PROJECT REPORT

On

University Management App (Campus Connect)

Submitted in partial fulfillment of the requirement for the Course BEE (22CS026) of

COMPUTER SCIENCE AND ENGINEERING B.E.

Batch-2022 in Sep -2024

Under the Guidance of :

Ms. Preenu Mittan

Submitted By:

Bhuvesh Mittal

2210991450

Daanushi Sharma

2210991469

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING CHITKARA UNIVERSITY PUNJAB

CERTIFICATE

This is to be certified that the project entitled “Campus Connect”, a university management app has been submitted for the Bachelor of Computer Science Engineering at Chitkara University, Punjab during the academic semester July 2024- December 2024 is a bona fide piece of project work carried out by “Bhuvesh Mittal (2210991450) and Daanushi Sharma (2210991469)” towards the partial fulfillment for the award of the course Integrated Project (CS 203) under the guidance of “Ms. Preenu Mittan” and supervision.

Sign. of Project Guide :

Ms. Preenu Mittan

CANDIDATE'S DECLARATION

We , Bhuvesh Mittal (2210991450) and Daanushi Sharma (2210991469) G-18, B.E.-2022 of the Chitkara University, Punjab hereby declare that the Integrated Project Report entitled "Campus Connect" is an original work and data provided in the study is authentic to the best of our knowledge. This report has not been submitted to any other Institute for the award of any other course.

Bhuvesh Mittal	Daanushi Sharma
2210991450	2210991469

Place: Chitkara University

Date: September 26, 2024

ACKNOWLEDGMENT

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behavior and acts during the course of study.

We express our sincere gratitude to all for providing me an opportunity to undergo Integrated Project as the part of the curriculum.

We are thankful to "Ms. Preenu Mittan" for her support, cooperation, and motivation provided to us during the training for constant inspiration, presence and blessings.

We also extend our sincere appreciation to “Ms. Preenu Mittan” who provided his valuable suggestions and precious time in accomplishing our integrated project report.

Lastly, we would like to thank the almighty and our parents for their moral support and friends with whom we shared our day-to day experience and received lots of suggestions that improve our quality of work.

Bhuvesh Mittal **Daanushi Sharma**

2210991450 **2210991469**

CHITKARA UNIVERSITY

Table of Contents

1. Abstract

This project focuses on developing a comprehensive university management system using Node.js, Express.js, and HTML/CSS. The system is designed to streamline and integrate the diverse functions necessary for efficient university operations, addressing the needs of students, teachers, clubs, and administrators. It features four distinct dashboards:

1. Student Dashboard: Provides students with access to their profiles, attendance records, timetables, report cards, achievements, and current modules. This ensures students can easily track their academic progress and manage their schedules.
2. Teacher Dashboard: Enables teachers to view their profiles, timetables, and student lists. It allows them to upload attendance and marks, and access a shared noticeboard, facilitating effective communication and record-keeping.
3. Club Dashboard: Displays information about various clubs, including club members and upcoming events, promoting engagement and organization within the university's extracurricular activities.
4. Admin Dashboard: Offers administrative staff the capability to manage notices, and teacher and student information, ensuring smooth operation and dissemination of information across the university.

While the initial implementation focuses on core functionalities, future developments will incorporate MongoDB to enhance data management and scalability. The system is designed to improve efficiency by providing a centralized platform that integrates multiple administrative and academic functions, ensuring real-time access to information and enhancing overall university management. The use of modern technologies and a user-friendly interface aims to streamline processes and improve communication among all stakeholders.

2. Introduction

2.1 Background

The rapid advancement of technology has significantly transformed various sectors, including education. The integration of digital tools in education has led to the development of university management systems (UMS), which streamline administrative tasks, enhance communication between stakeholders, and improve the overall efficiency of academic institutions. In the context of higher education, a UMS can play a pivotal role in managing the extensive data and processes associated with students, teachers, clubs, and administrative functions.

University management systems are designed to handle a multitude of tasks, such as managing student information, teacher activities, club activities, and administrative functions, all through a centralized platform. This system not only facilitates better communication and information sharing but also reduces the administrative burden, allowing educators and administrators to focus more on core educational activities.

The importance of such systems has grown in recent years due to the increasing complexity of managing large educational institutions. With the proliferation of online learning and the need for real-time data access, a robust UMS is essential for maintaining the quality of education and ensuring smooth operations within the university. This background sets the stage for the development of a comprehensive university management app, which will serve the needs of students, teachers, clubs, and administrators.

2.2 Problem Statement

The traditional methods of managing university operations often involve manual processes, paper-based records, and isolated systems that do not communicate effectively with each other. This fragmentation leads to inefficiencies, delays in information dissemination, and increased workload for university staff. Moreover, students and teachers may find it challenging to access critical information, such as attendance records, timetables, and academic performance, in a timely manner.

The key problem addressed in this research is the lack of an integrated platform that can manage the diverse needs of a university, including student information, teacher activities, club management, and administrative functions. The absence of such system results in fragmented data, poor communication, and inefficiencies that hinder the smooth functioning of the university.

The specific research questions guiding this study are:

1. **Descriptive:** A university management system can be designed to integrate and streamline the diverse functions of students, teachers, clubs, and administrators by providing tailored dashboards for each user group. This ensures seamless communication and coordination across all university activities, from academic management to extracurricular involvement.
2. **Normative:** A university management system should include features such as personalized student and teacher dashboards, a comprehensive club management interface, and an administrative panel for managing notices and records. These features are essential to meet the needs of all stakeholders efficiently, ensuring that the system supports the university's operations and enhances overall productivity.

These research questions are closely tied to the methodology, which involves designing and developing a university management app using Node.js and EJS. The app will feature four distinct dashboards, each tailored to the specific needs of students, teachers, clubs, and administrators, ensuring a comprehensive and user-friendly system.

2.3 Research Questions

The introduction of a university management system raises several questions that are crucial to its successful implementation and adoption. These questions help to define the scope of the system and guide the development process. The research questions include:

1. **How can a centralized platform improve the efficiency of university management processes?**
   A centralized platform streamlines university management by consolidating key functions—student information, teacher activities, club operations, and administrative tasks—into a single, integrated system. This reduces redundancies, ensures real-time updates, and enhances communication between students, teachers, and administrators. For example, when teachers upload attendance or marks, students can immediately access this information, and administrators can efficiently manage records and notices. This integration minimizes errors and improves the overall efficiency of university operations.

2. **What are the essential features of a university management system that caters to the needs of students, teachers, clubs, and administrators?**
   A university management system should include features tailored to each user group:
   a. **Student Dashboard:** Access to personal profiles, attendance records, timetables, report cards, achievements, and current modules.
   b. **Teacher Dashboard:** Tools to manage personal profiles, timetables, student lists, attendance, and mark uploads, along with a shared noticeboard.
   c. **Club Dashboard:** Information on club members, events, and other club-related activities.
   d. **Admin Dashboard:** Capabilities to upload and manage student and teacher information, and publish notices on the noticeboards.

   These features ensure that each group can efficiently perform their respective tasks within the system, enhancing overall university management.

2.4 Previous Research and Methodology

The development of university management systems has been the subject of extensive research, with various approaches being proposed and implemented. Previous studies have highlighted the importance of user-centered design, the integration of modern technologies such as cloud computing and mobile applications, and the need for scalability and flexibility in such systems.

In this project, the methodology involves the use of Node.js and EJS to develop a university management app that provides tailored dashboards for students, teachers, clubs, and administrators. The system will be designed to be user-friendly, with an emphasis on ease of access to information, efficient management of data, and the ability to handle the unique needs of each user group.

3. Software and Hardware Requirement Specification

3.1 Methods

The development of the university management system involves utilizing agile methodology. Agile allows for iterative progress through small, manageable sections of the project, providing flexibility and continuous feedback. The main methods used include:

- **Requirement Gathering:** Engaging with stakeholders (students, teachers, admins, and club members) to gather requirements for each dashboard.
- **Design:** Creating wireframes and design layouts for different dashboards.
- **Development:** Coding the application using Node.js, Express.js for the backend, and HTML/CSS for the frontend.
- **Testing:** Continuous integration and testing using both manual and automated methods to ensure each feature works as intended.
- **Deployment:** Deploying the application on a server or cloud platform, ensuring accessibility and scalability.

3.2 Programming/Working Environment

The app is being developed in a full-stack JavaScript environment:

- **Node.js:** Server-side runtime environment that allows the use of JavaScript for backend development.
- **Express.js:** Web application framework for Node.js, providing robust features for building web and mobile applications.
- **HTML/CSS:** Frontend technologies used for structuring and styling the web pages.
- **EJS (Embedded JavaScript):** Templating engine used for generating HTML with embedded JavaScript code, allowing dynamic content rendering.
- **Development Tools:**
    - **Visual Studio Code:** Code editor with extensions for Node.js and web development.
    - **Git:** Version control system for managing code changes and collaboration.
    - **Postman:** API testing tool for validating the backend services.

