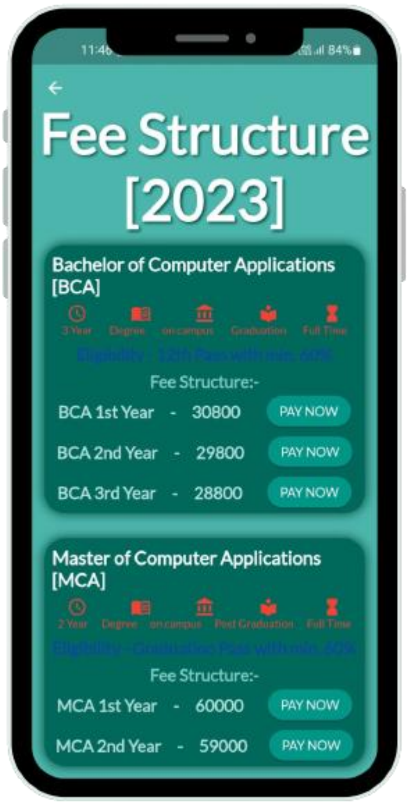
Get in touch

Subject
0/50
Message
0/1000
Submit

About I.P.(P.G.) College Campus-2, Bulandshahr

Phone: +918954892226
+918954892248
+918954892236




Fee Structure [2023]

Bachelor of Computer Applications [BCA]

Eligibility - 12th Pass with min. 40%

Fee Structure:-

BCA 1st Year - 30800 PAY NOW

BCA 2nd Year - 29800 PAY NOW

BCA 3rd Year - 28800 PAY NOW

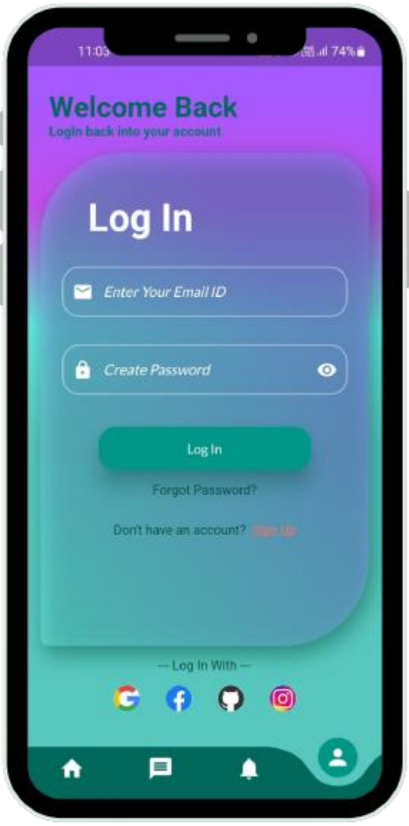
Master of Computer Applications [MCA]

