**“EDEX College Management System”**



**A PROJECT REPORT SUBMITTED TO GOA UNIVERSITY**

**IN PARTIAL FULFILMENT OF THE REQUIREMENT**

**FOR THE DEGREE OF BCA**

**By**

**Mr. Jarvis Silva**

**Mr. Anandhu Rengan**

**Mr. Abhishek Verma**

**Mr. Anish Pratap Parsekar**

**Mr. Kanishk Singh Negi**

**Mr. Nikhil Jaiswal**

**Mr. Soubhagya Ranjan Das**

**Ms. Sonia Sandeep Dhargalkar**

**ASST. PROF. SONALI KARAPURKAR DR. MANASVI KAMAT**

**INTERNAL GUIDE PRINCIPAL**

**Murgaon Education society**

**M.E.S VASANT JOSHI COLLAGE OF ARTS AND COMMERCE ZUARINAGAR GOA- 403 726**

**DEPARTMENT OF COMPUTER APPLICATION**

**2023-2024**



**Murgaon Education society**

**M.E.S VASANT JOSHI COLLAGE OF ARTS AND COMMERCE ZUARINAGAR GOA- 403 726**

**DEPARTMENT OF COMPUTER APPLICATION**

**2023-2024**

**Affiliated to Goa University**

**CERTIFICATE**

This is to certify a project on

**“EDEX College Management System”**

Has been successfully completed by

**Mr. Jarvis Silva**

**Mr. Anandhu Rengan**

**Mr. Abhishek Verma**

**Mr. Anish Pratap Parsekar**

**Mr. Kanishk Singh Negi**

**Mr. Nikhil Jaiswal**

**Mr. Soubhagya Ranjan Das**

**Ms. Sonia Sandeep Dhargalkar**

Studying in TYBCA during the academic year 2023-2024.

The project has been carried out under the supervisior of the internal guide(s).

**SONALI KARAPURKAR EXTERNAL EXAMINER DR. MANASVI KAMAT**

**INTERNAL GUIDE PRINCIPAL**

**PROJECT CO-ORDINATOR**

Date:

Place: Zuarinagar



**DECLARATION BY CANDIDATE**

We declare that this project report has been prepared by us to the best of our knowledge, it has not previously formed the basis for the award of any diploma or degree by this or any other University.

|  |  |  |
| --- | --- | --- |
| **ROLL NO** | **NAME** | **SIGNATURE** |
| **2105016** | **Jarvis Silva** |  |
| **2105101** | **Anandhu Rengan** |  |
| **2105002** | **Abhishek Verma** |  |
| **2105104** | **Anish Pratap Parsekar** |  |
| **2105017** | **Kanishk Singh Negi** |  |
| **2105021** | **Nikhil Jaiswal** |  |
| **2105117** | **Soubhagya Ranjan Das** |  |
| **2105116** | **Sonia Sandeep Dhargalkar** |  |

Date :

Place : Zuarinagar

**ACKNOWLEDGMENT**

The credit for this project goes to the sincere and timely guidance received from various people. We wouldn't be able to proceed smoothly with the design, without the constant encouragement and support received from our well-wishers. We are immensely grateful to our principal **DR MANASVI KAMAT** and **MS GAURI DESSAI** Head of the Department who assured us every support from the Institute.

We also wish to convey our sincere gratitude to our internal guide who encouraged and helped us in completion of the design phase. A word of gratitude to our project guide **SONALI KARAPURKAR** for providing all the necessary resources and guidance needed for the design of the project.

We would also like to thank my parents and friends for supporting and helping us a lot in finalizing this project work in the limited time frame. Last but not the least the god almighty, who gave us the knowledge and strength for this project.

TABLE OF CONTENT

[1: INTRODUCTION 1](#_Toc161937759)

[2: EXISTING SYSTEM & IT’S LIMITATIONS 2](#_Toc161937760)

[2.1 Existing System 2](#_Toc161937761)

[2.2 Limitations: 3](#_Toc161937762)

[3: PROPOSED SYSTEM 4](#_Toc161937763)

[4: USE CASE DIAGRAM 5](#_Toc161937764)

[5: ER DIAGRAM 6](#_Toc161937765)

[6: NEXT JS ARCHITECTURE 7](#_Toc161937766)

[7: LIST OF WEBSITE PAGES 9](#_Toc161937767)

[8: DATA MODELS 18](#_Toc161937768)

[9: AUTHENTICATION 20](#_Toc161937769)

[10: TECHNOLOGIES USED 22](#_Toc161937770)

[11: TESTING REPORT 25](#_Toc161937771)

[12: USER MANUAL 30](#_Toc161937772)

[13: FUTURE ENHANCEMENTS 34](#_Toc161937773)

[14: CONCLUSION 36](#_Toc161937774)

[15: REFERENCES 37](#_Toc161937775)

[16: GANTT CHART 38](#_Toc161937776)

**Table of Figures**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr** | **Fig** | **Name** | **Page No** |
| 1 | 4.1 | Use Case Diagram | 5 |
| 2 | 5.1 | ER Diagram | 6 |
| 3 | 8.1 | Data Models Diagram | 18 |
| 4 | 12.1 | College Setup Page | 30 |
| 3 | 12.2 | Login Page | 31 |
| 4 | 12.3 | Manage Course Page | 31 |
| 5 | 12.4 | Manage Classroom Page | 32 |
| 6 | 16.1 | Ghantt Chart | 38 |

# 1: INTRODUCTION

The Edex College Management System is a robust web-based application designed to revolutionize the administrative processes within educational institutions, specifically colleges.

Through its comprehensive suite of features and functionalities, Edex aims to streamline operations, enhance communication, and improve efficiency across various aspects of college management.

In this introduction, we will dive into the project's overview, elucidate its objectives and goals, define its scope and limitations, and identify its target audience. Additionally, we will provide an overview of the key components and functionalities of the Edex platform.

**Overview of the Project**

Edex represents a transformative solution for colleges seeking to optimize their administrative workflows. Developed on the Next.js framework and backed by a MongoDB database, Edex provides a centralized platform for managing critical aspects of college operations.

Its core functionalities include course management, classroom allocation, student attendance tracking, marks management, and resource sharing.

By consolidating these functionalities into a unified platform, Edex empowers colleges to streamline processes, improve transparency, and enhance collaboration among stakeholders.

# 2: EXISTING SYSTEM & IT’S LIMITATIONS

## 2.1 Existing System

The existing college management system relies predominantly on manual processes and traditional record-keeping methods, which are labor-intensive and prone to inefficiencies. The system comprises physical registers, paper-based documentation, and disjointed spreadsheets for managing various aspects of academic and administrative operations. Below is a detailed analysis of the existing system and its limitations:

1. **Manual Book Registers:**

The cornerstone of the existing system is the use of manual book registers to record vital information such as attendance, marks, and resource allocations. These registers require constant manual updates, making them susceptible to errors, inaccuracies, and data redundancy.

1. **Paper-Based Documentation:**

Administrative tasks, including course registrations, student enrollments, and faculty management, heavily rely on paper-based documentation. This reliance on physical documents introduces challenges related to data storage, retrieval, and sharing, often leading to delays and inefficiencies.

1. **Disparate Spreadsheets:**

Different departments and units within the college utilize disparate spreadsheets to manage academic and administrative data. However, this fragmented approach results in data silos, hindering seamless information exchange and integration across the institution.

1. **Limited Accessibility:**

Access to critical information is constrained by physical and organizational barriers, limiting collaboration and transparency. Information stored in manual formats is often inaccessible to stakeholders outside specific departments, impeding decision-making and operational agility.

