**Shri B.V.V. Sangha’s**

**BASAVESHWAR ENGINEERING COLLEGE, BAGALKOTE**

**[AICTE Approved, Government-Aided Institution Affiliated To VTU, Belagavi]**

****

**DEPARTMENT OF MCA**

#### **Mini-Project Synopsis on**

**“Academic Task Management System (ATMS)”**

***Submitted in partial fulfillment of requirement for the award of the degree***

**MASTER OF COMPUTER APPLICATIONS**

**2023-2024**

**Project Guide           CoordinatorHOD**

**Prof. S.S.Gujarathi Prof. C. M. Jangin Prof. Sudha. K. S**

**Associate Prof. Dept. of MCA  Asst Prof. Dept. of MCA Head, Dept. of MCA**

**BEC, Bagalkote BEC, Bagalkote**

***Submitted by:***

**Student Name 1:Cheta Mallappa Melavanki  USN:2BA23MC008**

**Student Name 2:Munni R Hanchwale                                USN:2BA23MC026**

**Student Name 3:Prutwiraj Hiremani                                USN:2BA23MC033**

****

**Shri B.V.V. Sangha’s**

**BASAVESHWAR ENGINEERING COLLEGE, BAGALKOTE**

**[AICTE Approved, Government Aided Institution Affiliated To VTU, Belagavi]**

***Department of MCA***

**CERTIFICATE**

This is to certify that **Chetan Mallappa Melavanki (2BA23MC008)**, **Munni R Hanchwale (2BA23MC026)**, and **Prutwiraj Hiremani (2BA23MC033)**, students of 2nd semester MCA at Basaveshwar Engineering College (Autonomous), Bagalkot, have satisfactorily completed the mini-project entitled **“Academic Task Management System (ATMS)”** as part of the partial fulfillment of the requirements for the **Master of Computer Applications** program, as prescribed by Basaveshwar Engineering College (Autonomous), Bagalkot, during the academic year 2023-2024.

**Project Guide           CoordinatorHOD**

**Prof. S.S.Gujarathi Prof. C. M. Jangin Prof. Sudha. K. S**

**Asst Prof. Dept. of MCA  Asst Prof. Dept. of MCA Head, Dept. of MCA**

**BEC, Bagalkote BEC, Bagalkote**

**Name of Examiner                                       Signature of Examiner**

**1……………………                                      1………………………**

**2…………………....                                     2………………………**

**DECLARATION**

We, **Chetan Mallappa Melavanki (2BA23MC008)**, **Munni R Hanchwale (2BA23MC026)**, and **Prutwiraj Hiremani (2BA23MC033)**, students of 2nd semester MCA at Basaveshwar Engineering College (Autonomous), Bagalkot, hereby declare that the mini-project entitled **“Academic Task Management System (ATMS)”** has been carried out independently by us as a team at the Master of Computer Applications Department, Basaveshwar Engineering College (Autonomous), Bagalkot.

This mini-project is submitted in partial fulfillment of the requirements for the award of the degree **MASTER OF COMPUTER APPLICATIONS** by Basaveshwar Engineering College (Autonomous), Bagalkot, during the academic year 2023-2024.

**Place: Baglkot.**  **Student Name 1: Cheta Mallappa Melavanki**

**Date: USN: 2BA23MC008**

**Student Name 2: Munni R Hanchwale**

**USN: 2BA23MC026**

**Student Name 3: Prutwiraj Hiremani**

**USN: 2BA23MC033**

**AKNOWLEDGEMENT**

"I take this moment to express my heartfelt gratitude to those who have been integral to the success of this Mini-Project, offering unwavering support since its inception.

I extend my sincere thanks to **Dr. Basavaraj Herakal**, our esteemed Principal, for his invaluable support throughout this endeavor.

I am deeply grateful to **Mrs. Sudha K. S.**, Head of the Department, for her continuous encouragement and guidance. Her dedication to our growth and success has been truly inspiring.

Special appreciation goes to my guide, **Prof. Shrinivas S. Gujarathi**, for his invaluable cooperation, encouragement, and esteemed guidance. His expertise and mentorship have been instrumental in navigating the challenges of this Mini-Project.

Additionally, I would like to express my gratitude to the seminar coordinator, **Prof. C. M. Jangin**, for his overall support in organizing and facilitating this project. His commitment to its success has been commendable.

Once again, I sincerely thank everyone for their unwavering support and guidance. I am truly honored to have had the opportunity to work on this Mini-Project and grow through the experience."

**Yours Sincerely,**

**Student Name 1: Cheta Mallappa Melavanki**

**USN: 2BA23MC008**

**Student Name 2: Munni R Hanchwale**

**USN: 2BA23MC026**

**Student Name 3: Prutwiraj Hiremani**

**USN: 2BA23MC033**

**Abstract:**

The Academic Task Management System is innovative software designed to record, manage, and report on various tasks conducted by academic institutions or departments. Academic institutions generate substantial amounts of data, including information on employees, supported branches, students, subjects offered, events conducted, faculty training programs, student training, and assessments.

Effectively managing this data requires a robust procedure for recording these processes and providing relevant information, such as branch details, faculty details, student details, branch-wise activity details, faculty training details, student training details, and student assessment details. The system aims to streamline these operations, ensuring that data is accurately captured and easily accessible.

A critical feature of the system is its ability to generate insights that highlight faculty and student performance. Additionally, the system's forecasting capabilities aim to predict student performance, allowing for proactive measures to enhance educational outcomes. By integrating AI models, the application not only manages academic details but also provides advanced analytics and foresight.

The AI models further enhance the system by enabling predictive analytics, personalized learning pathways for students, and automated scheduling and resource allocation. These models can identify trends and patterns in student performance, suggest tailored interventions for at-risk students, and optimize faculty workload distribution. The incorporation of machine learning algorithms allows for continuous improvement and adaptation to the evolving needs of the institution.

This system is designed to be an essential tool for academic institutions, improving efficiency, providing valuable insights, and supporting data-driven decision-making processes.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Title** | **Page no**. |
|  | **Certificate**  **Declaration**  **Acknowledgement**  **Abstarct** |  |
| 1 | Introduction | 1-2 |
| 2 | Literature survey  2.1 Existing System  2.2 Proposed System | 3-4 |
| 3 | Problem Statement  3.1 Problem Definition  3.2 Challenges  3.3 Limitations | 5-7 |
| 4 | Software Requirement and Specification (H/w or S/w) | 8-15 |
| 5 | System Design  5.1 Methdology  5.2 Module Design  5.3 Detailed descriptions | 16-23 |
| 6 | System Implementation | 24-32 |
| 7 | System Testing and Results | 33-36 |
| 8 | Future Enhancement and Conclusion | 37 |
| 9 | Document References and Web page links | 38 |
| 10 | Index | 39 |

**1. INTRODUCTION**

In this project, the **Academic Task Management System** aims to address the growing need for effective management of academic-related activities, primarily focusing on educational institutions. Before delving into the specifics of the project, it is crucial to understand the domain and organizational environment where the system will be implemented.

#### **Domain of Work**

The domain of this project is **Educational Management Systems**, which encompasses tools and platforms designed to streamline academic operations. This domain integrates technology to enhance the administration and coordination of academic tasks such as scheduling, assignment tracking, evaluation, and communication among stakeholders.

**What is the Domain?** The Educational Management domain focuses on managing academic activities efficiently using technology. It supports institutions by reducing manual workloads, improving task visibility, and ensuring effective communication between students, faculty, and administrators.

**How Things Are Involved in the Domain?**

* 1. **Academic Operations**: Timetable management, assignment distribution, and resource allocation.
  2. **Communication**: Seamless interaction between students, faculty, and administrators.
  3. **Task Monitoring**: Tracking deadlines, evaluating submissions, and providing feedback.

**Focus of the Domain**  
The primary focus is to optimize resource utilization, enhance transparency, and ensure timely task completion. It supports structured workflows that simplify the management of academic processes.

**How Things Are Carried Out?**

* 1. Faculty assigns tasks or projects.
  2. Students complete tasks within deadlines and submit them for evaluation.
  3. Administrators oversee processes, ensuring compliance with institutional standards.

#### **Organization Overview**

This system is being designed for **educational institutions** such as schools, colleges, or universities. These organizations handle a variety of academic tasks, which include:

**Products or Services Provided**

* 1. Education delivery across various levels and streams.
  2. Online and offline academic content.
  3. Administrative support for academic coordination.

**Workflow or Process Flow**

* 1. Tasks are allocated by faculty and monitored by administrators.
  2. Students interact with the system to access tasks, submit work, and review feedback.
  3. Administrators ensure task records are maintained and accessible.

**Structure of Organization**

* 1. **Branches**: Academic departments handling specialized domains (e.g., science, arts, management).
  2. **Employees**: Faculty, administrative staff, IT personnel, and management.
  3. **Technology Infrastructure**: Existing systems for attendance, grading, and academic records.

#### **Current Functioning and Issues**

Currently, many institutions rely on manual methods or fragmented tools for managing academic tasks. These practices lead to several challenges:

* **Inefficiency**: Delays in task allocation and monitoring.
* **Limited Transparency**: Lack of clarity in communication between stakeholders.
* **Scattered Resources**: Difficulty in managing academic materials and submissions.

## ****2. LITERATURE SURVEY****

### **2.1 EXISTING SYSTEM**

The existing systems used by academic institutions for data management are often outdate and inefficient. Many institutions still rely on manual processes for recording and managing data. These processes, such as using spread sheets and paper records, lead to fragmented, inaccurate, and labour-intensive workflows. The lack of integration between different systems further exacerbates these issues.

#### **Key Characteristics of the Existing System:**

* **Manual Data Entry:**
  + **Spread sheets (Excel, Google Sheets):** These are commonly used for storing student data, faculty information, course details, attendance records, and grades. Administrative staff or faculty members manually input and update the data, which can lead to frequent errors.
  + **Paper Records:** Some institutions still depend on physical forms for student enrolment, attendance sheets, and other academic records. These records are often transcribed into digital formats when needed, creating opportunities for transcription errors.
* **Data Storage:**
  + **Standalone Databases:** Many institutions use basic databases (e.g., Microsoft Access) or simple file storage systems. These databases are not interconnected with other systems, leading to data silos and duplication.
  + **File Systems:** Data is often stored in multiple, disconnected file directories. For example, student data might be stored in one folder, faculty records in another, and course schedules in yet another. This lack of integration makes accessing data difficult and can result in redundancies.