3.3 Requirements to Run the Application

To run the university management system, the following software and hardware requirements are necessary:

- **Software Requirements:**
    - **Node.js:** Version 14.x or higher, along with npm (Node Package Manager).
    - **Express.js:** Installed via npm as part of the Node.js ecosystem.
    - **Browser:** Any modern web browser (Chrome, Firefox, Edge, etc.) to access the dashboards.
    - **Database:** A NoSQL database like MongoDB or a relational database like MySQL for storing and managing user data.
    - **Operating System:** The application can be run on any OS that supports Node.js, including Windows, Linux, and macOS

- **Hardware Requirements:**
    - **Server:** A server with at least 2 GB RAM and 20 GB disk space for hosting the application.
    - **Client Machines:** Any standard computer or mobile device with internet access to interact with the web application.
    - **Network:** Stable internet connection for accessing and using the application, both for client and server-side interactions.

4. Database Analysis, Design, and Implementation

4.1 Database Analysis

In the context of the university management system, the database is crucial for managing various data related to students, teachers, clubs, and administrators. The analysis focuses on understanding the types of data, their relationships, and the requirements for future database implementation using MongoDB.

- **Data Requirements:**
  - **Student Data:** Includes personal and academic details such as name, date of birth, email, mobile number, gender, roll number, batch, program, program duration, hostel facility, receipt date, and receipt number. Also includes dynamic records like attendance, marks, achievements, and current modules.
  - **Teacher Data:** Contains personal information (name, email, mobile number) and professional details including subjects taught, timetable, attendance records, marks records, and contributions to the noticeboard.
  - **Club Data:** Encompasses club information, member details, events, and announcements.
  - **Administrative Data:** Includes system configurations, noticeboard updates, and management of student and teacher profiles.
- **Data Relationships:**
  - **Students and Teachers:** Manage many-to-many relationships regarding attendance and marks.
  - **Clubs and Members:** Establish a one-to-many relationship between clubs and their members.
  - **Administrators and Users:** Administrators manage notices and user profiles, requiring robust data management and access control.

4.2 Database Design

The design phase involves creating a conceptual schema that outlines how data will be stored and related in MongoDB. While specific implementation details will be addressed later, the initial design considerations include:

- **Collections:**
  - **Students Collection:** Will store detailed student profiles, academic records, and dynamic information like attendance and marks.
  - **Teachers Collection:** Will include teacher profiles, subject details, and records of attendance and marks.
  - **Clubs Collection:** Will document club details, member lists, events, and announcements.
  - **Administrators Collection:** Will manage administrator profiles, roles, and system notices.
- **Indexes:**
  - Indexes will be created on key fields to enhance query performance and ensure efficient data retrieval.

4.3 Database Implementation

The implementation details for MongoDB will be addressed in future stages of the project. Key steps include:

- **MongoDB Setup:** Installation and configuration of MongoDB on the server, ensuring compatibility with the application.
- **Schema Definition:** Creation of collections and indexes based on the design schema.
- **Data Integration:** Migration of existing data into MongoDB collections, involving data transformation and cleaning as needed.
- **Data Access Layer:** Development of a data access layer using Mongoose for interaction with MongoDB.
- **Testing and Optimization:** Conducting tests to ensure database performance and making necessary optimizations.

5. Program's Structure Analyzing and GUI Constructing (Project Snapshots)

5.1 Program Structure Analysis

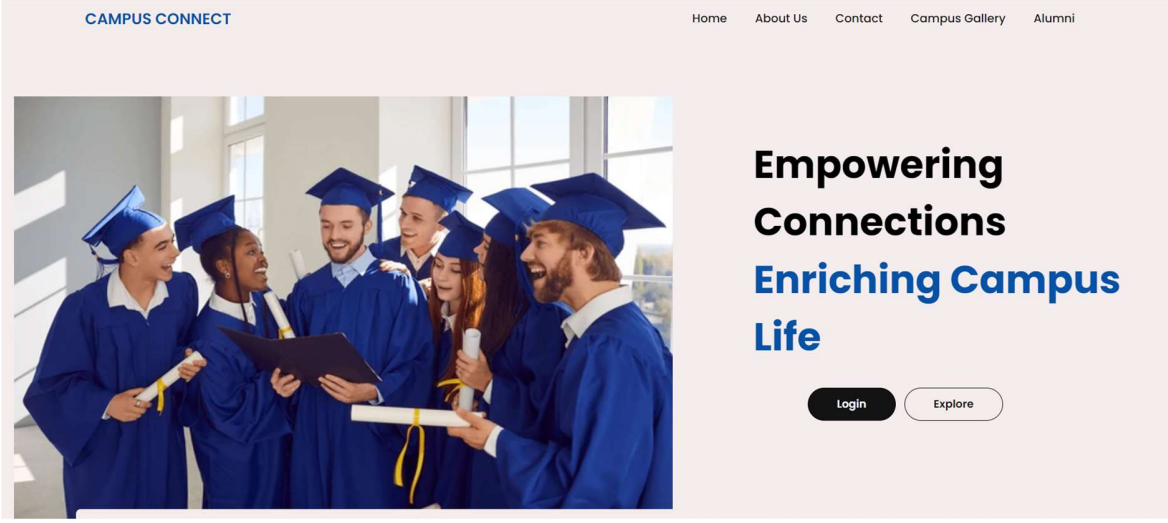
The university management app is structured into multiple interconnected modules, each serving a distinct purpose to streamline university operations. Below is an analysis of the key components of the program's structure:

1. **Student Dashboard:**
    - **Profile:** Displays student information including personal details and academic records.
    - **Report Card**: Displays notice board showing recent and important notices.
    - **Attendance Tracking:** Shows attendance data uploaded by teachers.
    - **Timetable:** Provides the student's class schedule.
    - **Report Card:** Presents academic performance in a structured format.
    - **Quiz:** Displays quizzes uploaded by teachers for students.
    - **Current Modules:** Shows the current subjects or modules the student is enrolled in.
2. **Teacher Dashboard:**
    - **Profile:** Displays teacher's personal and professional details.
    - **Timetable:** Provides the teacher's class schedule.
    - **Student:** Allows viewing students assigned to the teacher.
    - **Attendance Upload:** Enables teachers to upload and update attendance records.
    - **Marks Upload:** Facilitates the entry of student marks for various subjects.
    - **Noticeboard:** A shared page for posting notices that are visible to both students and teachers.
3. **Club Dashboard:**
    - **About club:** Displays details about the clubs.
    - **Club Members:** Shows the list of members associated with each club.
    - **Notice board:** A page where club members and heads can upload and view notices for students and teachers.
    - **Club Gallery:** Displays pictures of events and activities conducted by the club.

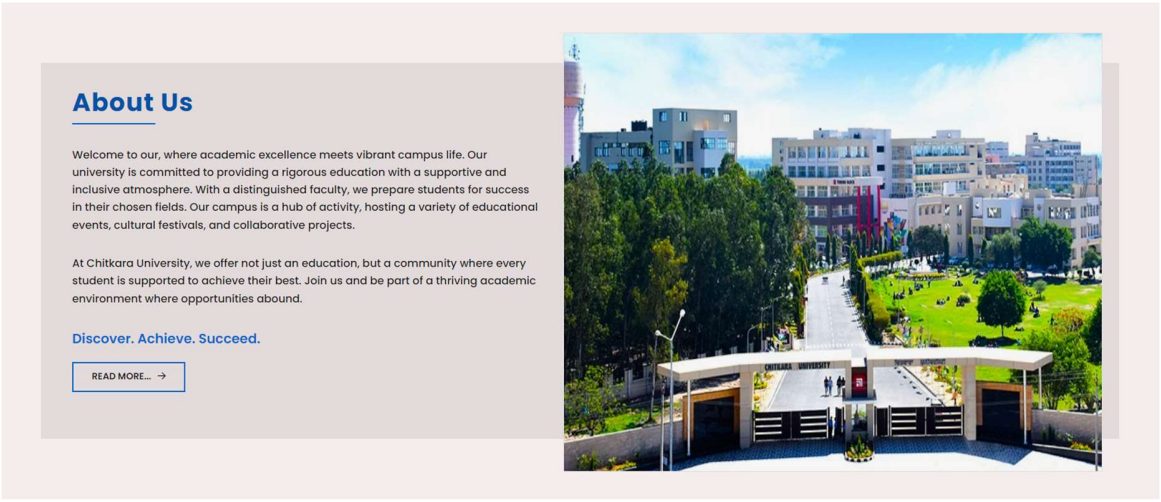
4. **Admin Dashboard:**
    - **Noticeboard:** Allows admins to post notices on the student and teacher noticeboards.
    - **Admin profile:** Displays the role and profile of the admin.
    - **Students' management:** From this page admin can manage and upload the students and their information and also can view the list of current students.
    - **Teachers' management:** From this page admin can manage and upload the teachers and their information and also can view the list of current teachers.
    - **Clubs' management:** Similar to teachers and students, club information and new clubs can also be added by admin.