1. **Lack of Integration:**

The absence of an integrated system leads to redundant data entry and duplication of efforts across different processes.

## 2.2 Limitations:

* **Time-Consuming Processes:** Manual data entry and record-keeping processes are time-consuming and labor-intensive, diverting resources from more strategic tasks.
* **Error-Prone:** Manual processes are inherently prone to errors, leading to inaccuracies in attendance records, student grades, and other critical information.
* **Limited Accessibility:** Access to information is restricted, hindering collaboration and decision-making among faculty, students, and administrative staff.
* **Data Fragmentation:** Information is fragmented across various physical and digital formats, making it difficult to consolidate and analyze data for informed decision-making.
* **Inefficiency:** Manual systems lack automation and efficiency, resulting in delays, redundancies, and operational bottlenecks.
* **Scalability Challenges:** As the college grows, manual systems become increasingly unsustainable, leading to scalability challenges and administrative burdens.

# 3: PROPOSED SYSTEM

The proposed Edex CMS aims to revolutionize the management of academic and administrative tasks by introducing an integrated and user-friendly platform. Below is an overview of the proposed system:

1. **Unified Authentication and Access Control:**

The system provides a unified authentication mechanism for all users, with a common login page accessible to administrators, teachers, and students. Upon successful authentication, users are directed to the dashboard, where they can manage their respective tasks based on their roles.

1. **Efficient Course and Classroom Management:**

Edex offers comprehensive functionalities for course and classroom management. Administrators and course heads can create, edit, and view courses, while classroom heads have similar privileges for classrooms. New courses and classrooms can be added, ensuring efficient organization and planning of academic programs.

1. **Subject Management and Resource Sharing:**

The system facilitates the creation, editing, and viewing of subjects within classrooms. Teachers and administrators can upload and share resources, such as notes and study materials, enhancing the learning experience for students. Additionally, attendance and marks for each subject are conveniently accessible.

1. **Comprehensive User Management:**

Edex enables administrators to manage users, including teachers and students, through dedicated pages. New users can be added, existing profiles can be edited, and detailed information about users.

1. **Monitoring and Reporting:**

The system provides monitoring and reporting capabilities, allowing admins and other users to track attendance, marks, and other key metrics. Users can view attendance, add new attendance entries, and make edits as needed. Similarly, marks can be accessed, added, and edited through the marks page, facilitating comprehensive student assessment and evaluation.

# 4: USE CASE DIAGRAM

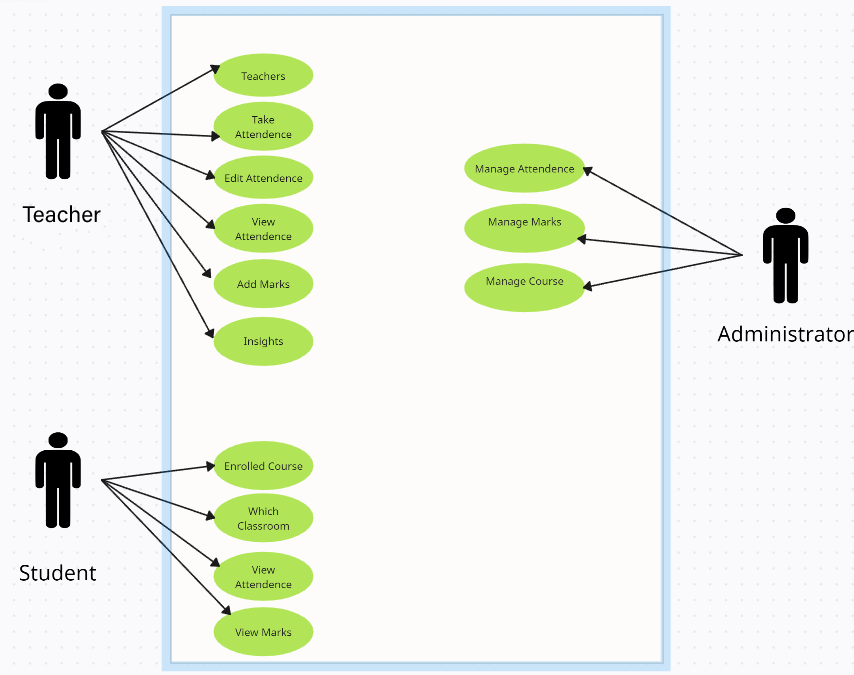


Fig 4.1 User Case Diagram

# 5: ER DIAGRAM

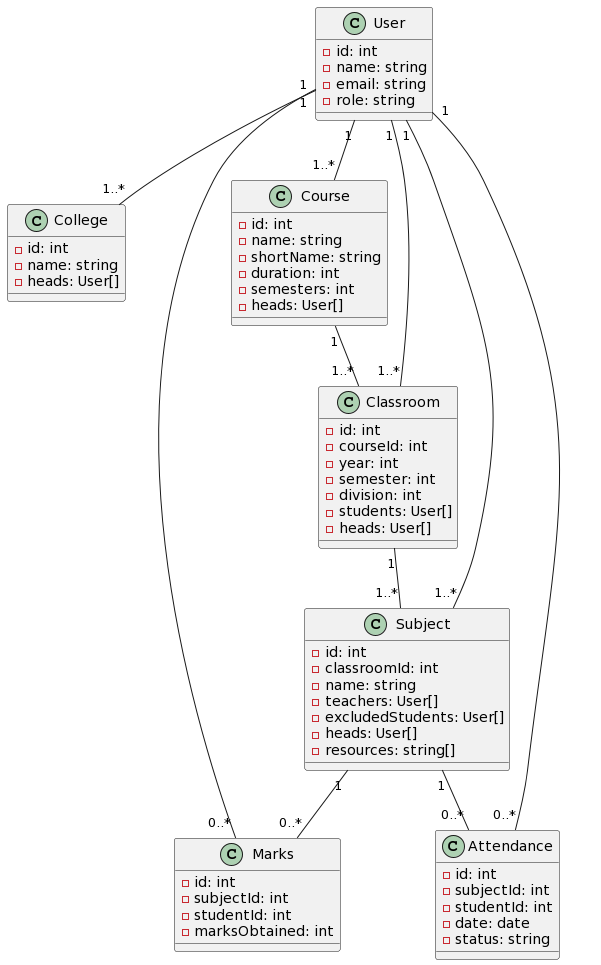
****

Fig 5.1 ER Diagram

# 6: NEXT JS ARCHITECTURE

The architecture of the Edex College Management System (CMS) is designed to provide a comprehensive solution for managing various aspects of a college's operations. Built on Next.js 14 framework, Edex CMS leverages advanced architectural concepts to deliver a scalable, secure, and efficient platform for colleges.

**Key Components:**

1. **Next.js Framework:**
   * Edex CMS is built on the Next.js framework, which offers features such as server-side rendering, static site generation, and API routes.
   * Next.js provides a robust foundation for building dynamic web applications with enhanced performance and SEO capabilities.
2. **Folder-Based Routing System:**
   * The application follows a folder-based routing system, organized within the **app** directory.
   * Each folder within **app** represents a specific route in the application, promoting modularity and separation of concerns.
3. **Server Actions:**
   * Server Actions are asynchronous functions that execute on the server-side, providing access to server resources and enabling tasks such as database interactions, file system operations, and authentication.
   * These actions enhance security, performance, and maintainability by offloading complex tasks from the client-side.
4. **Tailwind CSS for Styling:**
   * Tailwind CSS is used for styling the user interface of Edex CMS.
   * Tailwind CSS is a utility-first CSS framework that offers a wide range of pre-built styles and components, allowing for rapid development and customization of UI elements.