### **2.2 PROPOSED SYSTEM**

The proposed **Academic Task Management System (ATMS)** seeks to modernize and streamline academic operations, addressing the limitations of the existing system by integrating various functions and automating many manual processes. This system promises to improve data accuracy, reduce human errors, and enable advanced analytics for better decision-making.

#### **Key Features of the Proposed System:**

1. **Centralized Data Management:**
   * **Unified Repository:** All data related to students, faculty, courses, events, and assessments is stored in a single, accessible platform. This eliminates the need for fragmented systems and makes it easier to maintain a holistic view of the institution’s data.
   * **Automated Data Entry:** Instead of manually entering data, the system automates data collection and input by integrating with existing data sources (e.g., student registration forms, faculty profiles). This reduces manual labour and human errors.
2. **Advanced Reporting and Analytics:**
   * **Comprehensive Reporting:** The system provides the ability to generate detailed, customized reports that can be tailored to the institution’s needs. For example, reports on student performance, faculty productivity, and academic events can be easily generated with real-time data.
   * **Advanced Analytics:** The ATMS will support advanced analytics, such as trend analysis, performance forecasting, and predictive models. This allows the institution to identify potential issues early and plan interventions before problems escalate.
3. **Increased Efficiency:**
   * **Faster Data Access:** The system enables quick access to data across departments, significantly reducing the time spent searching for information.
   * **Reduced Errors:** Automated processes and centralized data reduce the likelihood of human error, leading to more accurate records.
   * **Greater Storage Capacity:** With cloud-based or integrated server solutions, the system can handle large volumes of data, allowing the institution to scale without worrying about storage limitations.
4. **User-Friendly Interface:**
   * **Easy Navigation:** The system’s interface is designed to be intuitive, making it easy for users (administrators, teachers, students) to navigate and access the information they need without technical expertise.

**3. Problem statement**

### **3.1 What is a Problem Definition?**

A **problem definition** is a clear and concise statement that describes the issue or challenge the project seeks to address. It outlines the current situation, identifies gaps or inefficiencies, and explains why resolving this problem is essential. The problem definition helps set the context for the project and ensures all stakeholders understand its purpose.

### **Problem Definition for Academic Task Management System**

In educational institutions, managing academic tasks such as assignments, deadlines, evaluations, and communication among students, faculty, and administrators is a complex process. Currently, many institutions rely on manual methods or disconnected tools, leading to several challenges:

**Inefficient Processes**

* 1. Manual allocation and tracking of tasks result in delays and errors.
  2. Faculty and administrators struggle to keep track of multiple activities simultaneously.

**Lack of Transparency**

* 1. Students face difficulties in accessing task-related information and feedback.
  2. Communication between stakeholders is often unclear or delayed, impacting productivity.

**Resource Mismanagement**

* 1. Academic materials, deadlines, and task updates are scattered across multiple platforms or remain inaccessible.
  2. Manual record-keeping leads to inconsistencies and additional administrative overhead.

**Overburdened Faculty and Administrators**

* 1. Without a centralized system, faculty and administrators spend significant time coordinating and tracking academic tasks, diverting focus from core academic responsibilities.

**Impact on Academic Performance**

* 1. Inefficient task management affects student engagement and learning outcomes.
  2. Lack of timely feedback and clarity leads to missed deadlines and lower-quality submissions.

#### **Problems in Data Management:**

* **Data Accuracy & Integrity:** Manual data entry is error-prone, and multiple sources of data can result in conflicting information.
* **Limited Accessibility:** Manual processes require staff to spend time searching through files and spread sheets, which can delay decision-making and reduce efficiency.
* **Time-Consuming:** Data updates are performed manually, consuming significant time and resources from administrative staff.
* **Data Fragmentation:** Different departments or units within the institution may maintain their own separate records, making it difficult to access a holistic view of the institution’s data.

#### **Reporting and Analytics:**

* **Limited Reporting Capabilities:** Existing systems rely on manual reporting processes, such as extracting data from spread sheets, compiling it, and generating reports. This is a time-consuming process and lacks flexibility in customizing reports.
* **Basic Analytics:** Current systems only support basic data analysis, such as summing totals or calculating averages. More sophisticated analyses, such as performance trends or forecasting, are not supported.

#### **Resource Allocation:**

* **Manual Scheduling:** Scheduling of courses, exams, and faculty assignments is often done manually, using paper-based planners or spread sheets. This can lead to errors, conflicts, and inefficiencies, especially as the institution scales up.

### **3.2 Challenges:**

#### **Data Collection:**

* **Manual Data Entry:**
  + **Spread sheets & Paper Records:** Both spread sheets and paper records require significant manual effort to input and update data. Even minor changes need to be reflected across multiple systems, which increase the risk of inconsistencies.
* **Data Storage:**
  + **Standalone Databases & File Systems:** The lack of integration means that data stored in one place (e.g., faculty data in one file) cannot easily be linked with other records (e.g., student grades). As a result, retrieving and consolidating information is difficult and time-consuming.

#### **Data Management:**

* **Updating Records:**
  + **Manual Updates:** The need to update records by hand in spread sheets or databases increases the likelihood of human errors. A change made in one document may not be properly reflected in another, leading to inconsistencies across the institution’s data.
* **Data Retrieval:**
  + **Manual Searches:** Finding specific pieces of data is a manual and time-consuming process. Staff have to search through large spread sheets or file systems to locate the necessary information, which can result in delays and mistakes.

#### **Reporting and Analysis:**

* **Basic Reporting:**
  + **Custom Reports:** Generating reports requires extracting data manually and compiling it, which is slow and prone to errors. Custom reports are created by manually manipulating data, but these reports lack depth and flexibility.
* **Limited Analytics:**
  + **Basic Analysis:** Simple statistics like averages or totals can be computed, but advanced analytics (e.g., performance trends, predictive models) are not available in these systems.

#### **Resource Allocation:**

* **Manual Scheduling:**
  + **Scheduling Conflicts:** Scheduling classes, exams, and faculty assignments manually can result in errors, such as double-booked rooms, overlapping class schedules, or understaffed courses. This inefficiency grows as the institution expands, making it harder to manage resources effectively.

#### **3.3 Limitations of the Existing System:**

* **Inefficiency:**
  + Time-consuming, labour-intensive processes hinder administrative productivity and prevent staff from focusing on more strategic tasks.
* **Error-Prone:**
  + Human errors in manual data entry, corrections, and updates create inaccuracies in institutional records.
* **Fragmented Data:**
  + With data stored in separate files or databases, it is difficult to maintain a unified view of institutional data, making decision-making harder and slower.
* **Limited Reporting and Analytics:**
  + Reporting capabilities are basic and manually driven, which limits the institution’s ability to conduct in-depth analysis, track trends, or plan for future needs effectively.
* **Difficulty in Scalability:**
  + As institutions expand, manual processes become increasingly difficult to scale. The administrative burden increases, leading to delays, inefficiencies, and possible errors in resource management.

## ****4. SOFTWARE REQUIREMENT SPECIFICATIONS****

### **Purpose and Scope**

**Purpose:**  
The purpose of this Software Requirements Specification (SRS) document is to define the core requirements for the software application to be developed. This document serves as a reference for the development team, stakeholders, and testing team throughout the lifecycle of the project. It outlines the necessary features, functionalities, and specifications that the software must meet to ensure it fulfils the needs of the users and the institution effectively.

**Scope:**  
The proposed software system will be a web-based application designed to provide robust functionality for managing academic data and tasks. The system will enable seamless interaction between users, including students, faculty, and administrators, while offering powerful data management tools, reporting capabilities, and performance analytics. The application will be built using modern web technologies like HTML, CSS, JavaScript, and will use Java on the server side. The system will run on Apache Tomcat as the server, with MySQL used for data storage and retrieval. It will be compatible with Windows 7 and above and developed using Visual Studio Code as the primary Integrated Development Environment (IDE).

### **Product Overview**

This software system will serve as a comprehensive Academic Task Management System (ATMS), providing the functionality needed to manage and analyse academic data effectively. The system will allow users to manage their profiles, access educational resources, perform CRUD operations on student and faculty data, generate reports, and track academic performance. The software will facilitate better communication, decision-making, and resource allocation through efficient data management and reporting.  
 The application will be hosted on an Apache Tomcat server, ensuring scalability and performance. It will interact with a MySQL relational database to persist and retrieve data. The user interface will be responsive and user-friendly, ensuring accessibility across a wide range of devices, from desktop computers to mobile devices.

**Employee Management:**

**Objective**: Efficiently track and manage all faculty and staff details to ensure the smooth functioning of the institution.

* **Faculty and Staff Profiles**: Storing key personal and professional details like name, contact information, department, position, qualifications, years of experience, and roles. This could also include tenure or contract status.
* **Roles and Responsibilities**: Tracking each employee’s assigned duties, such as teaching assignments, administrative roles, or committee memberships.
* **Performance Tracking**: Monitoring faculty and staff performance through evaluations, feedback, and assessments. This helps identify high performers and areas where improvement is needed.
* **Career Development**: Tracking professional development opportunities, such as workshops, certifications, and promotions, to ensure faculty and staff continue to grow in their roles.

**Why it's important**: Ensures an organized and up-to-date record of all staff, which is crucial for making informed decisions about staffing, budgeting, and institutional growth.

**Student Data Management:**

**Objective**: Systematically manage student information to support academic progress and track performance.

* **Student Profiles**: Storing personal information, including contact details, admission data, academic history, and extracurricular involvement.
* **Academic Progress**: Tracking students' grades, attendance, and completed courses. This includes GPA, subject-specific performance, and whether the student is on track to graduate.
* **Assessments and Feedback**: Documenting test scores, assignments, projects, and feedback from instructors. It also includes tracking any remedial or special academic support students may need.
* **Engagement and Participation**: Monitoring students' involvement in campus life, events, and other non-academic activities, which can help identify leadership or career potential.

**Why it's important**: Helps schools keep a comprehensive record of each student’s academic journey, making it easier to monitor progress, address challenges early, and provide the right support for student success.

**Course and Subject Management:**

**Objective**: Organize, schedule, and manage the academic curriculum to ensure it aligns with the institution’s goals.

* **Course Cataloguing**: Creating and maintaining a database of all available courses, including course codes, descriptions, prerequisites, and objectives.
* **Scheduling**: Managing the timetable to ensure courses are available at suitable times, considering faculty availability and student demand. This includes coordinating with departments to avoid class conflicts and ensure optimal classroom usage.
* **Curriculum Development**: Reviewing and updating course content to reflect new academic trends, research, or industry needs. This may also involve managing subject-specific resources such as textbooks, software, or labs.
* **Course Enrolment**: Tracking student registrations for courses, making sure class sizes are appropriate, and managing waitlists if courses are over-subscribed.