Eligibility - Graduation Pass with min. 40%

Fee Structure:-

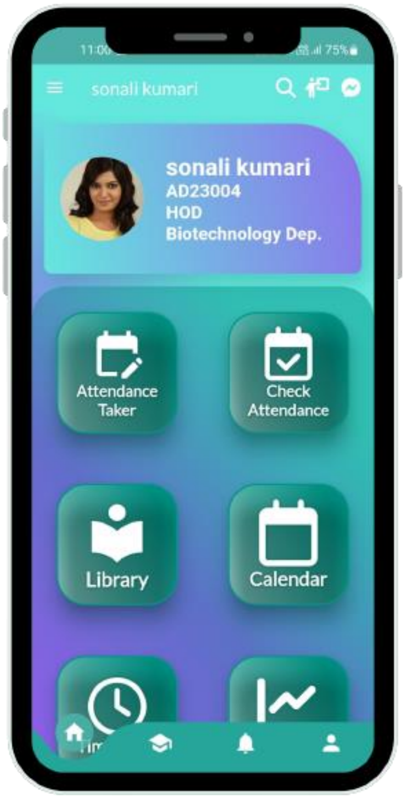
MCA 1st Year - 60000 PAY NOW

MCA 2nd Year - 59000 PAY NOW

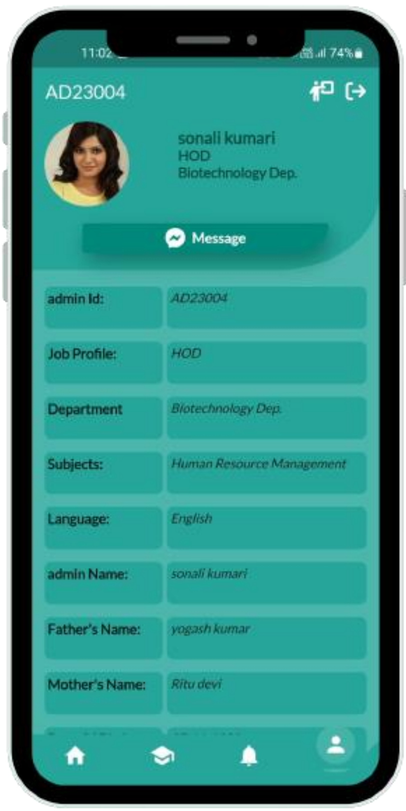

Welcome Back

Login back into your account

Log In



sonali kumari
AD23004
HOD
Biotechnology Dep.

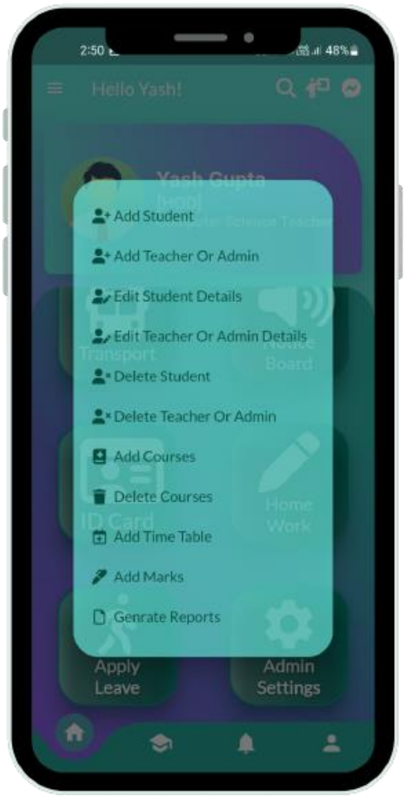


AD23004

sonali kumari
HOD
Biotechnology Dep.

Message

admin Id:	AD23004
Job Profile:	HOD
Department	Biotechnology Dep.
Subjects:	Human Resource Management
Language:	English
admin Name:	sonali kumari
Father's Name:	yogash kumar
Mother's Name:	Ritu devi

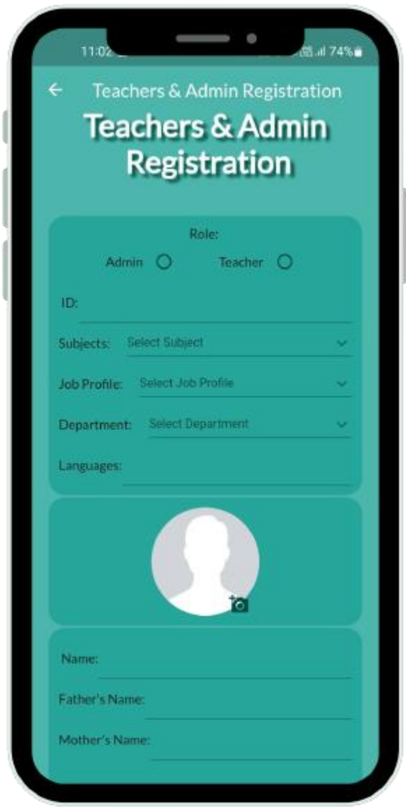


Hello Yash!