**Benefits:**

* **Scalability:** The architecture of Edex CMS is designed to scale seamlessly as the application grows, ensuring performance and reliability even in large-scale deployments.
* **Security:** By leveraging JWT-based authentication and server-side logic, Edex CMS enhances security by protecting sensitive user data and preventing common vulnerabilities.
* **Performance:** Server-side rendering and optimized client-server communication contribute to improved performance and faster response times, enhancing the overall user experience.
* **Modularity:** The folder-based routing system, combined with Tailwind CSS for styling, facilitates code organization and UI customization, making it easier to maintain and extend the application over time.

The architecture of Edex CMS combines the power of Next.js framework, folder-based routing, server-side rendering, and authentication mechanisms with Tailwind CSS for styling to deliver a robust and feature-rich college management system. With its emphasis on scalability, security, performance, and modularity, Edex CMS provides colleges with a reliable platform to streamline their operations and enhance collaboration between stakeholders.

# 7: LIST OF WEBSITE PAGES

**CollegeSetup & UserLogin - "/"**

This page serves as the entry point for the Edex application. If the college setup is not completed, users are directed to the College Setup page to configure the initial settings. Otherwise, users are redirected to the User Login page for authentication.

**Functionality:**

* College setup: Configure initial settings if not set up.
* User login: Authenticate users to access the dashboard.

**Access:** Public.

**VerifyLogin - "/verify-login"**

This page verifies the JWT token from the verification link sent via email during the registration process. It ensures the security and validity of user authentication tokens.

**Functionality:** Verify JWT token from email verification link.

**Access:** Public.

**Dashboard - "/dashboard"**

The Dashboard is the main interface after user authentication. It provides access to various modules and functionalities of the Edex system.

**Functionality:**

* Access to different modules (Courses, Classroom, Subject, Teachers, Students, Attendance, Marks).
* View overall system insights.
* Profile management.

**Access:** Authenticated users (Admin, Teachers, Students).

**Profile - "/profile"**

The Profile page allows users to view and manage their personal information and settings.

**Functionality:**

* View user profile details.
* Update personal information.
* Change password.

**Access:** Authenticated users (Admin, Teachers, Students).

**ViewCourses - "/dashboard/courses"**

Displays all courses offered by the college. Administrators and authorized users can manage courses from this page.

**Functionality:**

* View list of courses.
* Edit course details.
* Delete courses (for admins).

**Access:** Authenticated users (Admins, Course Heads).

**NewCourse - "/dashboard/courses/new"**

Allows administrators and authorized users to create a new course.

**Functionality:**

* Fill out course details.
* Submit to create a new course.

**Access:** Authenticated users (Admins, Course Heads).

**EditCourse - "/dashboard/courses/:courseId/edit"**

Enables administrators and authorized users to edit existing course details.

**Functionality:**

* Edit course information.
* Save changes.

**Access:** Authenticated users (Admins, Course Heads).

**ViewCourse - "/dashboard/courses/:courseId"**

Displays detailed information about a specific course, including associated classrooms, subjects, and other relevant details.

**Functionality:**

* View course details.
* Manage associated classrooms and subjects.

**Access:** Authenticated users (Admins, Course Heads).

**NewClassroom - "/dashboard/courses/:courseId/classroom/new"**

Allows authorized users to create a new classroom under a specific course.

**Functionality:**

* Fill out classroom details.
* Submit to create a new classroom.

**Access:** Authenticated users (Admins, Course Heads).

**EditClassroom - "/dashboard/courses/:courseId/classroom/:id/edit"**

Enables authorized users to edit existing classroom details.

**Functionality:**

* Edit classroom information.
* Save changes.

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads).

**ViewClassroom - "/dashboard/courses/:courseId/classroom/:id"**

Displays detailed information about a specific classroom, including associated subjects and other relevant details.

**Functionality:**

* View classroom details.
* Manage associated subjects.

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads).

**NewSubject - "/dashboard/courses/:courseId/classroom/:id/subject/new"**

Allows authorized users to create a new subject under a specific classroom.

**Functionality:**

* Fill out subject details.
* Submit to create a new subject.

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads).

**EditSubject - "/dashboard/courses/:courseId/classroom/:id/subject/:id/edit"**

Enables authorized users to edit existing subject details.

**Functionality:**

* Edit subject information.
* Save changes.

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads, Subject Heads).

**ViewSubject - "/dashboard/courses/:courseId/classroom/:id/subject/:id"**

Displays detailed information about a specific subject, including enrolled students, teachers, and shared resources.

**Functionality:**

* View subject details.
* Access shared resources.
* View attendance and marks of students.

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads, Subject Heads, Teachers, Students).

**ViewTeachers - "/dashboard/teachers"**

Displays a list of teachers registered in the system.

**Functionality:**

* View list of teachers.
* Edit teacher details (for admins and authorized users).
* Delete teachers (for admins).

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads).

**NewTeacher - "/dashboard/teachers/new"**

Allows administrators and authorized users to add a new teacher to the system.

**Functionality:**

* Fill out teacher details.
* Submit to add a new teacher.

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads).

**EditTeacher - "/dashboard/teachers/:id/edit"**

Enables administrators and authorized users to edit existing teacher details.

**Functionality:**

* Edit teacher information.
* Save changes.

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads).

**ViewTeacher - "/dashboard/teachers/:id"**

Displays detailed information about a specific teacher, including assigned subjects and other relevant details.

**Functionality:**

* View teacher details.

**Access:** Authenticated users (Admins, Course Heads, Classroom Heads).

**ViewStudents - "/dashboard/students"**

Displays a list of students registered in the system.

**Functionality:**

* View list of students.
* Edit student details (for admins and authorized users).
* Delete students (for admins).

**Access:** Authenticated users (Admins, Teachers).

**NewStudent - "/dashboard/students/new"**

Allows administrators and authorized users to add a new student to the system.

**Functionality:**

* Fill out student details.
* Submit to add a new student.

**Access:** Authenticated users (Admins, Teachers).

**EditStudent - "/dashboard/students/:id/edit"**

Enables administrators and authorized users to edit existing student details.

**Functionality:**

* Edit student information.
* Save changes.

**Access:** Authenticated users (Admins, Teachers).

**ViewStudent - "/dashboard/students/:id"**

Displays detailed information about a specific student, including enrolled courses, attendance, and marks.

**Functionality:**

* View student details.

**Access:** Authenticated users (Admins, Teachers).

**ViewAttendance - "/dashboard/attendance"**

Displays attendance records of students.

**Functionality:**

* View attendance records.
* Edit attendance records (for admins and authorized users).

**Access:** Authenticated users (Admins, Teachers).

**NewAttendance - "/dashboard/attendance/new"**

Allows administrators and authorized users to add new attendance records.

**Functionality:**

* Fill out attendance details.
* Submit to add new attendance records.

**Access:** Authenticated users (Admins, Teachers).

**EditAttendance - "/dashboard/attendance/:id/edit"**

Enables administrators and authorized users to edit existing attendance records.

**Functionality:**

* Edit attendance details.
* Save changes.

**Access:** Authenticated users (Admins, Teachers).

**ViewMarks - "/dashboard/marks"**

Displays marks and grades of students.

**Functionality:**

* View marks and grades.
* Edit marks and grades (for admins and authorized users).

**Access:** Authenticated users (Admins, Teachers).

**NewMarks - "/dashboard/marks/new"**

Allows administrators and authorized users to add new marks and grades.