**Why it's important**: Keeps academic offerings organized, ensures courses are taught efficiently, and helps ensure students have access to the courses they need to complete their degrees on time.

**Event and Training Management:**

**Objective**: Organize and document academic events and professional development opportunities for both students and faculty.

* **Academic Events**: Managing events like conferences, seminars, guest lectures, workshops, and student presentations. This includes tracking event details (date, location, speakers, etc.) and registration.
* **Faculty Development**: Organizing training programs for faculty to stay current with teaching strategies, technology tools, and research methods. This could include workshops, certifications, or peer teaching sessions.
* **Student Development Programs**: Coordinating initiatives like internships, career counseling, leadership workshops, or study-abroad opportunities. These programs often focus on building students' soft skills or enhancing their employability.

**Why it's important**: These initiatives help foster a vibrant academic community, encourage professional growth, and provide opportunities for networking and skill-building for both students and staff.

**Performance Insights:**

**Objective**: Analyse performance data to identify trends and areas for improvement for both students and faculty.

* **Faculty Performance**: Analysing faculty performance through evaluations, feedback from students, and peer reviews. Identifying areas where faculty can improve, whether in teaching methods, research, or departmental contributions.
* **Student Performance**: Tracking academic progress over time to identify trends. This could include identifying students who are excelling, those who are struggling, or those showing signs of disengagement.
* **Data Analysis Tools**: Using analytics to spot patterns, such as which courses are underperforming or which teaching methods yield better student outcomes. It can also include benchmarking performance against other institutions.
* **Improvement Action Plans**: Based on insights, institutions can implement targeted interventions, whether offering faculty development programs, adjusting course content, or providing students with additional support such as tutoring or counselling.

**Why it's important**: By integrating these diverse data streams, institutions can enhance academic processes, improving performance and student outcomes through data-driven decisions.

**The Academic Task Management System (ATMS)**

The **Academic Task Management System (ATMS)** is innovative software designed to record, manage, and report on the various academic and administrative tasks conducted by educational institutions. The current system often involves manual processes, which are time-consuming and costly. ATMS aims to streamline and automate these tasks, improving efficiency and reducing operational costs.

**Objective: The primary goal of ATMS is to improve the efficiency and effectiveness of educational institutions while providing a seamless experience for students, faculty, and administrators.**

**By centralizing the management of academic data (student, faculty, course, event, and performance data), ATMS serves as the backbone for managing all academic processes.**

**Key Modules in ATMS**

The ATMS is designed with three main modules to address the needs of different stakeholders in an educational institution:

**Admin Module**

The **Admin Module** is the central hub where all backend management and configuration tasks are handled. Typically used by administrators or system users with high-level permissions.

**Key Features of the Admin Module:**

* **User Management:**
  + Add/Edit/Delete Users: Admins can create accounts for students, teachers, and other staff, as well as edit or deactivate user information.
  + Role Assignment: Admins assign roles (e.g., Student, Teacher, Staff) and define permissions based on the user’s role.
* **Course & Subject Management:**
  + Course Creation: Admins can create and manage courses, including defining course content, prerequisites, and credit hours.
  + Schedule Management: Admins allocate classrooms, assign teachers to courses, and ensure courses fit within the academic timetable.
  + Course Catalogue Maintenance: Ensures the course catalogue remains current and aligned with academic regulations.
* **Student & Teacher Data Management:**
  + Student Enrolment: Admins manage student enrolments and track academic progress.
  + Faculty Management: Admins monitor faculty performance, schedules, and workloads.

**Student Module**

The **Student Module** allows students to interact with the system, manage their academic progress, register for courses, and track performance.

**Key Features of the Student Module:**

* **Profile Management:**
  + Personal Information: Students can view and update their contact details, emergency contacts, and address.
  + Academic Profile: Students can view their academic history, grades, attendance, and GPA.
* **Course Registration:**
  + Browse & Register for Courses: Students can view available courses, check prerequisites, and register for classes.
  + View Timetable: Students can access a personalized schedule with dates, times, and locations for their enrolled courses.
* **Grades & Performance Tracking:**
  + View Grades: Students can track their grades for assignments, exams, and overall course performance.
  + Track Progress: Students can monitor their academic progress, including GPA and course completion status.
* **Attendance Management:**
  + Track Attendance: Students can view attendance records for each class.
* **Assignments & Exams:**
  + Access Assignments/Exams: Students can download, submit, and check deadlines for assignments.
  + View Feedback: Students can view feedback from teachers on their assessments.

**Teacher Module**

The **Teacher Module** is designed to help faculty members manage their courses, interact with students, and monitor student performance.

**Key Features of the Teacher Module:**

* **Profile Management:**
  + View/Edit Profile: Teachers can update their personal information, teaching schedules, and office hours.
  + Subject Management: Teachers can manage the courses they teach, including course content and resources.
* **Course Management:**
  + Course Creation & Modification: Teachers can create and modify course materials, set objectives, and update syllabi.
  + Assignments & Exams: Teachers can create and assign homework, quizzes, projects, and exams.
* **Grading and Evaluation:**
  + Grade book Management: Teachers can enter grades for assignments, exams, and class participation.
  + Track Student Progress: Teachers can monitor each student’s performance across assessments.

**Key Features:**

* **User Authentication & Authorization:** Secure login system with role-based access (admin, faculty, student, etc.).
* **Data Management (CRUD Operations):** Allows creation, retrieval, updating, and deletion of student, faculty, and academic data.
* **Responsive Design:** Adaptive layout using Bootstrap to ensure usability across desktops, tablets, and smartphones.
* **Real-Time Updates & Interactive Features:** Utilization of JavaScript for dynamic content updates and real-time notifications.
* **Data Persistence with MySQL:** Storing academic data, user profiles, and transactional information in an integrated MySQL database.

### **Functional Requirements:**

1. **User Management:**
   * Users must be able to register, log in, and manage their accounts with secure password recovery options.
   * Admin users will have the ability to manage user roles and permissions (e.g., assigning roles like student, faculty and administrator).
2. **Data Operations:**
   * The application will support Create, Read, Update, and Delete (CRUD) operations on academic data entities (students, faculty, courses, assessments, etc.).
   * Data will be validated and stored securely in the MySQL database.
3. **Search and Filter:**
   * Users should be able to search for specific records or filter data based on multiple criteria (e.g., course name, student grade, faculty department).
4. **Reporting:**
   * The system should allow generation of customizable reports, such as student performance reports, faculty attendance, course statistics, and academic trends.
   * Reports should be exportable in common formats (e.g., PDF, CSV).
5. **Responsive Design:**
   * The web application will use Bootstrap to ensure it is responsive and performs well across a wide range of devices (desktops, tablets, and mobile devices).
   * The interface will adapt dynamically to various screen sizes without sacrificing usability.
6. **Error Handling:**
   * The application will provide meaningful error messages for invalid operations or input.
   * It will gracefully handle errors such as incorrect login attempts, missing data, and system failures.

### **Performance Requirements**

1. **Response Time:**
   * The application should respond to user interactions within 2 seconds under typical load conditions.
2. **Scalability:**
   * The system must be capable of handling up to 500 simultaneous users without noticeable degradation in performance. This includes managing multiple concurrent logins, data retrieval, and report generation.
3. **Availability:**
   * The application should maintain an uptime of 99.5% or higher, ensuring minimal downtime for users.
4. **Data Handling:**
   * The system must be able to efficiently handle a large volume of academic data (e.g., 50,000 student records, 500 faculty members, 200 courses), allowing for smooth operations even as data grows over time.

### **Exception Handling:**

1. **Error Reporting:**
   * All application errors (e.g., system failures, input errors) should be logged with detailed information for debugging.
   * Users will be provided with clear, user-friendly error messages in case of an issue (e.g., "Invalid username or password" or "Database connection error").
2. **Recovery:**
   * The system should include mechanisms to recover from common errors, such as:
     + Data input issues (e.g., duplicate records, missing fields).
     + Connection failures (e.g., automatic retry for database connections).
3. **Security:**
   * Any security-related exceptions, such as unauthorized access attempts, should be logged and addressed with appropriate security measures (e.g., lock accounts after multiple failed login attempts, prompt for CAPTCHA after suspicious activities).

### **Design Hints and Guidelines**

1. **Modular Design:**
   * The software should be built following modular design principles to ensure ease of maintenance and scalability. Each module (e.g., user management, reporting, data processing) should be independent, with clearly defined interfaces.
2. **Code Standards:**
   * Adhere to established coding standards for all technologies used (e.g., Java, HTML, CSS, JavaScript). This includes using descriptive variable names, maintaining code consistency, and documenting complex sections of code.
3. **User Interface:**
   * The user interface should be consistent, leveraging Bootstrap for responsiveness and ensuring the application feels cohesive across all pages and functions.
   * Ensure all UI elements (buttons, forms, navigation menus) are easy to understand and interact with.
4. **Security:**
   * Implement secure coding practices, such as:
     + SQL injection prevention using parameterized queries.
     + Cross-Site Scripting (XSS) protection.
     + Secure user authentication with password hashing and salting.
5. **Testing:**
   * Perform unit tests, integration tests, and user acceptance testing (UAT) throughout the development process to ensure the application meets all requirements.
   * Ensure automated testing for regression to minimize the risk of breaking features in future updates.

In the academic world, efficiently managing large volumes of data is both a challenge and a necessity. Educational institutions, from schools to universities, are overwhelmed with a vast array of information, including staff records, student data, course specifics, faculty development programs, and evaluation results. This data is essential for maintaining smooth operations and providing high-quality education. However, managing this information effectively demands a refined approach, particularly as institutions aim to improve performance and optimize administrative processes.

This project centre on **academic task management** within educational institutions. The domain of this project encompasses the recording, management, and reporting of a variety of academic activities and data. It involves:

**Technology Requirements**

**Hardware:**

* + Processor : Core i3 or above
  + Memory : 4GB or above
  + Hard Disk : 500GB or above

**Software:**

* + Operating System : Windows 7 or above
  + Server : Apache Tomcat Server 10
  + Web Technologies : HTML, CSS (Bootstrap), JavaScript
  + IDE : Visual Studio Code 1.90.2
  + Server Scripting : Java
  + Database  **:** MySQL 8.0

### **5. SYSTEM DESIGN/METHODOLOGY**

**5.1 Methodology:**

The Academic Task Management System follows the **Agile Development Methodology**, which emphasizes iterative development, regular feedback, and adaptability to evolving requirements. Each feature was developed and tested incrementally.