Yash Gupta

- Add Student
- Add Teacher Or Admin
- Edit Student Details
- Edit Teacher Or Admin Details
- Delete Student
- Delete Teacher Or Admin
- Add Courses
- Delete Courses
- Add Time Table
- Add Marks
- Genrate Reports

Transport
Notice Board
ID Card
Home Work
Apply Leave
Admin Settings



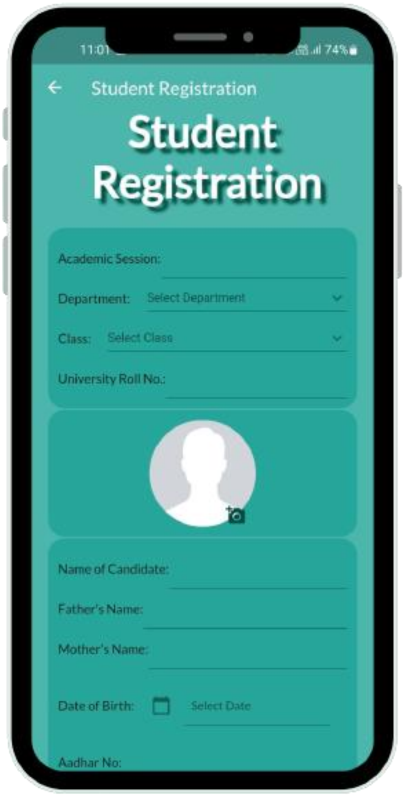
Teachers & Admin Registration

Teachers & Admin Registration

Role:
Admin ○ Teacher ○

ID:
Subjects: Select Subject
Job Profile: Select Job Profile
Department: Select Department
Languages:

Name:
Father's Name:
Mother's Name:

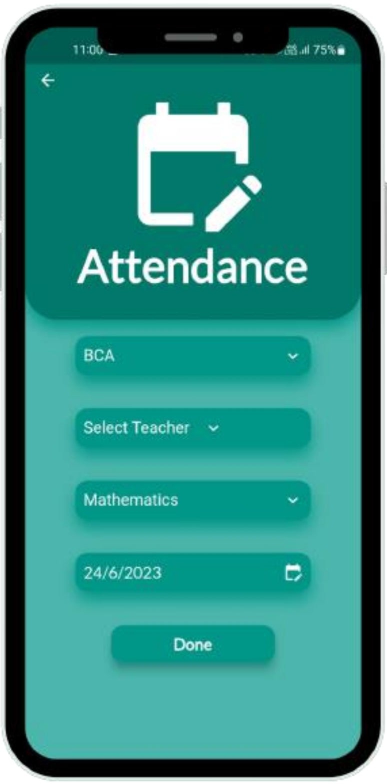


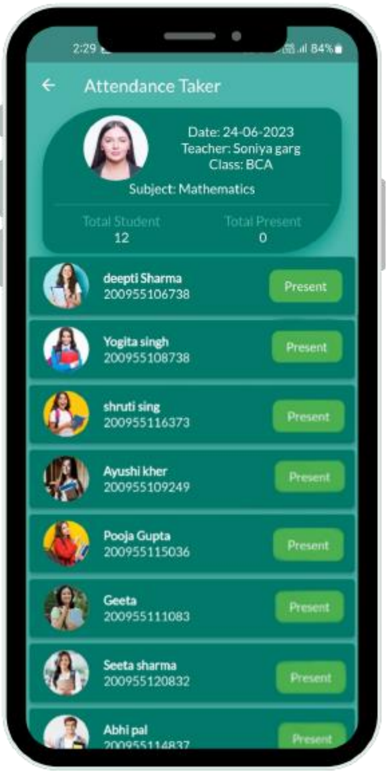
Student Registration

Student Registration

Academic Session:
Department: Select Department
Class: Select Class
University Roll No.:

Name of Candidate:
Father's Name:
Mother's Name:
Date of Birth: Select Date
Aadhar No:

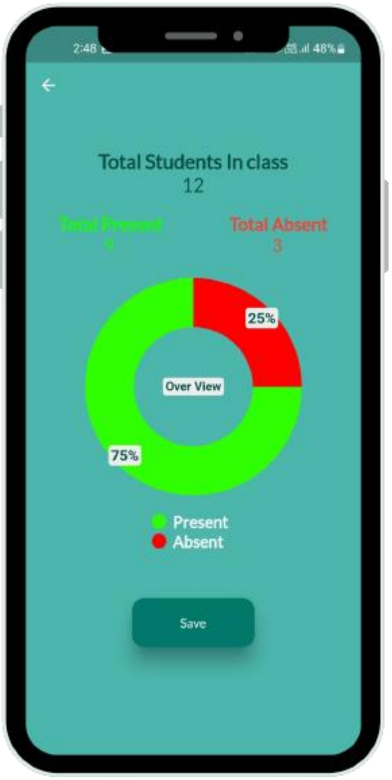



Attendance Taker

Date: 24-06-2023
Teacher: Soniya garg
Class: BCA
Subject: Mathematics

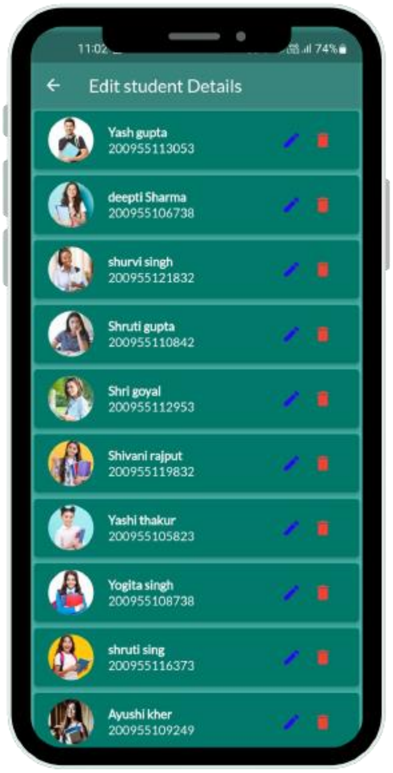
Total Student 12
Total Present 0

deepti Sharma 200955106738 Present
Yogita singh 200955108738 Present
shruti sing 200955116373 Present
Ayushi kher 200955109249 Present
Pooja Gupta 200955115036 Present
Geeta 200955111083 Present
Seeta sharma 200955120832 Present
Abhi pal 200955114837 Present

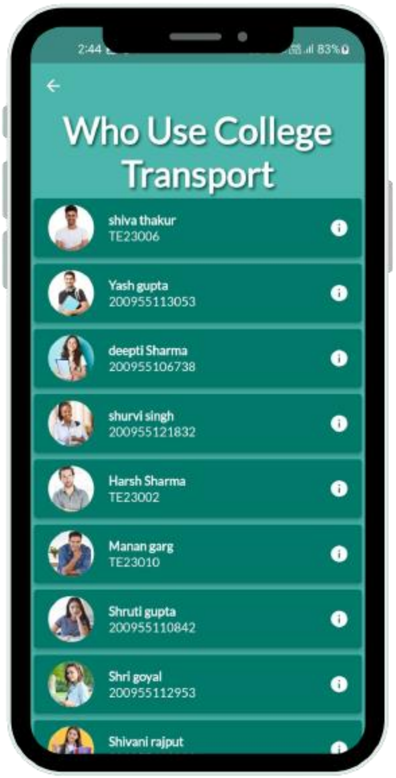

Total Students In class
12
Total Present
9
Total Absent
3

25%
Over View
75%
🟩 Present
🟥 Absent

Save

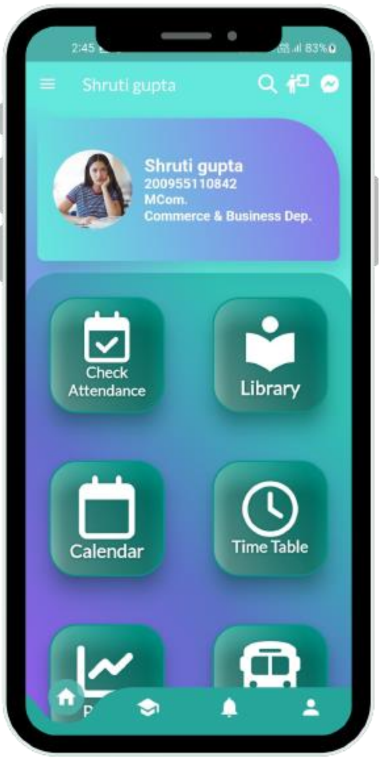

Edit student Details

Yash gupta 200955113053
deepti Sharma 200955106738
shurvi singh 200955121832
Shruti gupta 200955110842
Shri goyal 200955112953
Shivani rajput 200955119832
Yashi thakur 200955105823
Yogita singh 200955108738
shruti sing 200955116373
Ayushi kher 200955109249