**Functionality:**

* Fill out marks and grades details.
* Submit to add new marks and grades.

**Access:** Authenticated users (Admins, Teachers).

**EditMarks - "/dashboard/marks/:id/edit"**

Enables administrators and authorized users to edit existing marks and grades.

**Functionality:**

* Edit marks and grades details.
* Save changes.

**Access:** Authenticated users (Admins, Teachers).

# 8: DATA **MODELS**

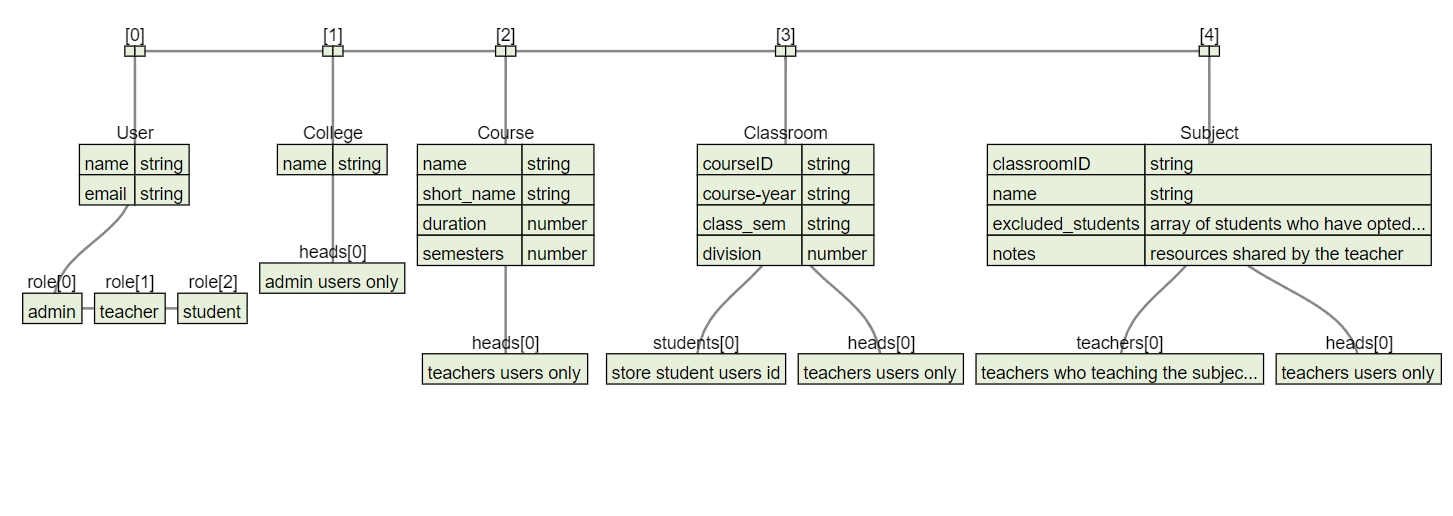
****

Fig 8.1 Data Models

In the Edex college management system, several database models are utilized to organize and manage various aspects of the college's operations. These models represent entities such as users, courses, classrooms, subjects, and the college itself. Let's explore each model and its relationships in detail:

**User Model:** The User model represents individuals who interact with the Edex college management system. Users are identified by attributes such as name and email. Additionally, users are assigned roles, including administrators, teachers, and students.

**College Model:** The College model encapsulates information about the educational institution. It includes attributes like the college's name. Furthermore, the College model maintains references to admin users who hold administrative roles within the institution.

**Course Model:** The Course model defines the academic courses offered by the college. Each course is characterized by attributes such as name, short name, duration, and the number of semesters it spans. Additionally, the Course model establishes relationships with teacher users who oversee course administration.

**Classroom Model:** The Classroom model represents physical or virtual learning environments where courses are conducted. It contains details such as the course ID, course year, class semester, and division. Moreover, the Classroom model maintains an array of student user IDs enrolled in the respective classroom. Teachers are also associated with classrooms as heads to manage classroom activities.

**Subject Model:**

The Subject model corresponds to the subjects or topics covered within a specific course. It includes attributes such as the subject name and resources, which may comprise notes or materials shared by teachers. Additionally, the Subject model maintains relationships with the Classroom model, indicating the classroom to which the subject belongs. Teachers associated with subjects teach the corresponding topics and may serve as heads for subject-related matters.

**Relationships:**

* Users can be associated with multiple roles, such as administrators, teachers, or students, allowing for flexibility in user management.
* Courses are administered by teachers who serve as heads, overseeing the course's management and operations.
* Classrooms are linked to specific courses, with teachers appointed as heads responsible for managing classroom activities.
* Subjects are associated with classrooms and taught by teachers, with designated heads overseeing subject-related matters.

# 9: AUTHENTICATION

Authentication in the Edex College Management System is implemented using JSON Web Tokens (JWT) and cookies, ensuring secure access to the application's features without the need for traditional password-based login. Additionally, email verification is utilized to enhance security and validate user identities. Below is an overview of the authentication and email verification processes:

**JSON Web Tokens (JWT) and Cookies**

* **JWT Generation**: When a user successfully logs in or registers, a JWT containing relevant user information (such as user ID, role, and expiration time) is generated on the server.
* **Cookie Storage**: The JWT is securely stored in an HTTP-only cookie on the client-side, ensuring that it is inaccessible to JavaScript and protected against cross-site scripting (XSS) attacks.
* **Authorization**: Upon subsequent requests to the server, the JWT is included in the request headers, allowing the server to authenticate and authorize the user based on the provided token.
* **Token Expiration**: JWTs have a limited lifespan, typically set to expire after a specified duration. Upon expiration, users are required to re-authenticate by obtaining a new JWT.
* **Logout**: Users can log out of the system by deleting the JWT cookie, effectively ending their session and revoking access to protected resources.

**Email Verification**

* **Registration Process**: During the user registration process, users provide their email address, which is then verified for authenticity.
* **Email Verification Link**: Upon successful registration, an email containing a unique verification link is sent to the user's provided email address.
* **Verification Link Activation**: Users must click on the verification link to activate their account and confirm their email address.
* **Token Generation**: A unique verification token is generated and associated with the user's account to ensure the validity of the verification process.
* **Account Activation**: Upon clicking the verification link, the user's account is activated, granting them access to the system's functionalities.
* **Expired Links**: Verification links have a limited validity period to prevent unauthorized activation attempts. If a link expires, users can request a new verification email to be sent.

**Benefits**

* **Enhanced Security**: JWTs and cookies provide a secure method for authenticating users without exposing sensitive information.
* **User Convenience**: Passwordless authentication via email verification simplifies the login process for users while maintaining security standards.
* **Scalability**: The authentication system is designed to scale with the growing user base of the Edex CMS, accommodating a large number of users while maintaining performance and security.

The combination of JWT-based authentication, cookie storage, and email verification ensures a seamless and secure user authentication experience within the Edex College Management System.

# 10: TECHNOLOGIES USED

**1. Next.js:**

Next.js is a React framework that simplifies the development of server-side rendered (SSR) and statically generated web applications. It offers built-in features such as automatic code splitting, routing, and server-side rendering, enhancing both developer experience and application performance.