**5.2 Software Architecture:**  
The system uses a **three-tier architecture**, comprising:

1. **Presentation Layer (Front-end):** Implements the user interface using technologies like HTML, CSS, JavaScript, and Bootstrap.
2. **Business Logic Layer (Back-end):** Handles application logic, written in Java, integrated with JSP.
3. **Data Layer (Database):** Uses MySQL to store and retrieve data related to tasks, users, deadlines, and statuses.

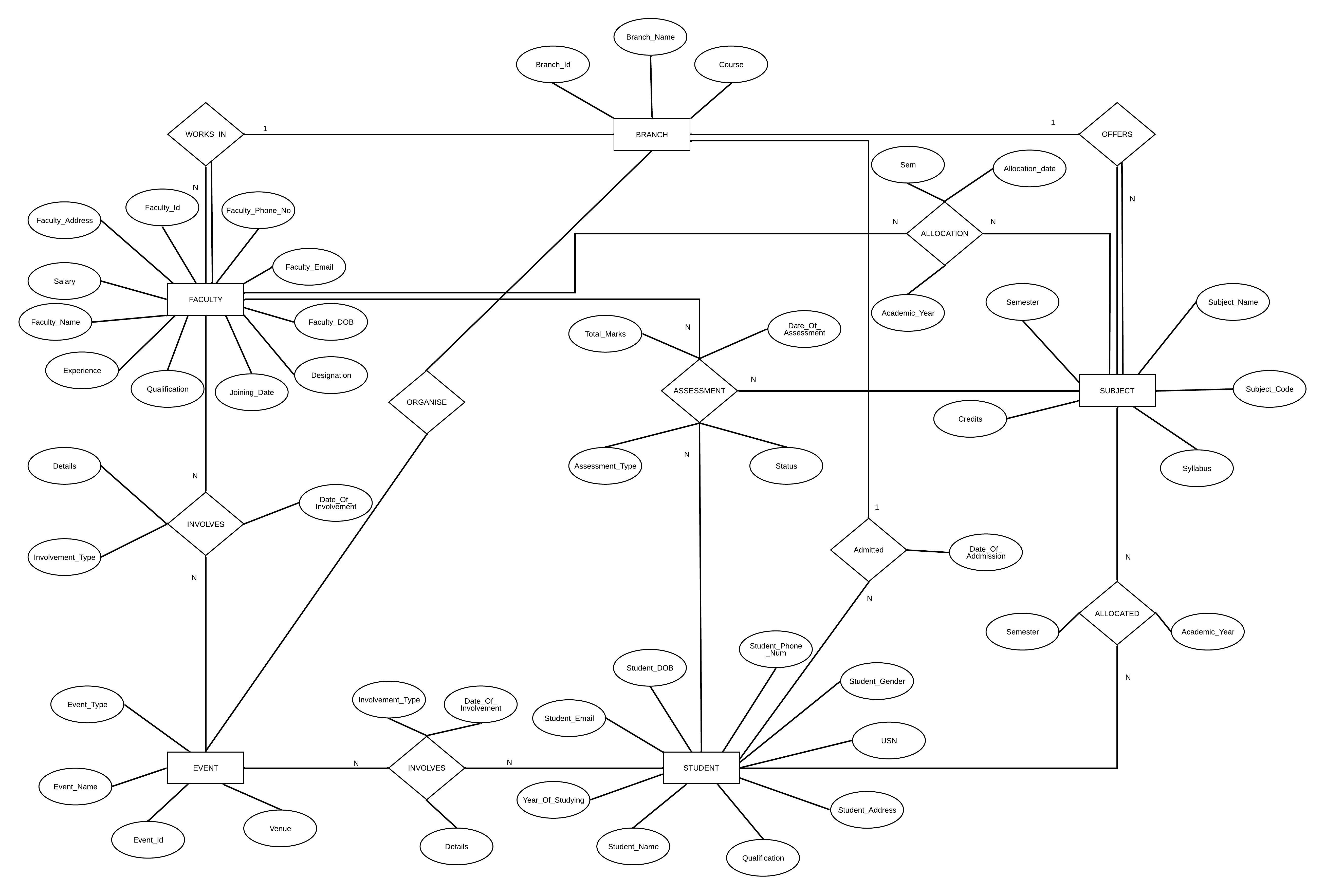
**5.3 Database Design:**

**ER-Diagram**

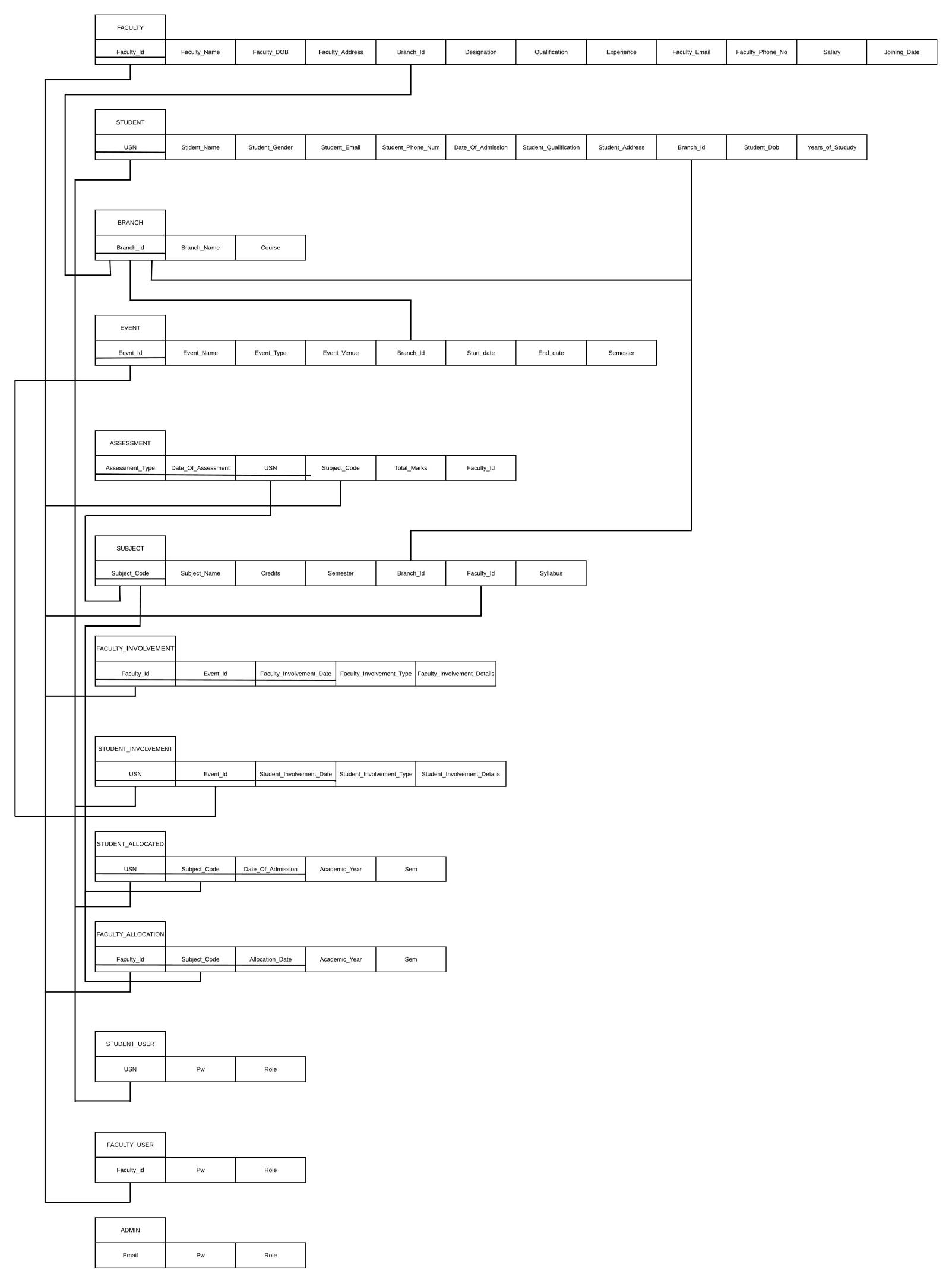
ER-Diagram is one of the data modeling method which is used to provide information of the system and the entities which gives relation between these entities.



**ER-Diagram:**



**Relationship Model:**



**5.4 UI Design:**

The system interface is simple and user-friendly, with the following features:

* A **dashboard** displaying pending tasks, completed tasks, and notifications.
* Separate pages for **task assignment** (faculty) and **task tracking** (students).
* Forms for task creation, submission, and evaluation, designed using Bootstrap for responsiveness.

#### 5.5. **Module Design \ Functional Design:**

### **Module 1: User Authentication and Authorization**

**Actors Involved**: Admin, Faculty, Student  
**Functionalities**:

* Secure login for users based on roles (Admin, Faculty, Student).
* Password recovery via registered email or security questions.
* Session management to track logged-in users and enforce role-based access control.  
  **Inputs**: Username, password, and (for recovery) registered email or security question answers.  
  **Outputs**: Role-specific access granted (Admin, Faculty, Student).

### **Module 2: User and Profile Management**

**Actors Involved**: Admin  
**Functionalities**:

* Create users (students, faculty) and assign roles using unique identifiers (USN for students, facultyId for faculty).
* Update user details (e.g., name, email, role, permissions).
* Delete users who are no longer part of the system.
* Manage branches, events, and subjects.  
  **Inputs**: User details (name, email, role, unique ID), branch names, event details, and subject details.  
  **Outputs**: Updated user list with role-based access and profiles, branch, event, and subject data added to the system.

**Module 3: Event Management**

**Actors Involved**: Admin, Faculty, Student  
**Functionalities**:

* **Admin**: Create events with details like name, date, type (e.g., participant or organizer roles).
* **Faculty and Students**: Apply for events created by the admin as either participants or organizers.
* Manage event participation and involvement data for all users.  
  **Inputs**: Event details (name, date, type, roles) and application requests from users.  
  **Outputs**: Event participation and organization records updated in user profiles.