5.2 GUI Construction (Project Snapshots)

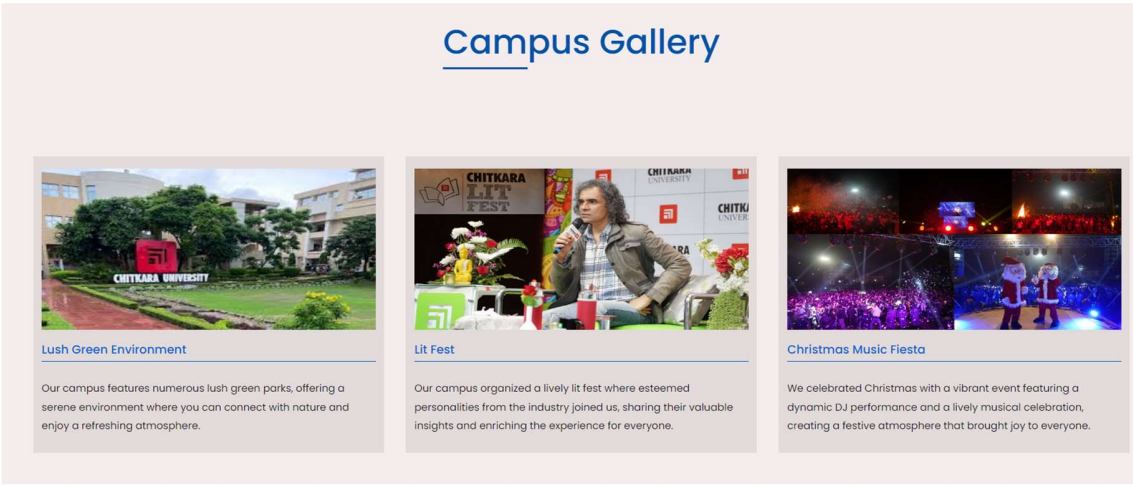
1. **Home page:**



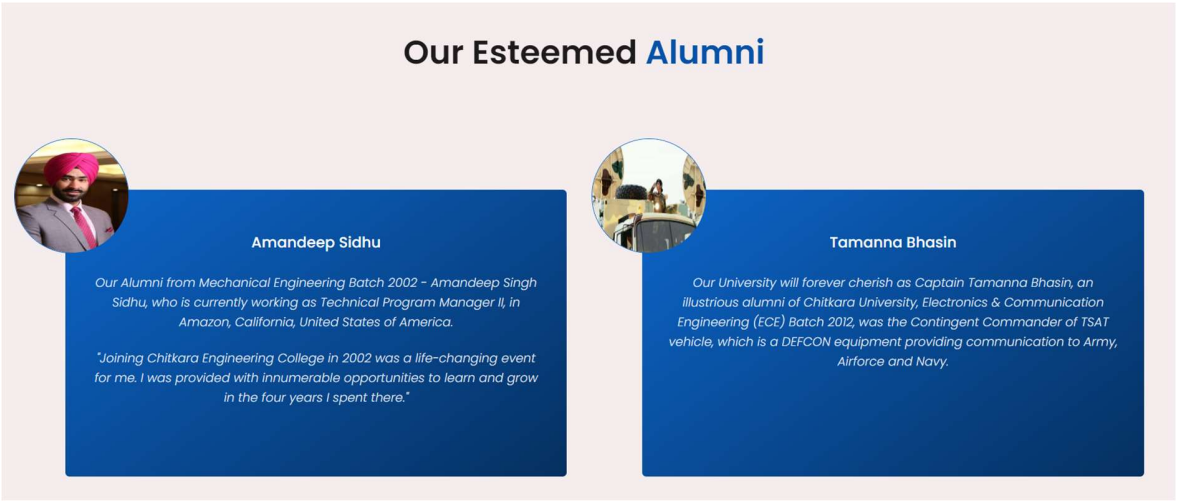
1.1 Landing page



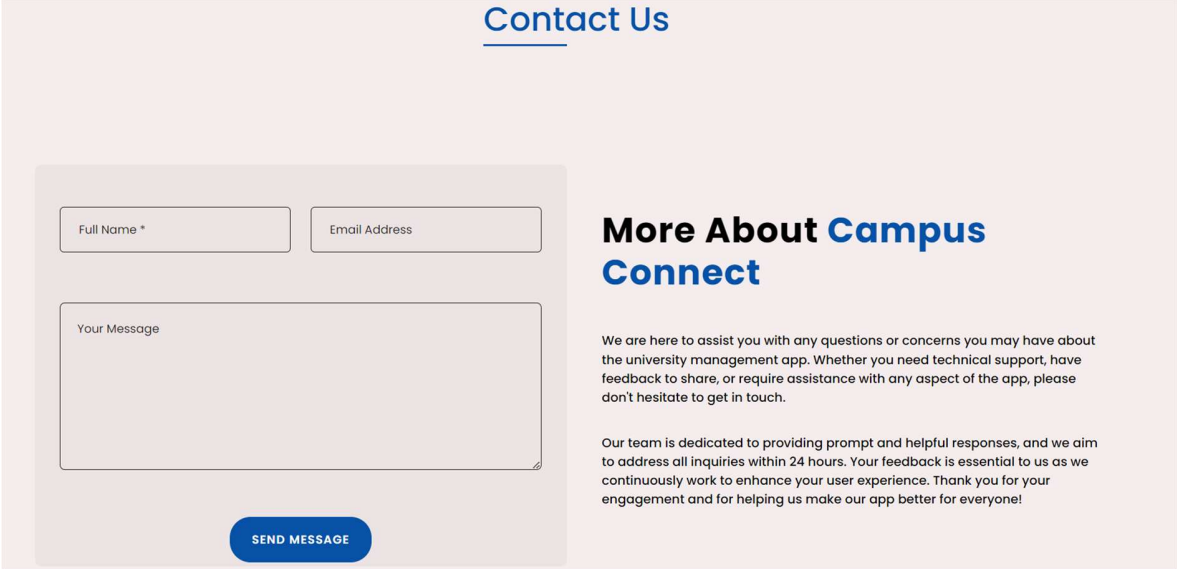
1.2 About us