**Benefits**:

* **Unified Development Environment**: Next.js provides a unified development environment for building both client-side and server-side rendered applications, streamlining the development process and reducing complexity.
* **Enhanced Performance**: Leveraging server-side rendering and automatic code splitting, Next.js optimizes loading times and initial page loads, resulting in improved performance and user experience.
* **SEO-Friendly**: Next.js supports server-side rendering, enabling search engines to crawl and index pages more effectively, leading to better search engine optimization (SEO) outcomes.
* **Effortless Deployment**: With built-in support for static site generation and serverless deployment options, Next.js simplifies the deployment process, allowing for seamless deployment to various hosting platforms.
* **Rich Ecosystem**: Next.js benefits from a vibrant ecosystem of plugins, tools, and community contributions, providing developers with additional functionality and resources to enhance their applications.

**2. MongoDB:**

MongoDB is a NoSQL database known for its flexibility and scalability. It utilizes a document-based model, making it ideal for storing and managing unstructured or semi-structured data. MongoDB's scalability allows it to handle large volumes of data and accommodate evolving data schemas.

**Benefits**:

* **Flexible Data Model**: MongoDB's document-oriented approach accommodates varying data structures and schema changes, providing flexibility for evolving application requirements.
* **Scalability and Performance**: MongoDB's distributed architecture and horizontal scaling capabilities ensure high availability, scalability, and performance, making it suitable for handling large-scale applications and workloads.
* **Rich Querying Capabilities**: MongoDB offers powerful querying capabilities, including support for complex queries, aggregation pipelines, and geospatial queries, enabling efficient data retrieval and analysis.
* **Developer Productivity**: MongoDB's intuitive query language, schema-less design, and rich feature set contribute to improved developer productivity and faster time-to-market for applications.
* **Community Support**: MongoDB benefits from a large and active community of users, developers, and contributors, providing resources, tutorials, and community-driven libraries to support developers in building MongoDB-powered applications.

**3. Tailwind CSS:**

Tailwind CSS is a utility-first CSS framework that streamlines the process of styling web applications. It provides a comprehensive set of pre-built utility classes that cover a wide range of design aspects, including layout, typography, and responsive design.

**Benefits**:

* **Modular and Maintainable Styling**: Tailwind CSS promotes a modular and maintainable approach to styling by abstracting common CSS patterns into reusable utility classes, reducing the need for custom CSS and enhancing code maintainability.
* **Rapid Prototyping**: With its extensive set of utility classes, Tailwind CSS enables rapid prototyping and iteration, allowing developers to quickly experiment with different design variations and iterate on UI/UX components.
* **Customizability**: Tailwind CSS offers customization options through configuration files and plugin integrations, allowing developers to tailor the framework to their specific design requirements and preferences.
* **Performance Optimization**: Tailwind CSS generates optimized CSS bundles by purging unused styles during the build process, resulting in smaller file sizes and faster loading times for web applications.
* **Community and Ecosystem**: Tailwind CSS benefits from an active community and ecosystem of plugins, themes, and resources.

**4. Nodemailer:**

Nodemailer is a popular and widely used module in the Node.js ecosystem that facilitates the sending of email messages directly from Node.js applications. It provides a simple and intuitive interface for composing, formatting, and delivering email messages using SMTP or other transport mechanisms.

**Benefits:**

1. **Ease of Use:** Nodemailer is designed to be easy to use, with a straightforward API that allows developers to quickly integrate email functionality into their Node.js applications.
2. **Multiple Transport Options:** Nodemailer supports various transport options for sending emails, including SMTP, sendmail, and Amazon SES (Simple Email Service), among others. This flexibility allows developers to choose the most suitable transport method based on their requirements and infrastructure setup.
3. **Comprehensive Email Composition:** With Nodemailer, developers can compose richly formatted email messages using HTML, plain text, or a combination of both. It supports the inclusion of attachments, inline images, and custom headers to meet diverse email communication needs.

# 11: TESTING REPORT

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NO.** | **TEST** | **TEST SCENARIO** | **EXPECTED RESULT** | **ACTUAL RESULT** | **PASS/FAIL** |
| 1 | CollegeSetup & UserLogin | Attempt to access the login page before college setup is completed. | The system should redirect to the College Setup page. | Upon accessing the login page, the system redirects to the College Setup page as expected. | Pass |
| 2 | VerifyLogin | Verify the verification of JWT token from the email link. | Upon clicking the verification link, the system should validate the JWT token and confirm the user's login. | The system successfully validates the JWT token from the email link and allows the user to log in. | Pass |
| 3 | Dashboard | Check the functionality of the dashboard after logging in. | After logging in, the dashboard should display relevant modules and options for managing the college. | Upon logging in, the dashboard displays the relevant modules and options for college management. | Pass |
| 4 | Profile | View the user profile page. | The user profile page should display the details of the logged-in user, such as name, email, and role. | The user profile page successfully displays the details of the logged-in user, including name, email, and role. | Pass |
| 5 | ViewCourses | Access the page displaying all courses. | The page should list all the courses available in the college. | The page successfully displays all available courses in the college. | Pass |
| 6 | NewCourse | Create a new course. | The form should allow the admin user to create a new course with relevant details. | The form successfully allows the admin user to create a new course with all required details. | Pass |
| 7 | EditCourse | Edit an existing course. | The edit form should allow authorized users to modify the details of an existing course. | The edit form successfully allows authorized users to modify the details of an existing course. | Pass |
| 8 | ViewCourse | Access the details of a specific course. | Clicking on a course should display its details, including classrooms and options to edit or add classrooms. | The details of the specific course are displayed, along with options to edit or add classrooms. | Pass |
| 9 | NewClassroom | Create a new classroom for a course. | The form should allow authorized users to create a new classroom under a specific course. | Authorized users can successfully create a new classroom under a specific course using the form. | Pass |
| 10 | EditClassroom | Edit an existing classroom. | Authorized users should be able to modify the details of an existing classroom using the edit form. | Authorized users can successfully modify the details of an existing classroom using the edit form. | Pass |
| 11 | ViewClassroom | Access the details of a specific classroom. | Clicking on a classroom should display its details, including subjects and options to edit or add subjects. | The details of the specific classroom are displayed, along with options to edit or add subjects. | Pass |
| 12 | NewSubject | Create a new subject for a classroom. | The form should allow authorized users to create a new subject under a specific classroom. | Authorized users can successfully create a new subject under a specific classroom using the form. | Pass |
| 13 | EditSubject | Edit an existing subject. | Authorized users should be able to modify the details of an existing subject using the edit form. | Authorized users can successfully modify the details of an existing subject using the edit form. | Pass |
| 14 | ViewSubject | Access the details of a specific subject. | Clicking on a subject should display its details, including enrolled students and resources. | The details of the specific subject are displayed, along with enrolled students and resources. | Pass |
| 15 | ViewTeachers | View the list of teachers. | The page should display all the teachers in the college, along with their details and options to edit. | The page successfully displays the list of teachers in the college, with options to edit. | Pass |
| 16 | NewTeacher | Add a new teacher to the system. | The form should allow authorized users to add a new teacher to the system with relevant details. | Authorized users can successfully add a new teacher to the system using the form. | Pass |
| 17 | ViewStudents | View the list of students. | The page should display all the students in the college, along with their details and options to edit. | The page successfully displays the list of students in the college, with options to edit. | Pass |
| 18 | ViewAttendance | Access the attendance records. | The page should display the attendance records of students, with options to view and edit. | The page successfully displays the attendance records of students, with options to view and edit. | Pass |
| 19 | ViewMarks | Access the marks of students. | The page should display the marks of students, with options to view and edit. | The page successfully displays the marks of students, with options to view and edit. | Pass |
| 20 | EditMarks | Check the editing of marks functionality. | Marks for a specific student should be editable. | The marks for specific student are editable | Pass |
| 21 | NewMarks | Check the creation of new marks functionality. | Marks for a new student should be saved successfully. | Marks for a new student saved successfully. | Pass |

# 12: USER MANUAL

Welcome to Edex CMS! This manual will guide you through the basic functionalities of the system.