**Module 4: Course and Subject Management**

**Actors Involved**: Admin, Faculty, Student  
**Functionalities**:

* **Admin**: Create, update, or delete courses and assign subjects to faculty.
* **Faculty**: View assigned subjects and check how many students are allocated to their subjects.
* **Students**: Allocate themselves to subjects created by the admin.  
  **Inputs**: Course details (name, schedule, prerequisites), subject details, and student-subject allocation data.  
  **Outputs**: Updated course and subject lists with faculty and student allocations.

**Module 5: Assessments and Grades**

**Actors Involved**: Faculty, Student  
**Functionalities**:

* **Faculty**: Create and assign assessments to students for their subjects, grade them, and upload marks.
* **Students**: View assigned assessments submit them, and access grades and feedback.  
  **Inputs**: Assessment details (name, due date, questions), grades, and student submissions.  
  **Outputs**: Completed assessments with feedback and grades visible to students.

**Module 6: Academic Data Management**

**Actors Involved**: Faculty, Admin  
**Functionalities**:

* Add or update grades for students.
* Record attendance data.
* Upload and manage course materials.  
  **Inputs**: Academic data (grades, attendance records, and course materials).  
  **Outputs**: Updated student profiles with academic performance and attendance details.

**For Faculty**:

* In the faculty profile:
  + Generate a PDF report showing:
    - List of subjects allocated to the faculty.
    - Details of students enrolled in each subject.
    - List of events applied for, along with the roles (organizer/participant).

#### **For Students**:

* In the student profile:
  + Generate a PDF report showing:
    - List of subjects the student is enrolled in.
    - List of events applied for, along with the roles (organizer/participant).

**Inputs**:

* User role (Faculty or Student).
* Data retrieved from subjects and events tables.

**Outputs**:

* **Faculty Profile Reports**: Downloadable PDF with subjects and event participation details.
* **Student Profile Reports**: Downloadable PDF with enrolled subjects and applied events.

### **Module 8: Course Enrolment**

**Actors Involved**: Admin, Student  
**Functionalities**:

* **Admin**: Manage course prerequisites, capacities, and schedules.
* **Students**: Enrol in courses for the upcoming semester.
* Perform conflict checks for duplicate enrolment or schedule overlaps.  
  **Inputs**: Course selection, prerequisites, and schedules.  
  **Outputs**: Enrolled courses updated in the student profiles.

### **5.6 Detailed Descriptions**

#### **Use Case 1: User Authentication**

**Description:**

* Ensures secure access to the system.
* After login, users are directed to dashboards based on their roles.
* Implements encryption for passwords stored in the database.

**Workflow:**

1. User enters username and password on the login page.
2. System verifies credentials using the database.
3. If valid, access is granted; otherwise, an error message is shown.
4. For password recovery, users can request a reset link sent to their email.

#### **Use Case 2: Manage Users**

**Description:**

* Admin can perform CRUD operations (Create, Read, Update, Delete) on user records.
* Role-based permissions ensure data privacy and secure access.

**Workflow:**

1. Admin logs in and navigates to the user management module.
2. Admin adds new users by filling out a form with user details.
3. System validates input and updates the database.

#### **Use Case 3: Manage Academic Data**

**Description:**

* Faculty and Admin can input, modify, and maintain academic records.
* Tracks grades, attendance, and course-related information.

**Workflow:**

1. Faculty accesses the module for a particular course or student.
2. Data such as grades or attendance is entered.
3. The system updates records and calculates overall student performance.

#### **Use Case 4: Manage Courses**

**Description:**

* Faculty creates or modifies courses, while Admin monitors scheduling and conflicts.
* Enables a seamless course management process.

**Workflow:**

1. Faculty or Admin adds course details such as name, prerequisites, and schedule.
2. System validates input and ensures no conflicts in schedules.
3. Updates are reflected in the course catalogue.

#### **Use Case 5: Manage Assignments and Grades**

**Description:**

* Faculty assigns and grades tasks, while students submit work online.

**Workflow:**

1. Faculty creates an assignment and sets a deadline.
2. Students upload their work before the due date.
3. Faculty evaluates submissions and records grades.

#### **Use Case 6: Generate Reports**

**Description:**

* Generates detailed insights into academic performance and participation.

**Workflow:**

1. Admin or Faculty selects the report type and filters criteria.
2. System processes data and generates a preview.
3. Finalized reports are exported in PDF or CSV formats.

#### **Use Case 7: Course Enrolment**

**Description:**

* Streamlines student course registration while preventing conflicts.

**Workflow:**

1. Students select courses from the available list.
2. The system checks prerequisites and capacity.
3. Approved courses are added to the student’s profile.

**6. System Development / Implementation**

**Creation of Database:**

Database is used to store the information related to data in a systematic way. This is used in storing and manipulating the data in an easy way. To develop this web application we need to use RDBM which is used to store the data in different tables where the table consists of a primary key and a foreign key in the database tables. By using this relational database management system we can implement the database with the tables, columns and by the indexes, which allows the coordination between the rows and to the variety of database tables.

Following are the terms which are used for designing the database of the application

**Database –** It is the group of associated data in the form of the tables.

**Tables–** Tables which are called as the relations that are used to store the proper data in the forms of rows and the columns.

**Column –** Columns are called as data element which is used to identify the field names in the tables.

**Row –** This consists of relatable data for the application. The data which is stored in the table is called as tuple. Row is called as a collection of a data which points to different columns of the same table.

**Primary key –** The table must consists of only one primary key which is always be unique in the table. Tables which consist of records must be always unique.

**Foreign key - This** key is used as a reference to the other table i.e., when two tables are connected the foreign key plays an important role in establishing the connection between these tables to provide link for the database.

**Code to connect DB:**

try {

String dbURL = "jdbc:mysql://localhost:3306/atms";

String dbUser = "root";

String dbPassword = "ROOT";

Class.forName("com.mysql.cj.jdbc.Driver");

conn = DriverManager.getConnection(dbURL, dbUser, dbPassword);

boolean validUser = false;

if (role.equals("Admin")) {

String query = "SELECT \* FROM ADMIN WHERE Email = ? AND Password = ?";

stmt = conn.prepareStatement(query);

stmt.setString(1, email);

stmt.setString(2, password);

rs = stmt.executeQuery();

if (rs.next()) {

validUser = true;

}

} else if (role.equals("Student")) {

String query = "SELECT \* FROM STUDENT\_USER WHERE USN = ? AND Pw = ?";

stmt = conn.prepareStatement(query);

stmt.setString(1, usn);

stmt.setString(2, password);

rs = stmt.executeQuery();

if (rs.next()) {

validUser = true;

}

} else if (role.equals("Faculty")) {

String query = "SELECT \* FROM FACULTY\_USER WHERE Faculty\_Id = ? AND Pw = ?";

stmt = conn.prepareStatement(query);

stmt.setString(1, facultyId);

stmt.setString(2, password);

rs = stmt.executeQuery();

if (rs.next()) {

validUser = true;

}

}

if (validUser) {

if (role.equals("Admin")) {

response.sendRedirect("admin-home.jsp");

} else if (role.equals("Student")) {

response.sendRedirect("student-home.jsp?usn=" + usn);

} else if (role.equals("Faculty")) {

response.sendRedirect("faculty-home.jsp?facultyId=" + facultyId);

}

} else {

out.println("<div class='alert alert-danger'>Invalid credentials or role</div>");

}

} catch (Exception e) {

e.printStackTrace();

out.println("<div class='alert alert-danger'>Error: " + e.getMessage() + "</div>");

} finally {

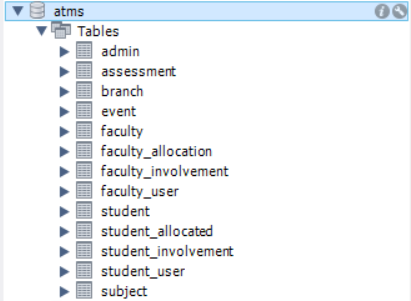
if (rs != null) try { rs.close(); } catch (SQLException ignore) {}

if (stmt != null) try { stmt.close(); } catch (SQLException ignore) {}

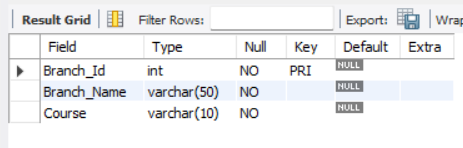
if (conn != null) try { conn.close(); } catch (SQLException ignore) {}