1.3 Campus gallery

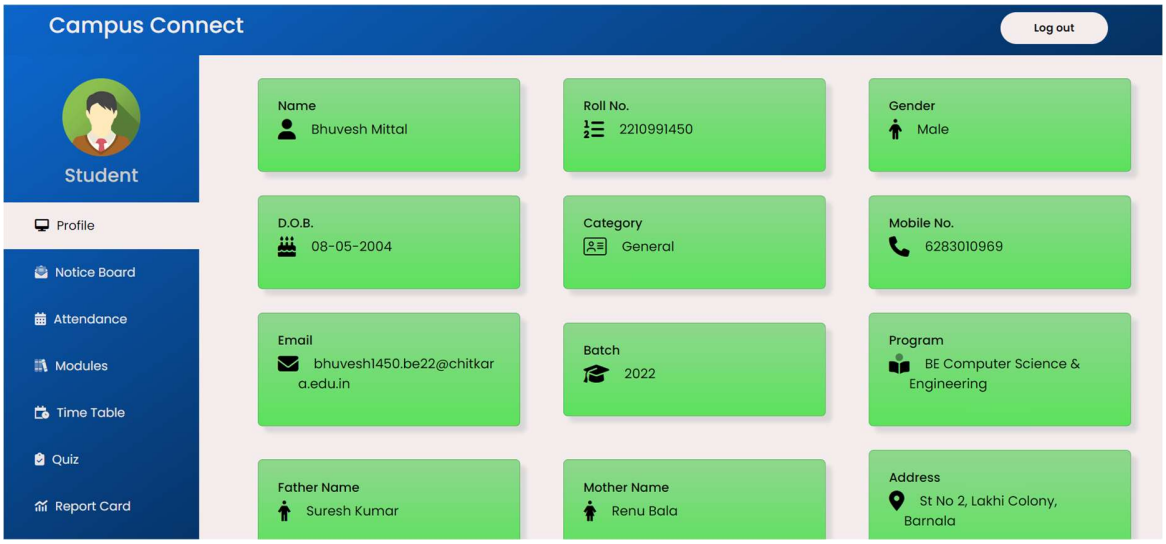


1.4 Our esteemed alumni

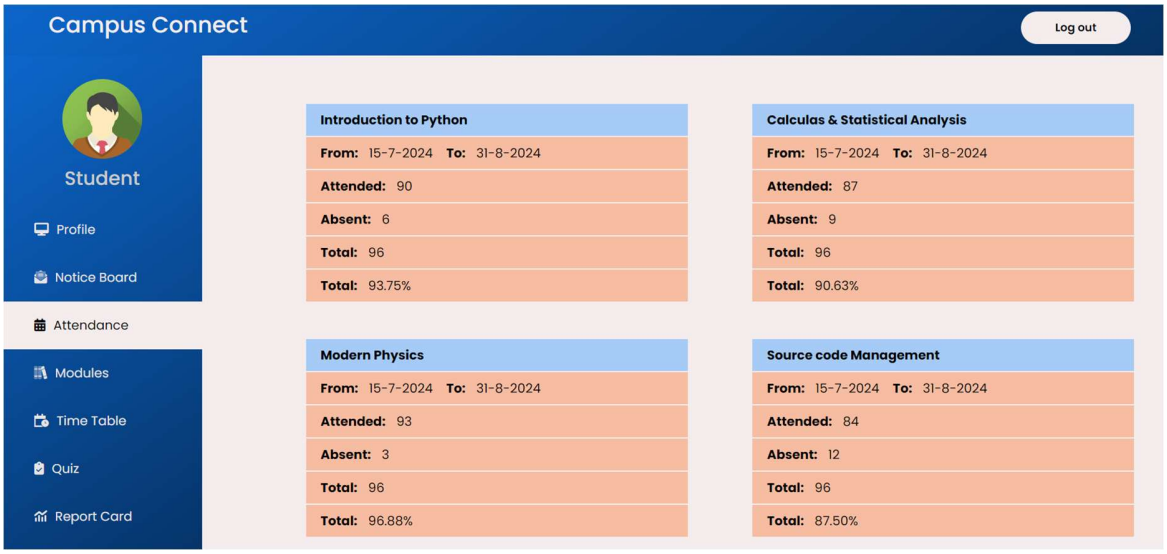


1.5 Contact us

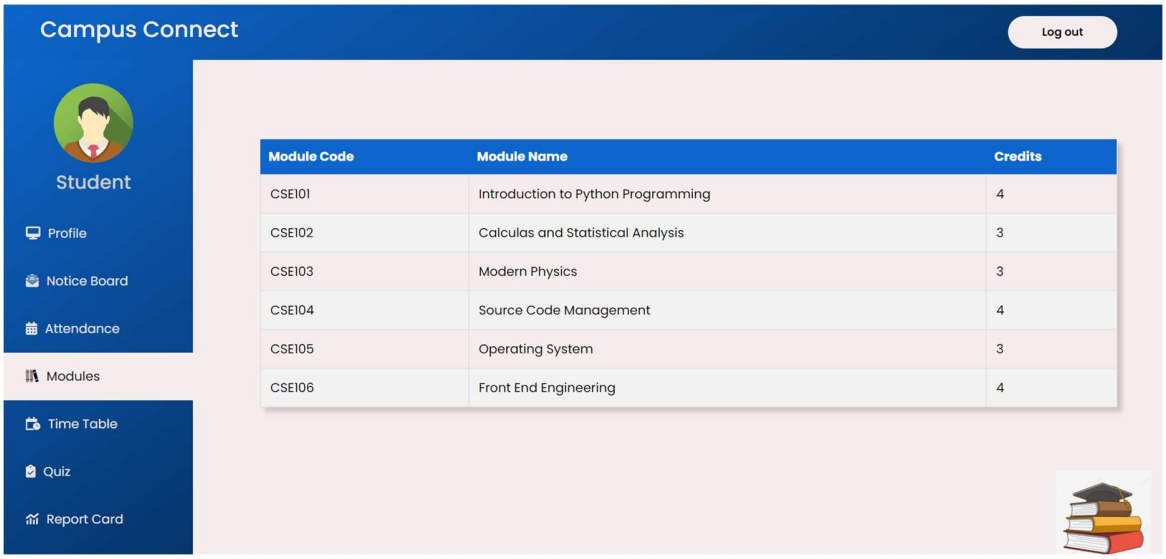
2. Student Dashboard Snapshot:



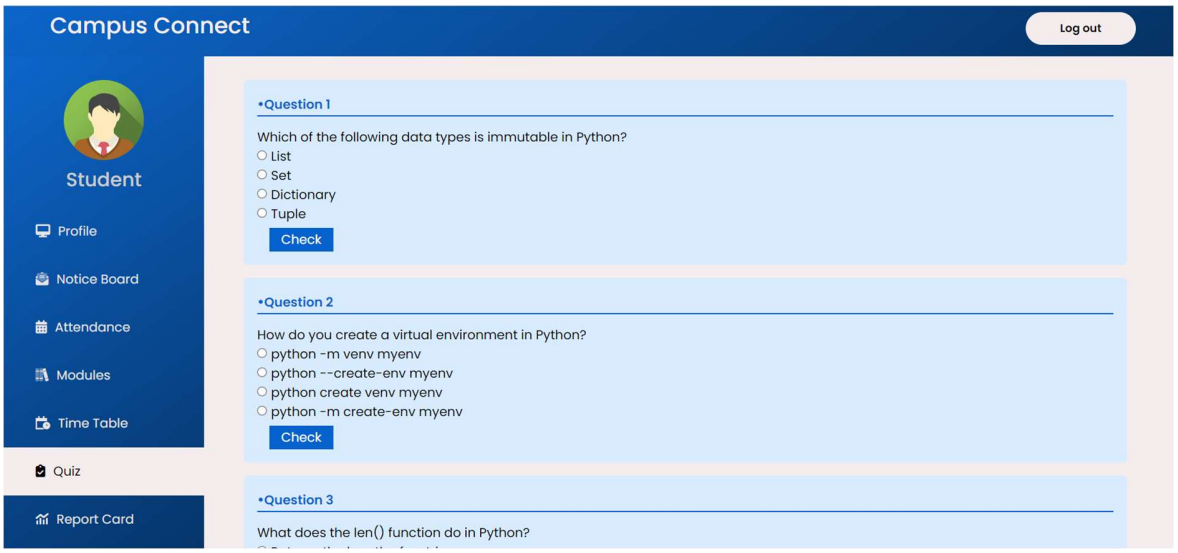
2.1 Student Profile



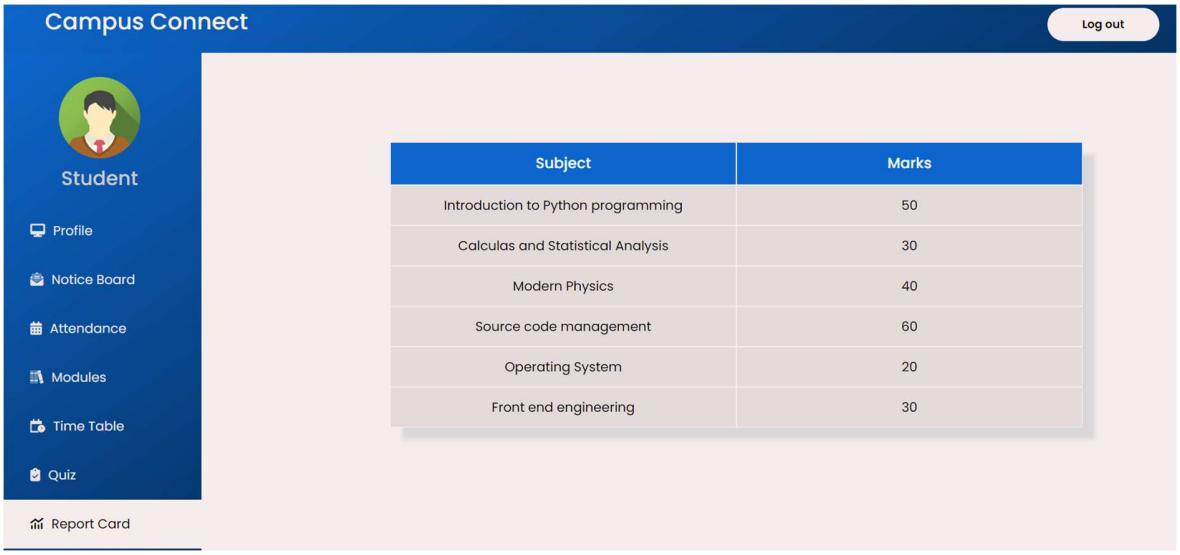
2.2 Student attendance



2.3 Student Modules

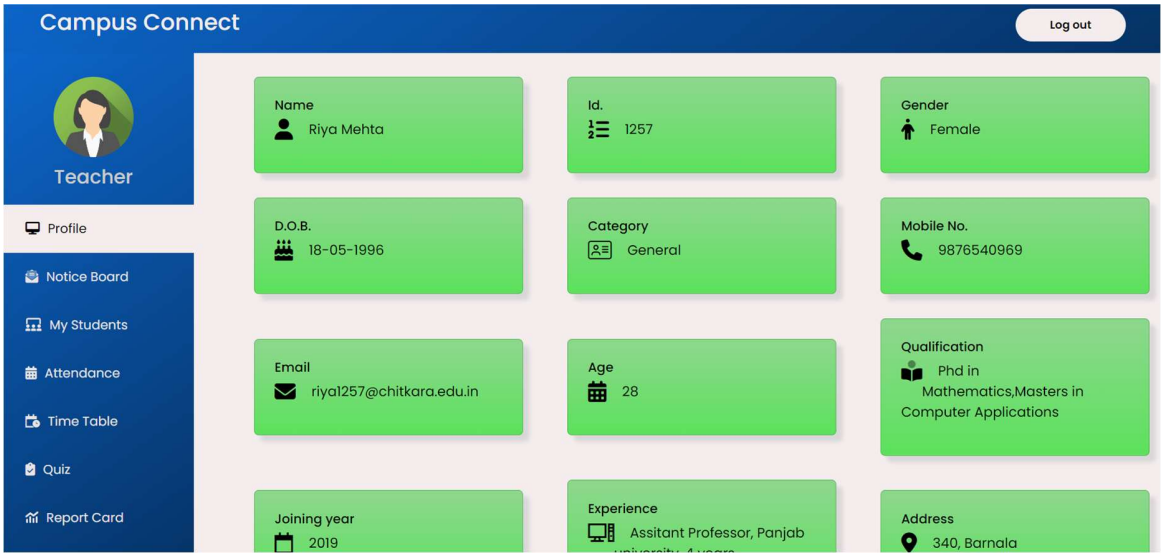


2.4 Student quizzes

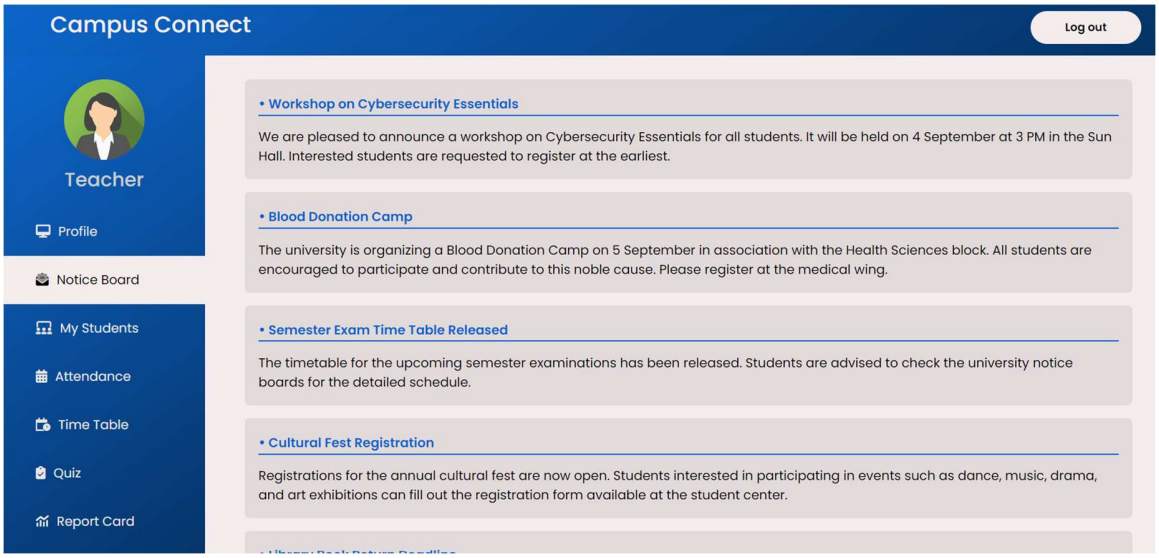


2.5 Student Report Card

3. Teacher Dashboard Snapshot:

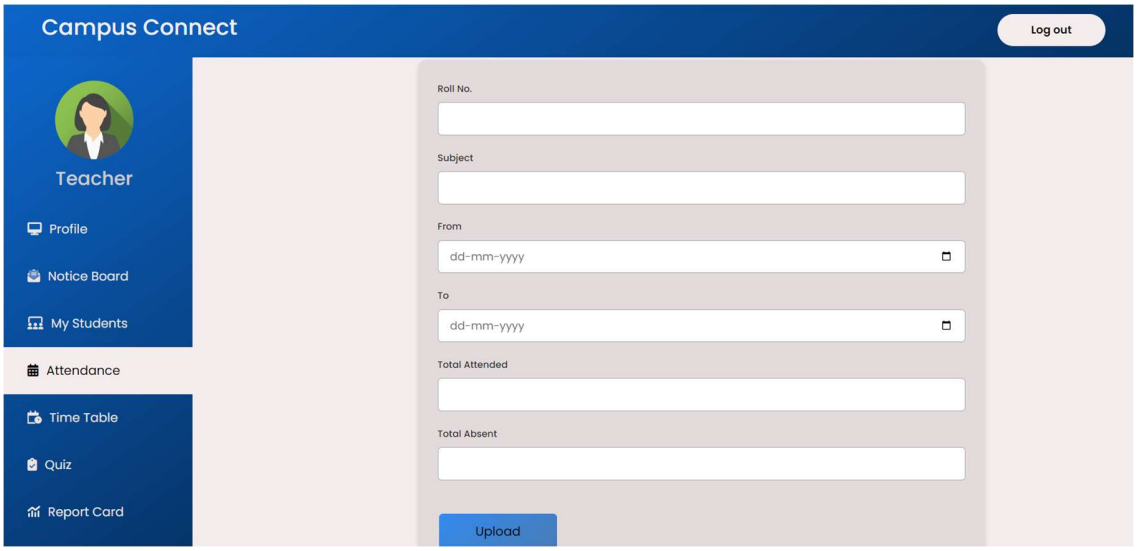


3.1 Teacher profile