**1. College Setup**

* If this is your first time accessing Edex CMS, you'll need to set up your college.
* Visit the Edex CMS login page and click on the "College Setup" option.
* Follow the prompts to enter your college details such as name and other relevant information.
* Once the setup is complete, you'll be redirected to the login page to access the dashboard.

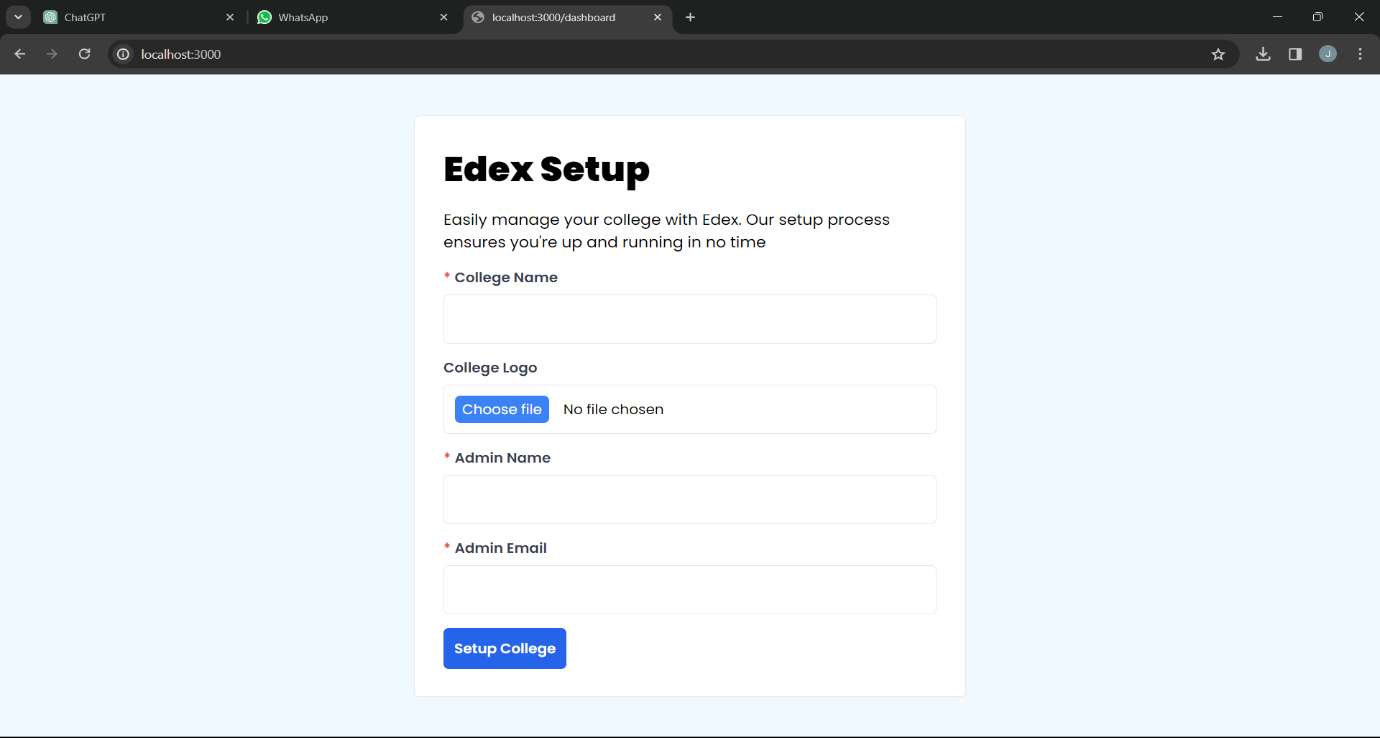


Fig 12.1 College Setup Page

**2. Logging In**

* After setting up your college, access the Edex CMS login page again.
* Enter your credentials (email and password) that you used during the college setup process.
* Click on the "Login" button to access the dashboard.

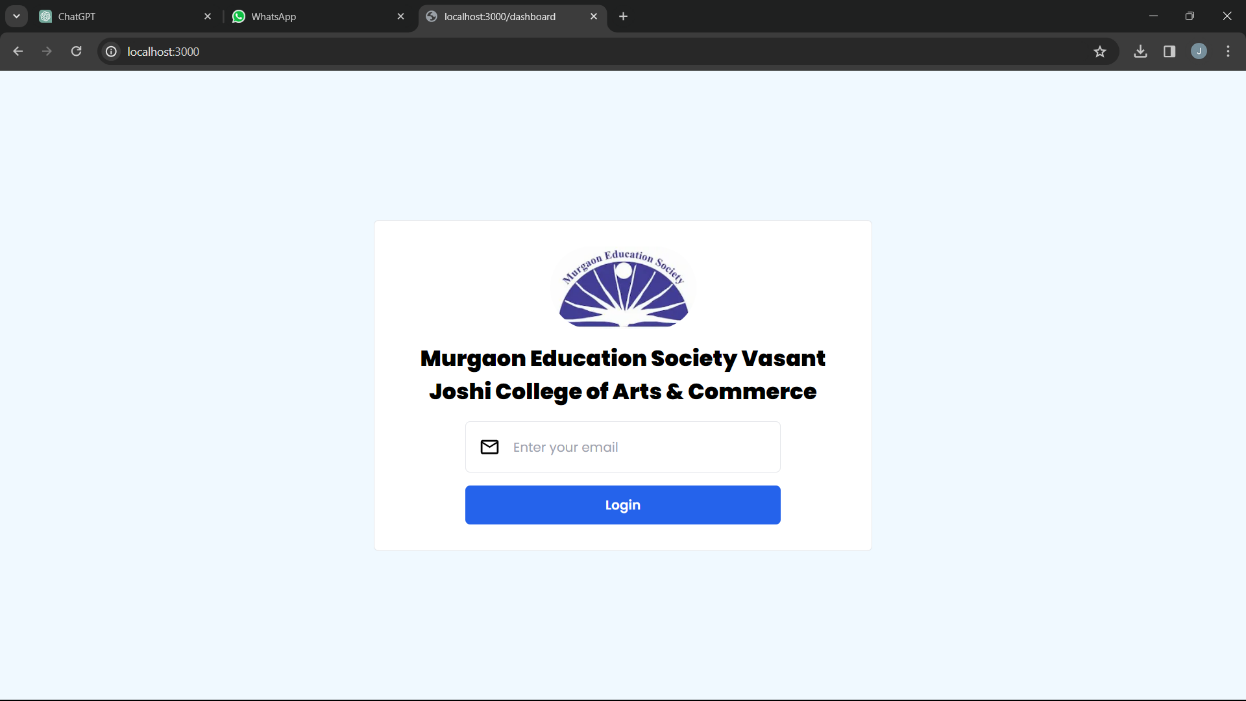


Fig 12.2 User Login Page

**3. Dashboard Overview**

* Upon successful login, you'll be directed to the dashboard.
* The dashboard provides access to various modules and features of Edex CMS.

**4. Managing Courses**

* Click on "Courses" in the dashboard menu.
* View existing courses or add a new course.
* Edit course details as needed.