2:44 83%

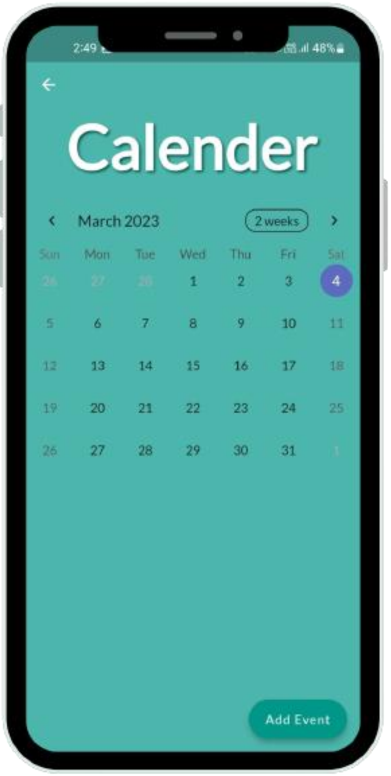
Who Use College Transport

shiva thakur	TE23006
Yash gupta	200955113053
deepti Sharma	200955106738
shurvi singh	200955121832
Harsh Sharma	TE23002
Manan garg	TE23010
Shruti gupta	200955110842
Shri goyal	200955112953
Shivani rajput	


2:45 83%

Shruti gupta

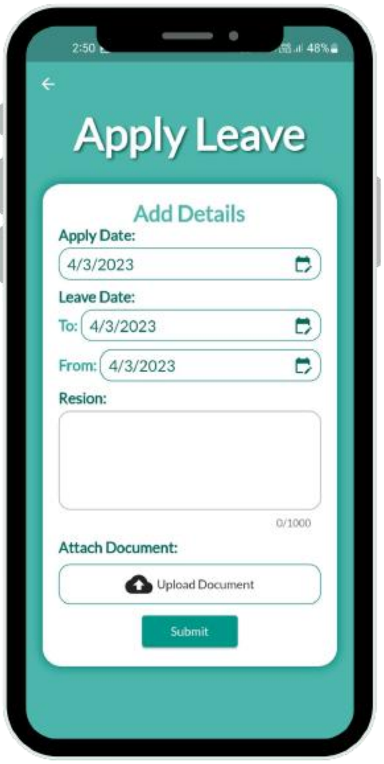
Shruti gupta
200955110842
MCom.
Commerce & Business Dep.


2:49 48%

Calender

March 2023 2 weeks

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1


2:50 48%

Apply Leave

Add Details

Apply Date: 4/3/2023

Leave Date:

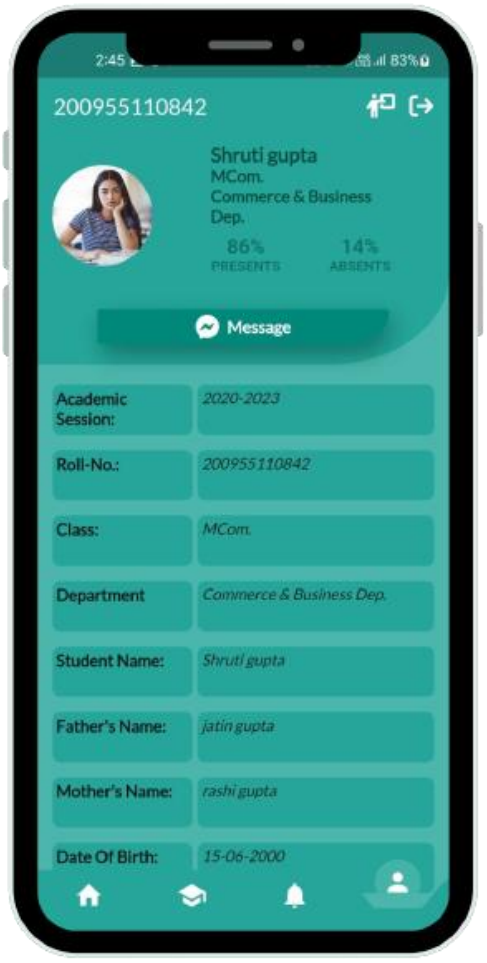
To: 4/3/2023

From: 4/3/2023

Resion:

0/1000

Attach Document:



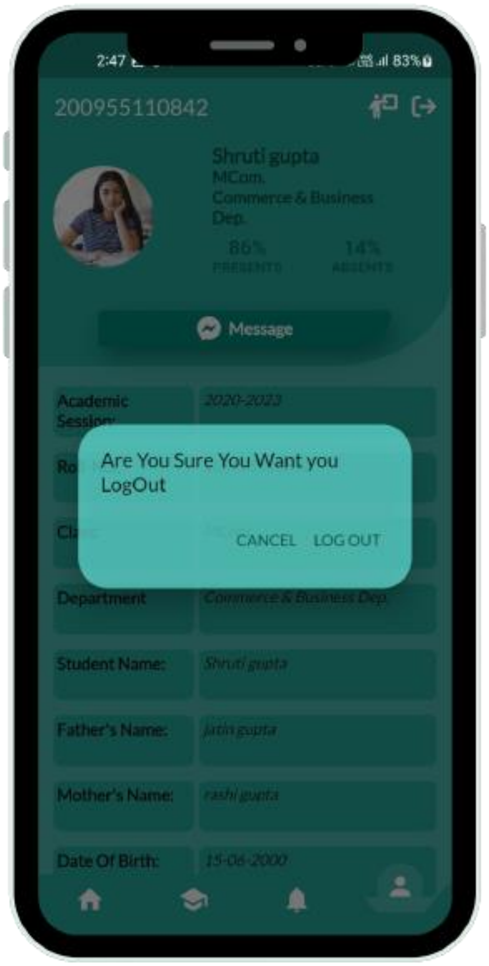
200955110842

Shruti gupta
MCom.
Commerce & Business Dep.

86% PRESENTS
14% ABSENTS

Message

Academic Session:	2020-2023
Roll-No.:	200955110842
Class:	MCom.
Department	Commerce & Business Dep.
Student Name:	Shruti gupta
Father's Name:	jatin gupta
Mother's Name:	rashi gupta
Date Of Birth:	15-06-2000



200955110842

Shruti gupta
MCom.
Commerce & Business Dep.

86% PRESENTS
14% ABSENTS

Message

Are You Sure You Want you LogOut

CANCEL LOG OUT

Academic Session:	2020-2023
Department	Commerce & Business Dep.
Student Name:	Shruti gupta
Father's Name:	jatin gupta
Mother's Name:	rashi gupta
Date Of Birth:	15-06-2000

12. LIMITATIONS / FUTURE SCOPE

- **Limitations**: -

The part of the system can be implemented using the current technology although some modifications had to be done at various places. At various places some alterations with the prototypes and functionalities would be done in order to work out the cost constraints and to cope with the scheduling constraints.

- In this application search is limited to String or by number. Cannot do search by photo and figureprints.
- If the main server has any problem, user has to wait till it solve to user application.

- **Future Scope: -**

As the current system is expected to add more functionality and dependency according to requirement changes and technology, proper coding standards and working platform have been keptin mind to produce a quality product. The future extension plan of the project is summarized as follows:

- Can be used to automate administrative tasks and analyze data to provide insights and recommendations for improvements.
- Can be used to improve security and transparency in the management of academic records and credentials.
- Its cloud computing can provide a more flexible and scalable infrastructure for the college management system. It can help to reduce costs and improve the reliability and availability of the system.
- This application can provide a more convenient and accessible way for students, faculty, and administrators to access the college management system. This can include features such as push notifications, reminders, and real-time updates.

13. REFERENCES/ BIBLIOGRAPHY

➢ Websites:-

- Flutter - Build apps for any screen
- Flutter Tutorial - GeeksforGeeks
- Stack Overflow - Where Developers Learn, Share, & Build Careers

➢ YouTube:-

- Flutter - YouTube
- What is Flutter? & How it is Better than it's Counterparts? - Cross Platform | Full Tutorial - YouTube
- Master Flutter in Just 8 Hours | Full Course Hindi @HindiCodepur - YouTube
- HeyFlutter. com - YouTube