}

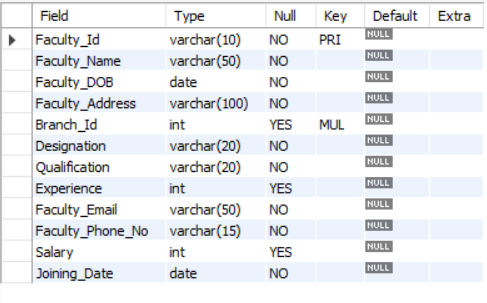
**DATABASE Schema**



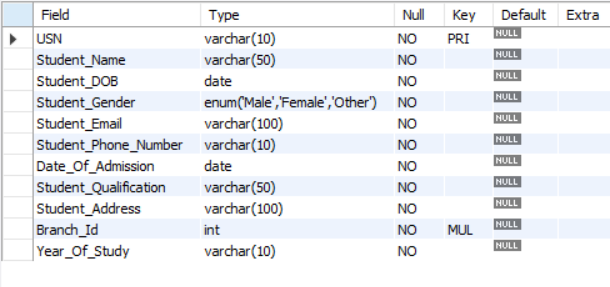
Branch Table



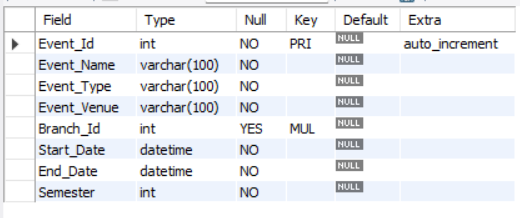
Faculty table



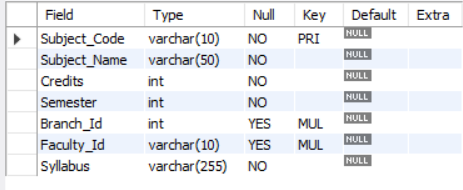
Student table



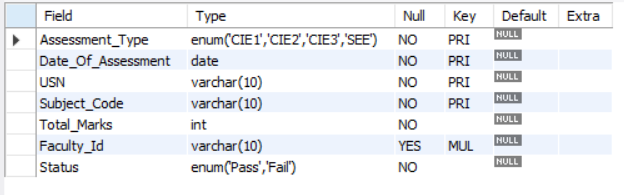
Event table



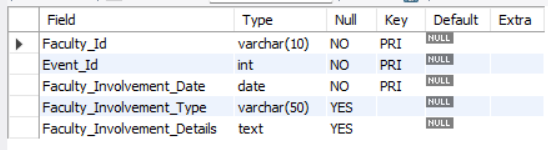
Subject table



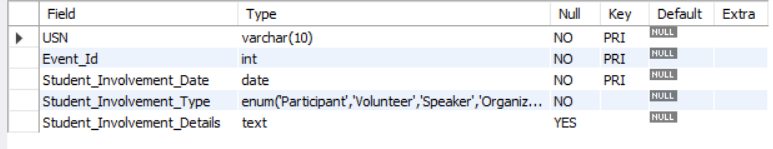
Assessment table



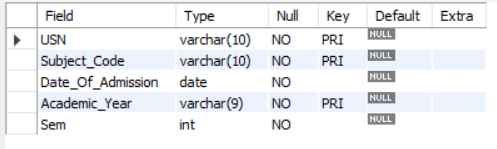
FACULTY\_INVOLVEMENT table



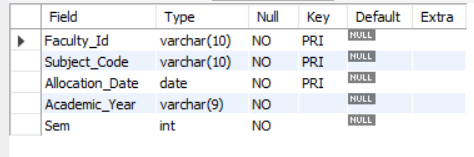
STUDENT\_INVOLVEMENT



STUDENT\_ALLOCATED

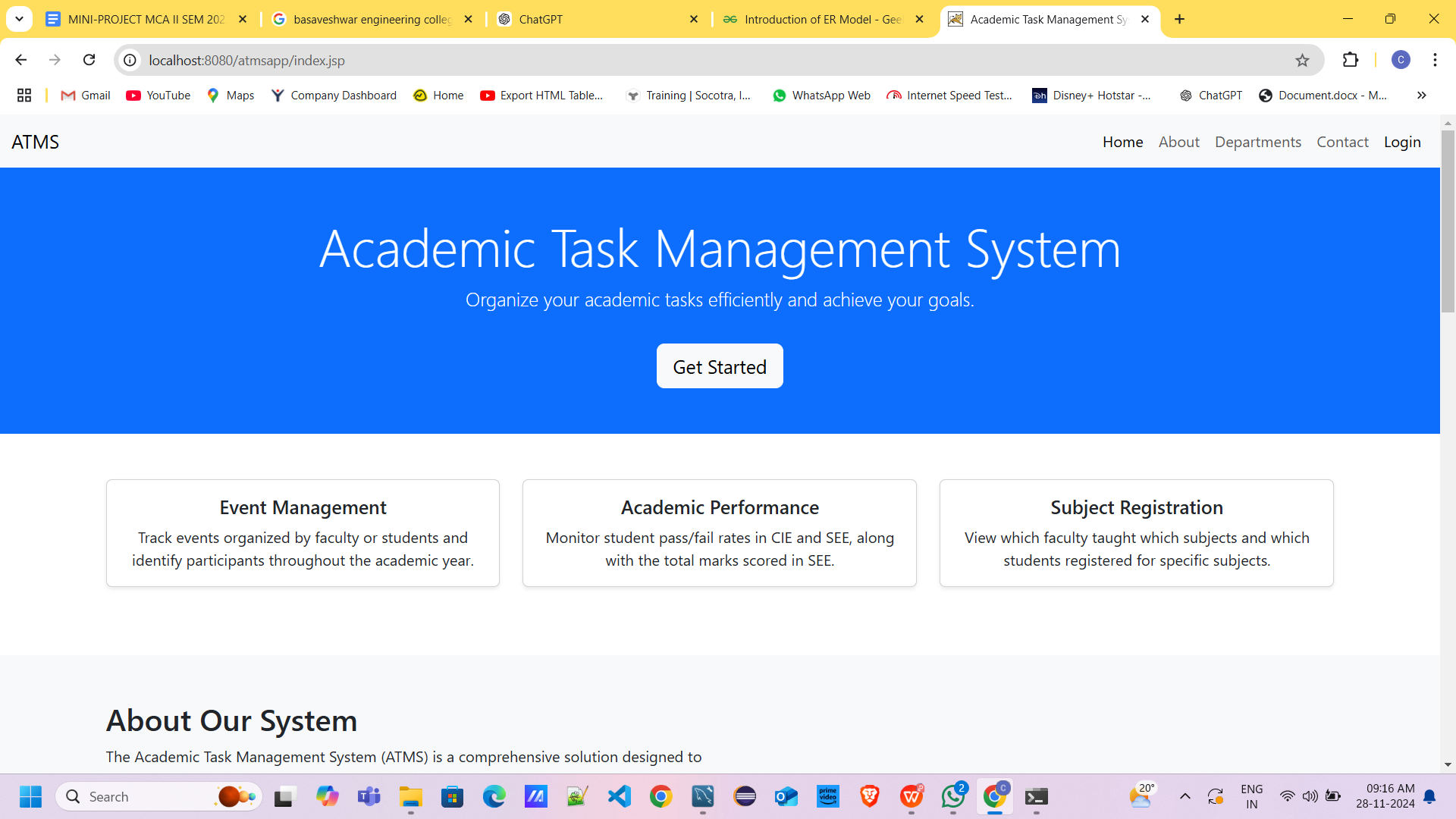


FACULTY\_ALLOCATION