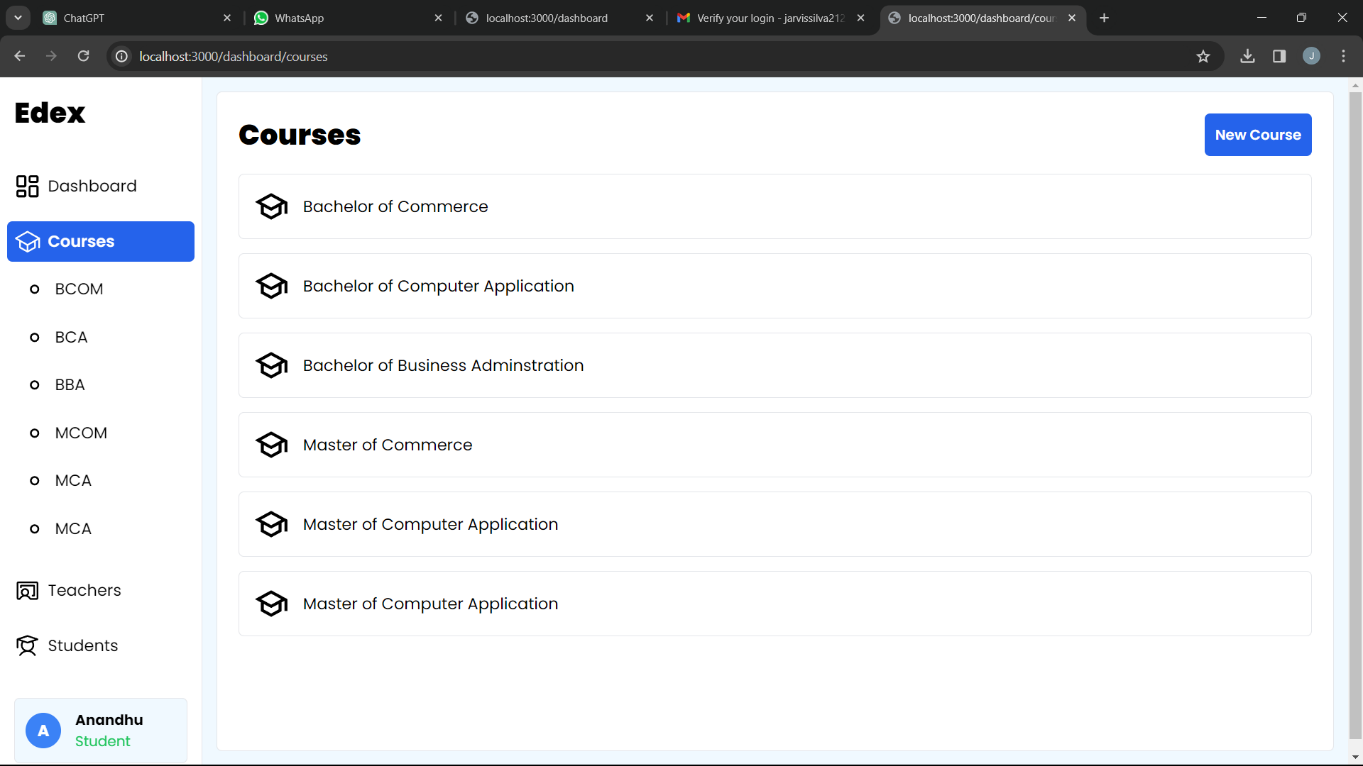


Fig 12.3 Course Manage Page

**5. Managing Classrooms**

* Navigate to the "Classrooms" section.
* Add new classrooms or edit existing ones.
* View subjects associated with each classroom.

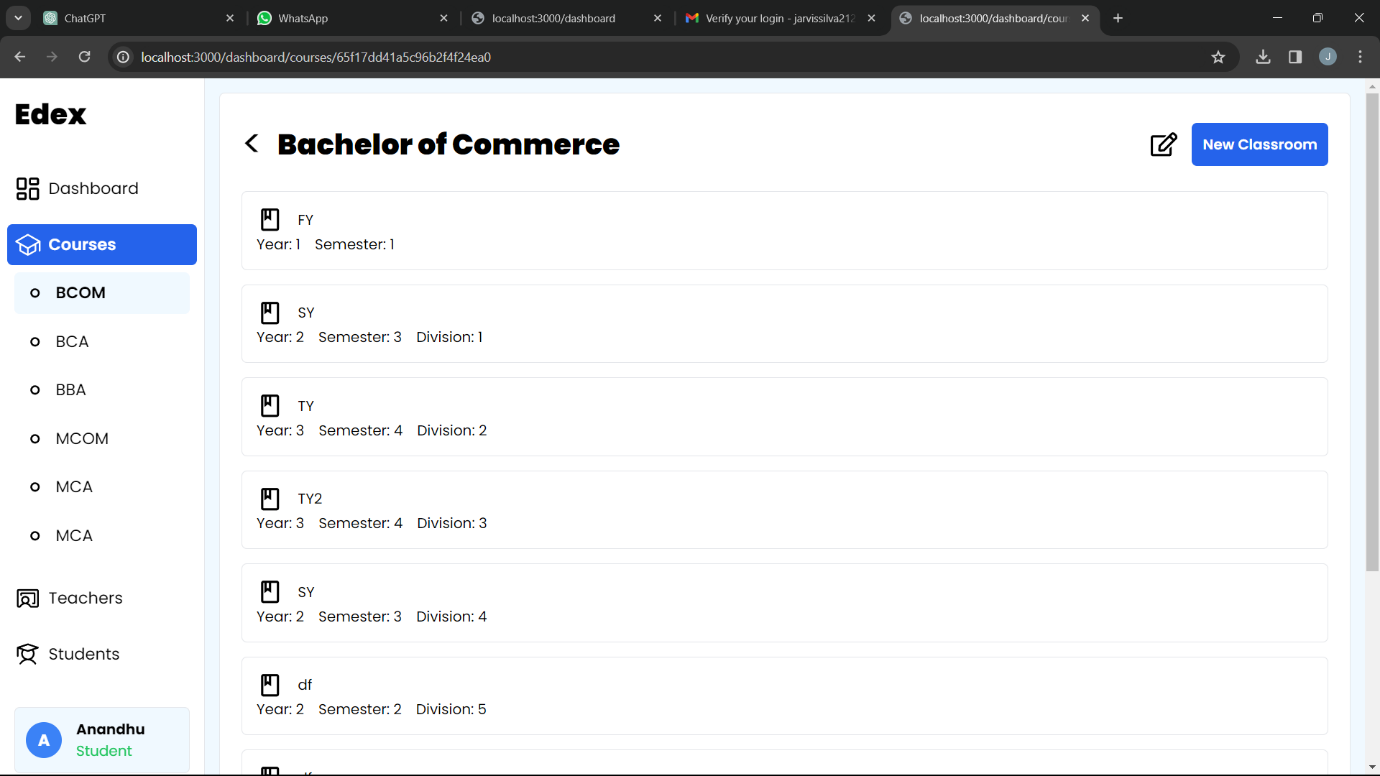


Fig 12.4 Classroom Manage Page

**6. Managing Subjects**

* Access the "Subjects" tab.
* Add new subjects or modify existing ones.
* View subject details and associated resources.

**7. Managing Users**

* Navigate to the "Users" section.
* Add new users (teachers, students) or edit existing ones.
* View user profiles and roles.

**8. Attendance Management**

* Access the "Attendance" module.
* Add new attendance records or edit existing ones.
* View attendance details for students.

**9. Marks Management**

* Go to the "Marks" section.
* Add new marks/grades for students or edit existing ones.
* View marks/grades details and student performance.