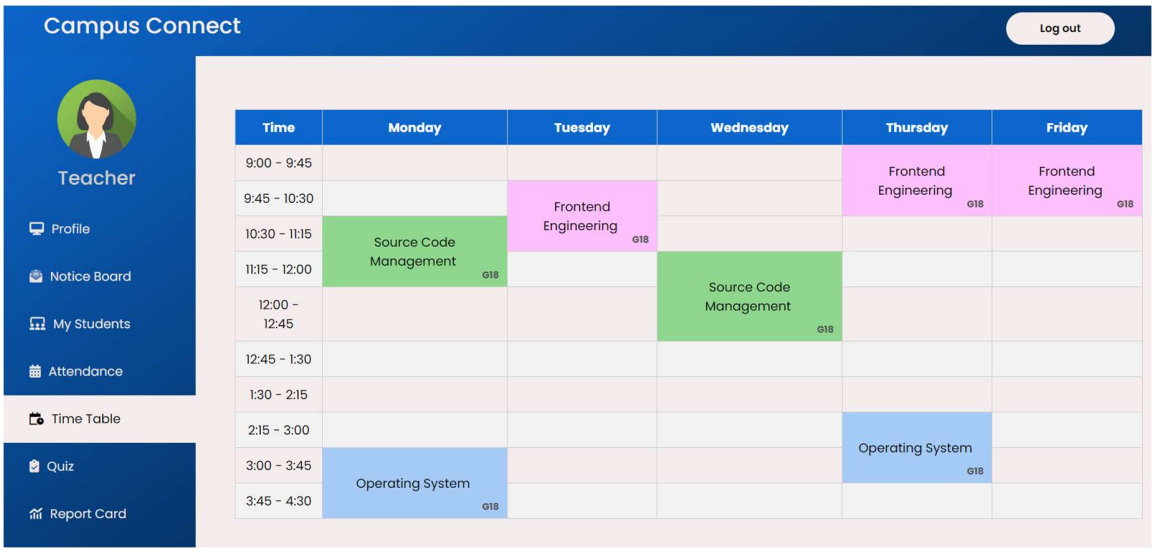


3.2 Teacher Noticeboard

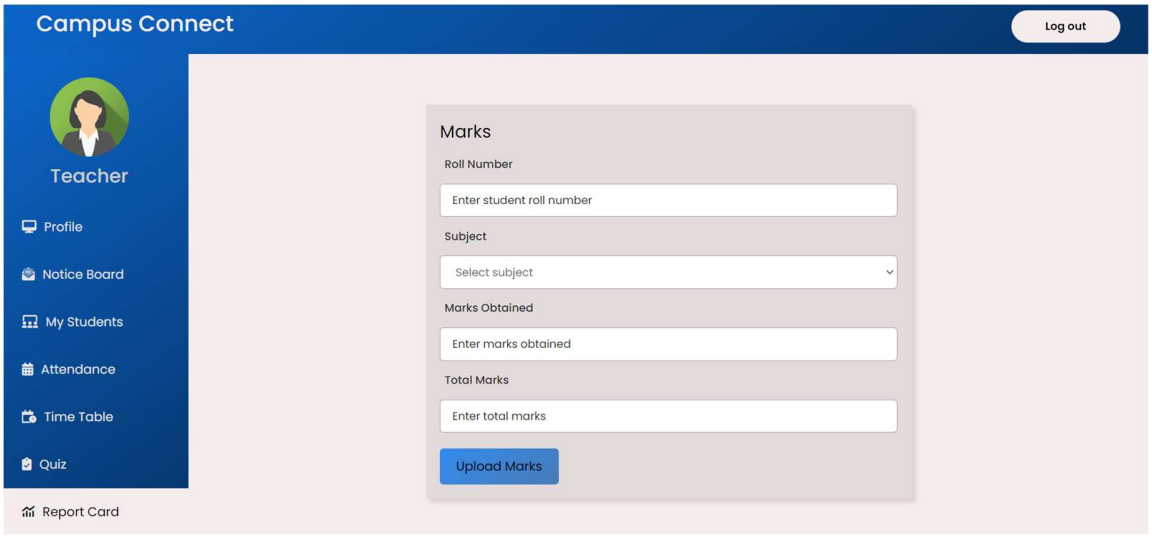
3.3 Teacher's list of students



3.4 Attendance uploaded by teacher

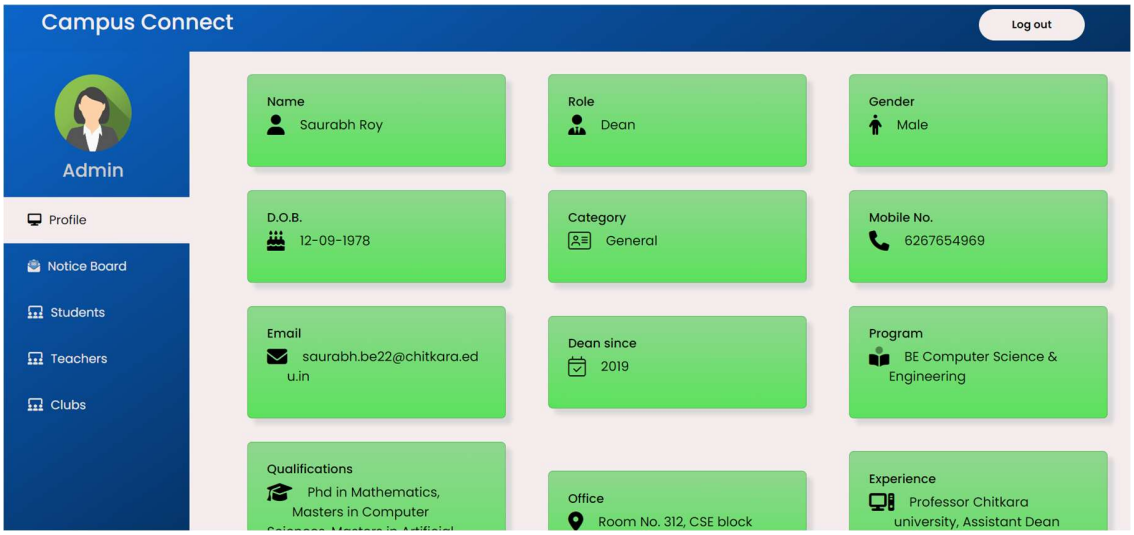


3.5 Teacher's timetable

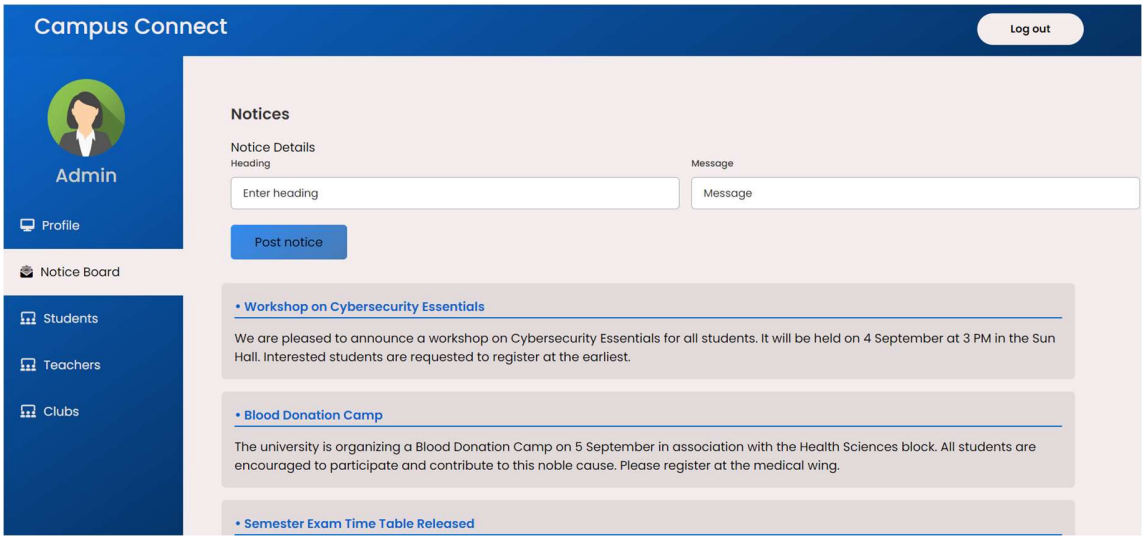


3.6 Marks uploaded by teacher

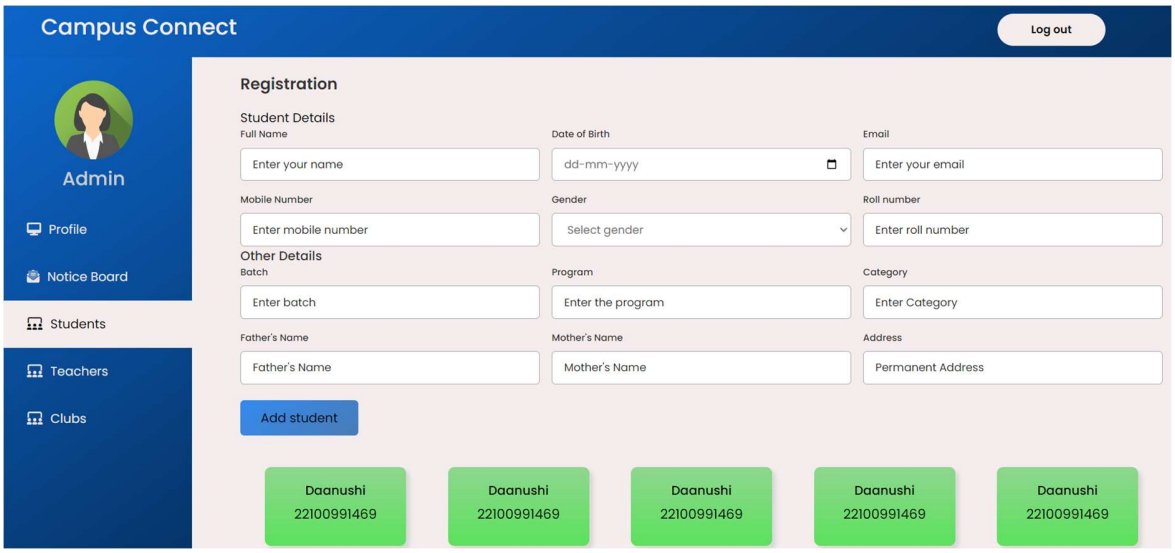
4. Admin Dashboard Snapshot:



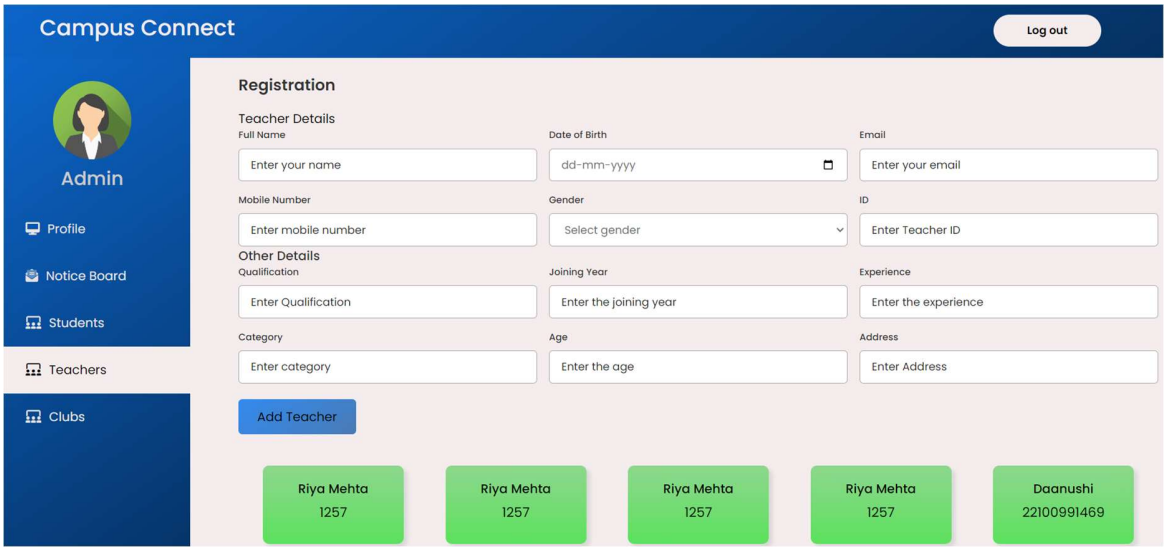
4.1 Admin profile



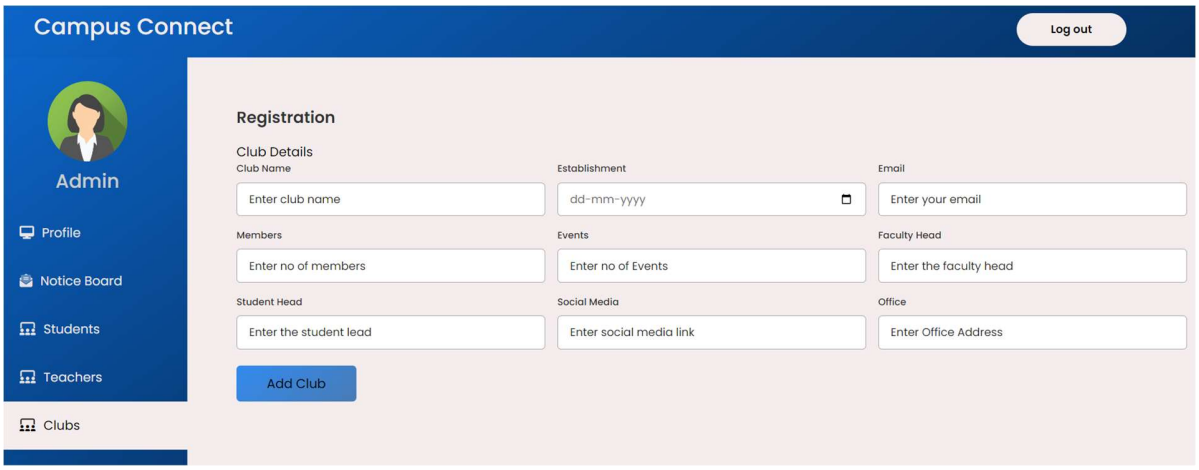
4.2 Notices uploading by admin



4.3 Students information uploaded by admin



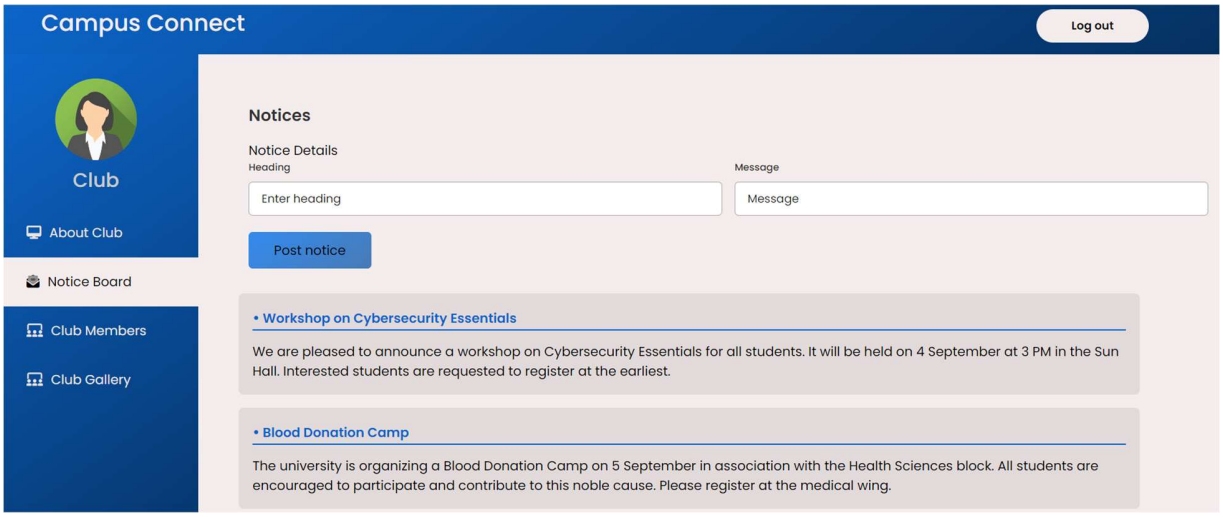
4.4 Teachers information uploaded by admin



4.5 Club information uploaded by admin

5. Club Dashboard Snapshot:

5.1 Club Information



5.2 Notices uploaded by admin

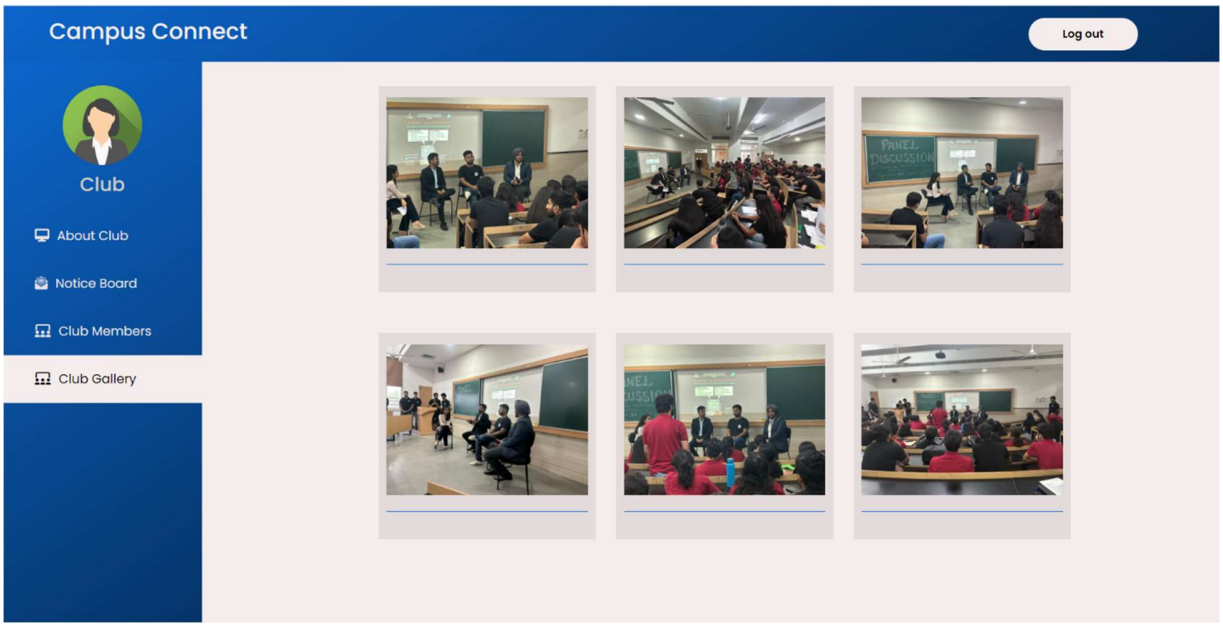
Club Heads

Name	Role	Contact
Abc	President	president_club@gmail.com
Efg	Vice President	vice_president_club@gmail.com
Hij	Technical Head	technical_head_club@gmail.com
Klm	Social Media Head	socialmedia_club@gmail.com

Club Members

Name	Team	Contact
Abc	Technical	m1@gmail.com
Efg	Technical	m2@gmail.com
Hij	Technical	m3@gmail.com
Klm	Technical	m4@gmail.com
Abc	Social Media	m5@gmail.com
Efg	Social Media	m6@gmail.com
Hij	Social Media	m7@gmail.com
Klm	Social Media	m8@gmail.com

5.3 Club Members



5.4 Club Gallery

5.3 GUI Design Considerations

- **User Experience (UX):** Each dashboard is designed to be intuitive, ensuring that users can easily navigate between sections and perform their tasks efficiently.
- **Responsive Design:** The interface is responsive, adapting to various screen sizes to provide a seamless experience on both desktop and mobile devices.
- **Consistency:** Uniform design elements and styles are used across different dashboards to maintain consistency and enhance usability.
- **Accessibility:** Ensures that the interface is accessible to users with disabilities, adhering to web accessibility standards.