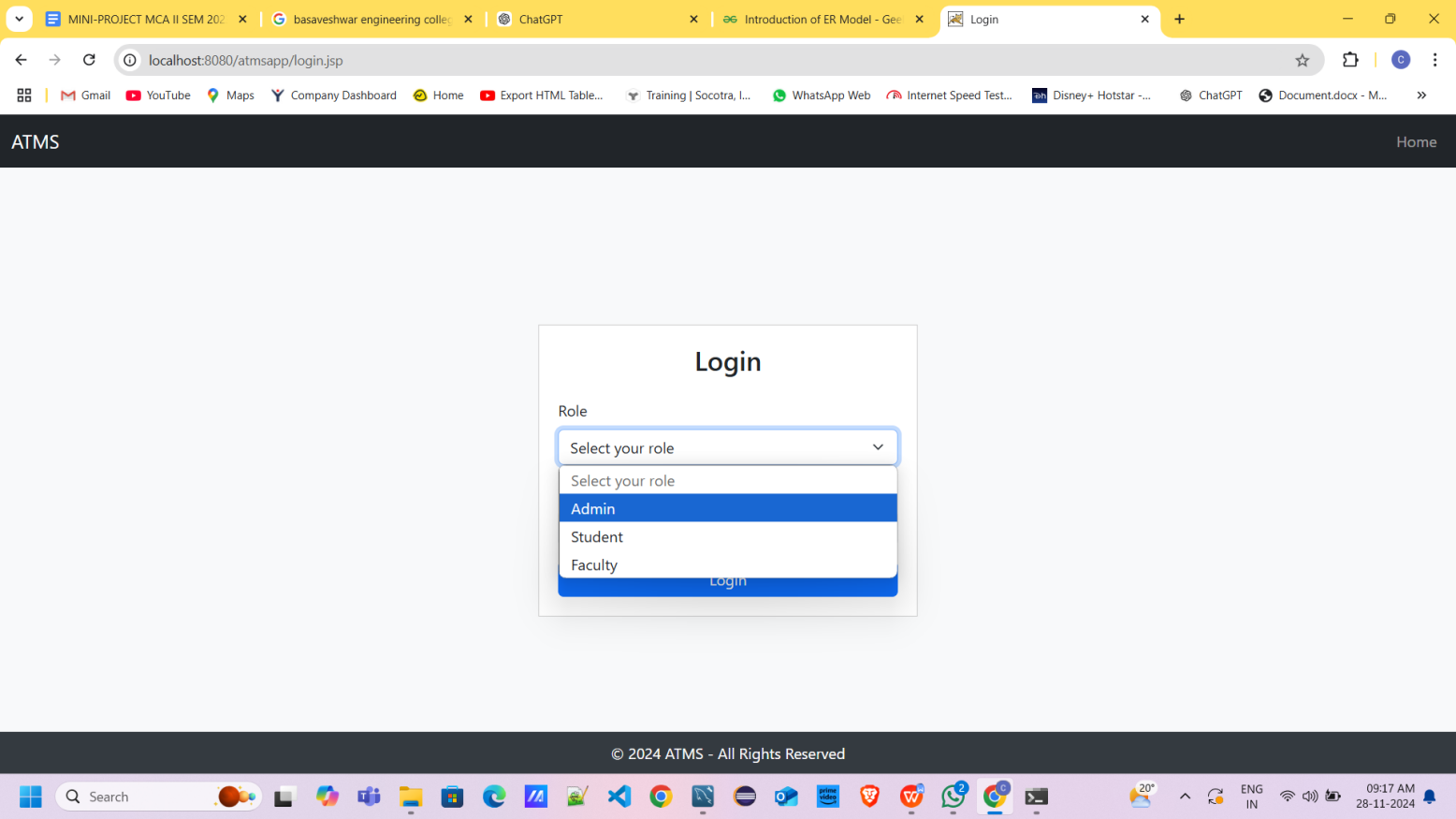


**UI design:**

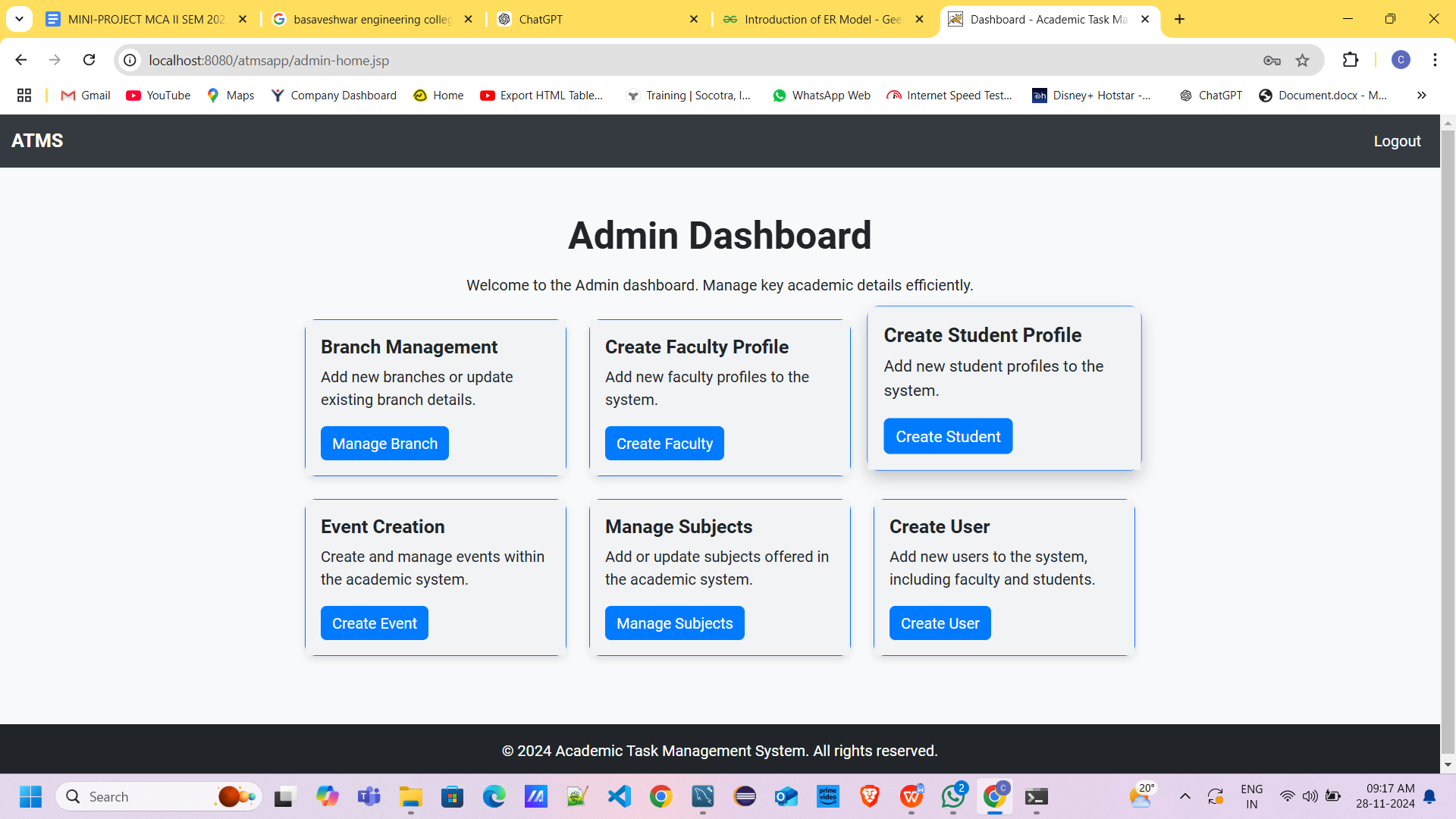
Index page



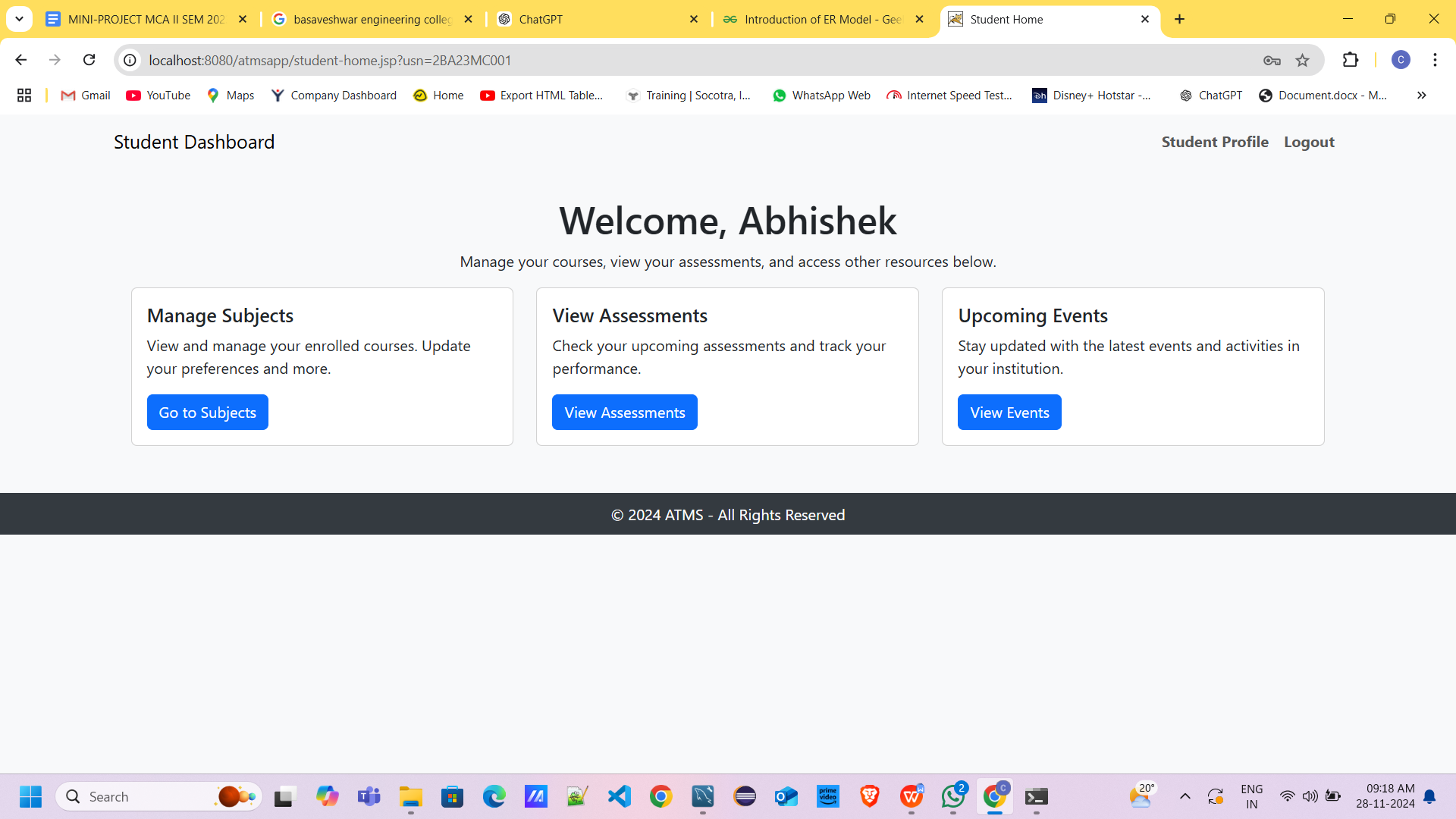
Login page



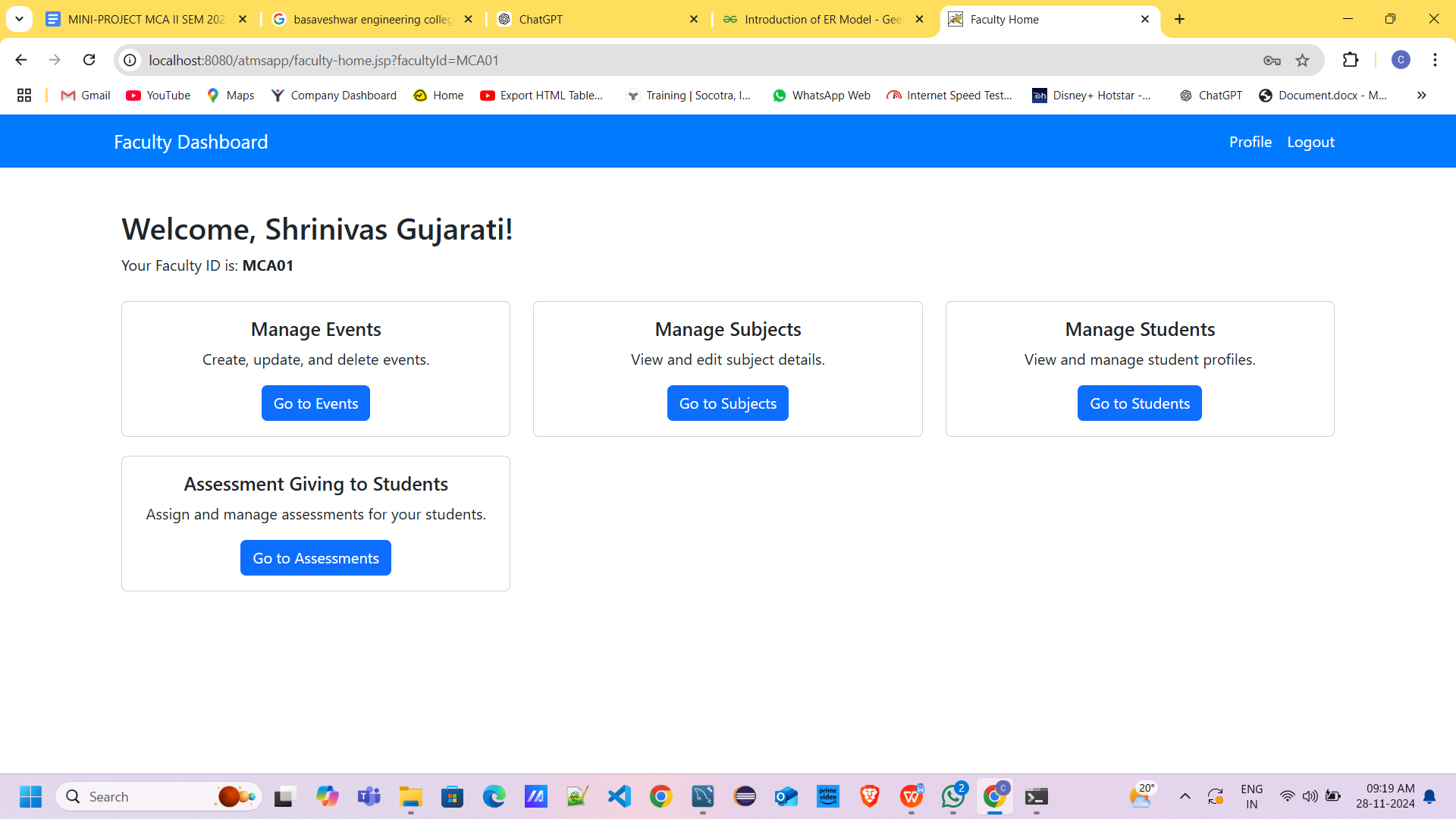
Admin home page



Student home page



Faculty home page



### **Software Requirements**

#### **1. Server**

* **Apache Tomcat Server 10**
  + **Purpose**:
    - Acts as the web server for hosting and running Java-based web applications.
    - Serves JSP (Java Server Pages) and Servlet requests, ensuring dynamic content delivery.
    - Provides a robust and scalable environment for deploying applications.