**10. Profile Settings**

* Click on your profile icon to access profile settings.
* View and update your personal information.

**11. Logout**

* To logout, click on the logout button located in the profile settings.

# 13: FUTURE ENHANCEMENTS

While Edex College Management System provides a comprehensive set of features to meet the current needs of educational institutions, there are several areas where future enhancements can further improve its functionality, usability, and effectiveness. Some potential areas for enhancement include:

1. **Insights and Analytics**: Enhance the system with advanced analytics and reporting capabilities to provide administrators, teachers, and students with valuable insights into attendance patterns, academic performance, and overall progress. Implement data visualization tools to present information in a clear and actionable format.
2. **Notifications and Reminders**: Implement a notification system to keep users informed about important events, deadlines, and updates related to courses, classes, assignments, and exams. Allow users to customize their notification preferences based on their role and preferences.
3. **Collaborative Tools**: Introduce collaborative features that enable teachers and students to collaborate on projects, share resources, and communicate effectively within the platform. This could include discussion forums, chat functionality, and document sharing capabilities.
4. **Mobile Application**: Develop a mobile application for Edex to provide users with access to key features and functionality on their smartphones and tablets. A mobile app would improve accessibility and convenience, allowing users to stay connected and productive while on the go.
5. **Integration with Learning Management Systems (LMS)**: Explore opportunities to integrate Edex with popular learning management systems (LMS) such as Moodle, Canvas, or Blackboard. This would allow for seamless transfer of course materials, assignments, grades, and other data between systems.
6. **Parent Portal**: Create a dedicated portal for parents or guardians to access information about their child's academic progress, attendance, and other relevant details. Provide parents with the ability to communicate with teachers and administrators and stay informed about important events and announcements.
7. **Accessibility and Internationalization**: Ensure that Edex is accessible to users with disabilities by adhering to web accessibility standards (e.g., WCAG). Additionally, support for multiple languages and localization features can make the platform accessible to users from diverse linguistic backgrounds.
8. **Integration with Student Information Systems (SIS)**: Integrate Edex with existing student information systems (SIS) used by educational institutions to streamline data management and ensure consistency across systems. This would facilitate seamless transfer of student records, course schedules, and other relevant information.
9. **Gamification and Engagement**: Incorporate gamification elements into the platform to enhance user engagement and motivation. This could include achievements, badges, leaderboards, and other gamified features that incentivize active participation and learning.
10. **Financial Management:** Enhance the financial management module to include budgeting, expense tracking, and revenue forecasting capabilities. Integration with accounting software for seamless handling of payroll, invoicing, and financial reporting.
11. **Continuous Improvement and Feedback**: Establish mechanisms for collecting user feedback and suggestions for improvement on an ongoing basis. Regularly review and prioritize feature requests and bug reports to ensure that Edex remains relevant, reliable, and user-friendly.

By focusing on these future enhancements, Edex can continue to evolve and adapt to the changing needs of educational institutions, empowering them to achieve their objectives and deliver an exceptional learning experience to students and faculty alike.

# 14: CONCLUSION

In conclusion, the Edex College Management System represents a comprehensive solution designed to streamline and enhance various aspects of college administration. Throughout this project report, we have explored the objectives, scope, architecture, and key features of the system, highlighting its significance in addressing the challenges faced by educational institutions in managing courses, classrooms, subjects, attendance, marks, and resources effectively.

The development of Edex was driven by the need to create a user-friendly, scalable, and feature-rich platform that caters to the diverse needs of college administrators, teachers, and students. By leveraging technologies such as Next.js, MongoDB, Tailwind CSS, and JSON Web Tokens (JWT) with cookies for authentication, we have built a robust and modern application that offers a seamless user experience while ensuring security and performance.

The architecture of Edex is designed to be modular, extensible, and scalable, allowing for future enhancements and integrations with third-party services. With a clear separation of concerns between frontend and backend components, adherence to design patterns and principles, and implementation of security best practices, Edex provides a solid foundation for managing college operations efficiently.

Looking ahead, Edex has the potential for further expansion and refinement, with opportunities to integrate additional features such as insights, notifications, and collaborative tools for teachers and students. By embracing innovation and continuous improvement, Edex aims to stay at the forefront of college management systems, empowering educational institutions to achieve their goals and deliver a superior learning experience to students.

In conclusion, the successful development and deployment of Edex represent a significant milestone in the journey towards modernizing college administration and fostering academic excellence. As we continue to iterate and evolve, we remain committed to delivering value to our stakeholders and making a positive impact on the education sector.

# 15: REFERENCES

* + <https://chat.openai.com>
  + <https://www.academia.edu/35401042/A_Research_Paper_on_College_Management_System>
  + <https://github.com>
  + <https://www.geeksforgeeks.org/college-management-system-using-django-python-project/>
  + <https://youtu.be/XIYdVPKQpiQ?si=8njpBze65Zs-eESz>
  + <https://w3schools.com>

# 16: GANTT CHART

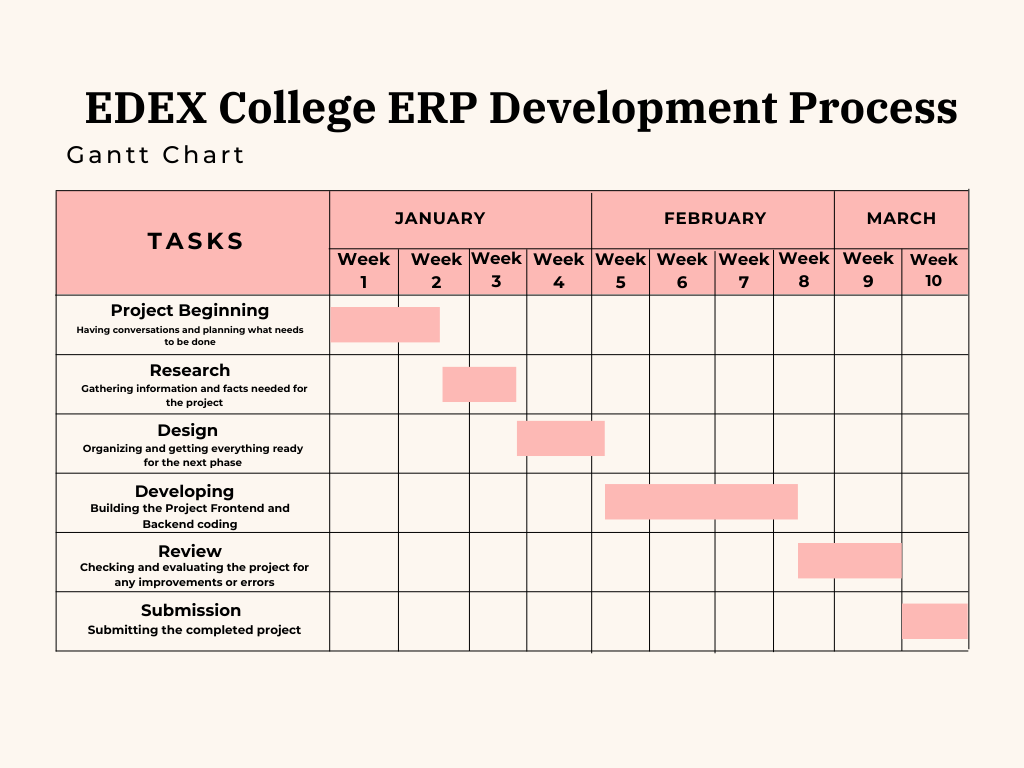


Fig no 16.1 Gantt Chart