6. Code-Implementation and Database Connections (If any)

```
import express from "express";
const app = express();
const port = 3000;
import web from "./routes/web.js";

app.set("view engine", "ejs");

app.use(express.static("public"));

app.use("/", web);

app.use(express.urlencoded({ extended: true }));
app.use(express.json());

const users = [
  { email: "student1@example.com", password: "student123", role: "student" },
  { email: "teacher1@example.com", password: "teacher123", role: "teacher" },
  { email: "club1@example.com", password: "club123", role: "club" },
  { email: "admin1@example.com", password: "admin123", role: "admin" },
];

app.post("/login", (req, res) => {
  const { email, password, role } = req.body;

  const user = users.find(
    (u) =>
      u.email === email &&
      u.password === password &&
      (u.role === role || u.role === "admin")
  );

  if (user) {
    if (user.role === "student") {
      return res.redirect("/student_dashboard");
    } else if (user.role === "teacher") {
      return res.redirect("/teacher_dashboard");
    } else if (user.role === "club") {
      return res.redirect("/club_dashboard");
    } else if (user.role === "admin") {
      return res.redirect("/admin_dashboard");
    }
  } else {      You, 17 hours ago • login completed
    return res.status(401).send("Invalid email, password, or role");
  }
});
```

CHITKARA UNIVERSITY

```
import { studentquizController } from "../controllers/studentquizController.js";
import { studentreportcardController } from "../controllers/studentreportcardController.js";
import { teachernoticeboardController } from "../controllers/teachernoticeboardControlle.js";
import { teacherstudentsController } from "../controllers/teacherstudent.js";
import { teacherattendanceController } from "../controllers/teacherattendanceController.js";
import { teachertimetableController } from "../controllers/teachertimetableController.js";
import { teacherquizesController } from "../controllers/teacherquizesController.js";
import { teacherreportcardController } from "../controllers/teacherreportcardController.js";
import { admindashboardController } from "../controllers/admindashboardController.js";
import { adminnoticeboardController } from "../controllers/adminnoticeboardController.js";
import { adminstudentsController } from "../controllers/adminstudentsController.js";
import { adminteachersController } from "../controllers/adminteachersController.js";

import { clubdashboardController } from "../controllers/clubdashboardController.js";
import { clubmembersController } from "../controllers/clubmembersController.js";
import { clubnoticeboardController } from "../controllers/clubnoticeboardController.js";
import { clubgalleryController } from "../controllers/clubgalleryController.js";
import { adminclubController } from "../controllers/adminclubController.js";

router.get("/", homeController);
router.get("/student_dashboard", studentdashboardController);
router.get("/student_noticeboard", studentnoticeboardController);
router.get("/student_attendance", studentattendanceController);
router.get("/student_modules", studentmoduleController);
router.get("/student_timetable", studenttimetableController);
router.get("/student_quizes", studentquizController);
router.get("/student_reportcard", studentreportcardController);
router.get("/teacher_dashboard", teacherdashboardController);
router.get("/teacher_noticeboard", teachernoticeboardController);
router.get("/teacher_students", teacherstudentsController);
router.get("/teacher_attendance", teacherattendanceController);
router.get("/teacher_timetable", teachertimetableController);
router.get("/teacher_quizes", teacherquizesController);
router.get("/teacher_reportcard", teacherreportcardController);
router.get("/admin_dashboard", admindashboardController);
router.get("/admin_noticeboard", adminnoticeboardController);
router.get("/admin_students", adminstudentsController);
router.get("/admin_teachers", adminteachersController);

router.get("/club_dashboard", clubdashboardController);
router.get("/club_members", clubmembersController);
router.get("/club_noticeboard", clubnoticeboardController);
router.get("/club_gallery", clubgalleryController);
router.get("/admin_club", adminclubController);

export default router;
```

```
const homeController = (req, res) => {
  res.render("pages/home", { title: "Home" });
};
export { homeController };      You, 3 weeks ago
```

7. Future Scope

The university management system project, while already offering a robust set of functionalities, has significant potential for further development and enhancement. Several key areas have been identified for future improvements:

1. Database Integration: Currently, the system operates without a database. Integrating MongoDB in the future will provide a scalable and efficient way to manage and store data. This will enable complex queries, data analysis, and improved data consistency. MongoDB's NoSQL architecture will support flexible data models, making it easier to handle various types of data related to students, teachers, and administrative records.

2. Advanced Analytics and Reporting: With the addition of a database, the system will be able to implement advanced analytics and reporting features. This will allow administrators and faculty to generate detailed reports on academic performance, attendance trends, and other key metrics. Data visualization tools can also be integrated to present these insights in a user-friendly manner.

3. Enhanced Security Measures: Future development will focus on bolstering security protocols to protect sensitive information. This includes implementing encryption for data at rest and in transit, multi-factor authentication for user access, and rigorous access controls. Regular security audits and updates will be crucial to safeguarding against potential threats.

4.Mobile Application Development: To enhance accessibility, a mobile application version of the system could be developed. This would allow students, teachers, and administrators to access the system's features on the go, increasing convenience and engagement. The mobile app could include push notifications for important updates and deadlines.

5. Integration with External Systems: The system could be extended to integrate with other university systems and third-party applications. For example, integration with library management systems, financial aid platforms, and online learning tools could provide a more holistic view of student and faculty activities.

6. User Experience Improvements: Continuous feedback from users will guide further enhancements in the user interface and experience. This includes refining the design for better usability, adding personalized features, and ensuring the system remains intuitive and efficient for all stakeholders.

By addressing these areas, the university management system can evolve to meet the growing and changing needs of the academic institution, ultimately contributing to a more efficient and effective university administration.

8. Conclusion

The development of the university management system represents a pivotal advancement in the administration and academic processes within a university environment. By leveraging Node.js, Express.js, EJS, and HTML/CSS, the project aims to integrate a variety of functionalities into a single, cohesive platform tailored to the needs of students, teachers, clubs, and administrators.

The system's design effectively centralizes key functions, such as managing student profiles, tracking attendance, and handling mark submissions, into a unified interface. This centralization not only simplifies administrative tasks but also enhances operational efficiency by providing seamless access to essential data. Each user group benefits from a dedicated dashboard—Student, Teacher, Club, and Admin—ensuring that the system caters specifically to their needs and responsibilities. For instance, students can view their profiles, attendance records, and academic achievements, while teachers can manage their schedules, upload attendance, and post marks.

Looking ahead, the integration of MongoDB into the project will be a crucial step, enabling robust data storage and management capabilities. This enhancement will support more complex data operations and scalability, which are essential for accommodating the growing needs of the university. Although the system currently lacks a database, the planned inclusion will facilitate comprehensive data handling and reporting.

The design prioritizes user experience through a responsive interface that adapts to various devices, ensuring that the system is both user-friendly and accessible. Security measures are anticipated to be a key focus in future developments, with plans to implement data encryption, secure connections, and strict access controls to safeguard sensitive information.

In conclusion, this university management system project lays a strong foundation for improving administrative and academic efficiency within the institution. With future enhancements planned for database integration and security, the system is poised to offer even greater support to the university's operations and its diverse stakeholders.

9. References

1. *Express.Js tutorial*. (n.d.). Www.javatpoint.com. Retrieved September 23, 2024, from https://www.javatpoint.com/expressjs-tutorial
2. *Express.Js tutorial*. (2024, January 24). GeeksforGeeks. https://www.geeksforgeeks.org/express-js/
3. *EJS -- embedded JavaScript templates*. (n.d.). Ejs.Co. Retrieved September 23, 2024, from https://ejs.co/
4. *Use EJS as template engine in node.Js*. (2019, June 27). GeeksforGeeks. https://www.geeksforgeeks.org/use-ejs-as-template-engine-in-node-js/
5. Kumar, S. (n.d.). *Schoolmanagement: School Management System*.
6. - *YouTube*. (n.d.). Youtu.Be. Retrieved September 25, 2024, from https://youtu.be/0pUFsItoDLo?si=CRSoUB7nOfw2zMHI
7. - *YouTube*. (n.d.-b). Youtu.Be. Retrieved September 25, 2024, from https://youtu.be/XIYdVPKQpiQ?si=zATDA3fUmoBsVyX7
8. *Express routing*. (n.d.). Expressjs.com. Retrieved September 25, 2024, from https://expressjs.com/en/guide/routing.html
9. *Serving static files in Express*. (n.d.). Expressjs.com. Retrieved September 25, 2024, from https://expressjs.com/en/starter/static-files.html
10. *Express JS HTTP methods*. (2023, November 21). GeeksforGeeks. https://www.geeksforgeeks.org/express-js-http-methods/
11. *Chitkara University*. (n.d.). Codebrigade.In. Retrieved September 25, 2024, from https://cuiet.codebrigade.in/loginManager/load