#### **2. Web Technologies**

**HTML (HyperText Markup Language)**:

* + Used for structuring and presenting content on the web.
  + Provides the basic layout and organization of the web pages, such as forms, tables, and headings.

**CSS (Cascading Style Sheets)**:

* + Enhanced with **Bootstrap** for designing responsive, mobile-friendly, and visually appealing interfaces.
  + Ensures uniform styling across all web pages, such as fonts, colours, and layouts.

**JavaScript**:

* + Adds dynamic and interactive features to web pages, such as form validation, interactive menus, and real-time feedback.
  + Enhances user experience by reducing the need for server interactions for minor tasks.

#### **3. Server-Side Scripting**

* **Java (JSP - Java Server Pages)**:
  + Handles server-side logic, such as processing form submissions, retrieving data from the database, and generating dynamic HTML pages.
  + Ensures secure communication between the client and the server.
  + Integrates seamlessly with MySQL using JDBC (Java Database Connectivity).

#### **4. Database**

* **MySQL 8.0**:
  + A relational database management system (RDBMS) used for storing, retrieving, and managing all system data.
  + Key uses in the project:
    - Stores user credentials, tasks, academic records, courses, assignments, and grades.
    - Ensures data integrity with features like primary and foreign key constraints.
    - Facilitates complex queries and report generation for academic analysis.
  + **Advantages**:
    - Supports scalability and high performance for handling large datasets.
    - Open-source and well-documented, making it a reliable choice for development.

#### **5. Integrated Development Environment (IDE)**

* **Visual Studio Code 1.90.2**:
  + A lightweight and powerful IDE used for developing and managing the project code.
  + **Key Features**:
    - Syntax highlighting for HTML, CSS, JavaScript, and Java.
    - Integration with MySQL plugins and tools for database management.
    - Debugging tools for identifying and resolving errors during development.
    - Version control integration (e.g., Git) for collaborative development.

#### **6. Operating System**

* **Windows/Linux/MacOS**:
  + The project is platform-independent and can be developed or deployed on any of these operating systems.
  + **Key Considerations**:
    - **Windows**: Popular among developers for ease of use and compatibility with Visual Studio Code and MySQL.

**7. System Testing and Results**

### **Module 1: Authentication**

#### **Test Case 1: Admin Login**

* **Input**:
  + Email: chetan@gmail.com
  + Password: Admin@123
* **Expected Output**: Successful login and access to admin functionalities.

#### **Test Case 2: Faculty Login**

* **Input**:
  + Faculty ID: MCA02
  + Password: Faculty@123
* **Expected Output**: Redirect to the faculty dashboard.

#### **Test Case 3: Student Login**

* **Input**:
  + USN: 2BA23MC001
  + Password: Abhi@123
* **Expected Output**: Redirect to the student dashboard.

#### **Test Case 4: Invalid Login**

* **Input**:
  + User Type: Faculty or Student or Admin
  + Faculty ID: InvalidID
  + Password: InvalidPassword
* **Expected Output**: Error message: "Invalid username or password." & USN starts from 2BA23MC

### **Module 2: Branch Management**

#### **Test Case 1: Add a New Branch**

* **Input**:
  + Branch ID: 101
  + Branch Name: Computer Science
  + Course: UG, PG
* **Expected Output**: Branch added successfully.

#### **Test Case 2: View All Branches**

#### **Query**:

SELECT \* FROM BRANCH;

* **Expected Output**: Display all branches in the database.

#### **Test Case 3: Delete a Branch**

* **Input**: Branch ID: 102
* **Expected Output**: Branch deleted successfully.

### **Module 3: Faculty Management**

#### **Test Case 1: Add Faculty**

* **Input**:
  + Faculty ID: MCA01
  + Faculty Name: Dr. Ram Kumar
  + Designation: Professor
  + Qualification: PhD
  + Branch ID: 101
  + Experience: 10
  + Salary: 75000
  + Joining Date: 2024-01-01
* **Expected Output**: Faculty added successfully.

#### **Test Case 2: View Faculty Details**

**Query**:

SELECT \* FROM FACULTY WHERE Faculty\_Id = 'MCA02';

* **Expected Output**: Details of faculty with ID MCA02 are displayed.

#### **Test Case 3: Update Faculty Salary**

**Query**:

UPDATE FACULTY SET Salary = 80000 WHERE Faculty\_Id = 'MCA02';

* **Expected Output**: Success message: "Salary updated successfully."

### **Module 4: Student Management**

#### **Test Case 1: Add Student**

* **Input**:
  + USN: 1RV23CS001
  + Name: Amit Shah
  + Branch ID: 101
  + Year of Study: 2nd
  + Date of Admission: 2022-06-15
* **Expected Output**: Student added successfully.

#### **Test Case 2: View All Students in a Branch**

SELECT \* FROM STUDENT WHERE Branch\_Id = 101;

* **Expected Output**: List of students in branch 101.

#### **Test Case 3: View Student Details**

**Query**:

SELECT \* FROM STUDENT WHERE USN = '1RV23CS001':

* **Expected Output**: Details of the student with USN 1RV23CS001 are displayed.

### **Module 5: Event Management**

#### **Test Case 1: Add Event**

* **Input**:
  + Event Name: Hackathon
  + Type: Coding
  + Venue: Main Auditorium
  + Branch ID: 101
  + Start Date: 2024-12-01
  + End Date: 2024-12-02
* **Expected Output**: Event added successfully.

#### **Test Case 2: Faculty Involvement in Event**

* **Input**:
  + Faculty ID: MCA02
  + Event ID: 1
  + Date: 2024-12-01
  + Type: Organizer
* **Expected Output**: Faculty involvement recorded successfully.

#### **Test Case 3: Student Involvement in Event**

* **Input**:
  + USN: 1RV23CS001
  + Event ID: 1
  + Date: 2024-12-01
  + Type: Participant
* **Expected Output**: Student involvement recorded successfully.

### **Module 6: Subject Management**

#### **Test Case 1: Allocate Subject to Faculty**

* **Input**:
  + Faculty ID: MCA02
  + Subject Code: CS101
  + Allocation Date: 2024-12-01
  + Academic Year: 2024-25
* **Expected Output**: Subject allocated to faculty successfully.

#### **Test Case 2: Allocate Subject to Student**

* **Input**:
  + USN: 1RV23CS001
  + Subject Code: CS101
  + Academic Year: 2024-25
  + Semester: 3
* **Expected Output**: Subject allocated to student successfully.

#### **Test Case 3: View Subjects of a Branch**

SELECT \* FROM SUBJECT WHERE Branch\_Id = 101;

* **Expected Output**: List of subjects for branch 101

### **Module 7: Assessment Management**

#### **Test Case 1: Record Assessment**

* **Input**:
  + Assessment Type: CIE1
  + USN: 1RV23CS001
  + Subject Code: CS101
  + Marks: 85
  + Status: Pass
* **Expected Output**: Assessment recorded successfully.

#### **Test Case 2: View Student Assessment Results**

**Query**:

SELECT \* FROM ASSESSMENT WHERE USN = '1RV23CS001';

* **Expected Output**: All assessments for student 1RV23CS001 are displayed.

#### **Test Case 3: View Subject-Wise Assessments**

**Query**:

SELECT \* FROM ASSESSMENT WHERE Subject\_Code = 'CS101';

* **Expected Output**: All assessments for the subject CS101 are displayed.

**8. Future Enhancement and Conclusion**

**Future Enhancement**

The AI models further enhance the system by enabling predictive analytics, personalized learning pathways for students, and automated scheduling and resource allocation.

**Conclusion**

The Academic Task Management System (ATMS) is a powerful tool that helps schools and universities manage their data more effectively. It combines various academic and administrative tasks into one easy-to-use platform, making it simpler to track Faculty records, student progress, and subject management.

1. **Document References and Web page Links**

#### **Official Documentation:**

**Apache Tomcat Server**: [https://tomcat.apache.org/tomcat-10.0-doc/index.html](https://tomcat.apache.org/tomcat-10.0-doc/index.html" \t "_new)

**JSP and Servlets**: [https://docs.oracle.com/javaee/7/tutorial/servlets.htm](https://docs.oracle.com/javaee/7/tutorial/servlets.htm" \t "_new)

**MySQL 8.0**: [https://dev.mysql.com/doc/refman/8.0/en/](https://dev.mysql.com/doc/refman/8.0/en/" \t "_new)

#### **Web Design Resources:**

**Bootstrap Official Documentation**: <https://getbootstrap.com/docs/5.0/getting-started/introduction/>

**W3Schools HTML/CSS/JavaScript Tutorials**: [https://www.w3schools.com/](https://www.w3schools.com/" \t "_new)

#### **Java Programming Resources:**

**Java Documentation (Oracle)**: [https://docs.oracle.com/en/java/javase/17/](https://docs.oracle.com/en/java/javase/17/" \t "_new)

#### **Task Management System Examples:**

**Task Management Best Practices**: <https://www.projectmanager.com/blog/task-management-guide>

**Creating a Task Management System with JSP**: <https://www.geeksforgeeks.org/jsp-task-management-application/>

#### **Project Resources:**

**Full-stack Java Project Tutorials**[: https://www.javatpoint.com/java-web-application-tutorial](:%20https:/www.javatpoint.com/java-web-application-tutorial)

**Integrating MySQL with Java (JDBC)**: <https://www.geeksforgeeks.org/jdbc-java-database-connectivity/>

#### **Sample Projects and Templates:**

**Open Source Task Management System Projects**: [https://github.com/topics/task-management](https://github.com/topics/task-management" \t "_new)

**Academic Management Project Ideas**: <https://nevonprojects.com/academic-management-projects/>

1. **Index**
2. Academic Task Management System……………………10
3. Course Management……………………………………..9
4. Database Design…………………………………………16
5. Functional Design………………………………………..19
6. Implementation…………………………………………..24
7. References………………………………………………..38
8. Testing……………………………………………………33