**ENTERPRISE RESOURCE PLANNING**

Project submitted to the

**SRM University – AP, Andhra Pradesh**

for the partial fulfillment of the requirements to award the degree of

**Bachelor of Technology**

In

**Computer Science and Engineering**

School of Engineering and Sciences

Submitted by

                                Sk Akhib Usman     --------------     AP21110011354

                                V Sumanth              ---------------    AP21110011358

                                S Prasanth               ---------------    AP21110011404

                                Meka Neeharika     ---------------    AP21110011417



Under the Guidance of

**Dr Sanjay Kumar**

**SRM University–AP**

**Neerukonda, Mangalagiri,**

**Guntur, Andhra Pradesh**

**April, 2024**

# Certificate

Date: 01-April -24

This is to certify that the work present in this Project entitled “Enterprise Resource Planning” has been carried out by **Usman, Sumanth, Neeharika, Prasanth** under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology in **School of Engineering and Sciences**.

**Supervisor**

(Signature)

Prof. Dr. Sanjay Kumar, Designation,

Affiliation.

# Acknowledgements

We extend our deepest gratitude to each member of our project team: Usman, Sumanth, Neeharika, and Prasanth. Their unwavering dedication and collaborative spirit have been fundamental to the success of our project. Each team member brought unique strengths and insights, contributing significantly to our collective achievements and ensuring a well-rounded approach to our challenges.

Special thanks are also due to our esteemed guide and mentor, Dr. Sanjay Kumar. His invaluable guidance and expert mentorship were instrumental throughout the duration of our project. Dr. Sanjay Kumar provided not only technical expertise but also continuous support and encouragement, fostering an environment where innovative ideas and solutions could flourish. His commitment to excellence and dedication to nurturing our potential have left a lasting impact on us all.

Furthermore, we would like to acknowledge the supportive environment provided by our institution, which has equipped us with the necessary tools and resources to execute this project effectively. The learning opportunities presented during this journey have been immensely beneficial, and we are grateful for the academic and administrative support that has accompanied our efforts.

This collaborative endeavor has indeed been a rewarding journey, and we wish to acknowledge the collective commitment and hard work of our entire team in achieving the desired outcomes. Together, we have not only reached our goals but have also grown as professionals and individuals. We are proud of what we have accomplished together and are excited for future opportunities to apply the skills and knowledge we have gained.

## Table of Contents

## Certificate……………………………………………………………….1

**Acknowledgement………………………………………………………2**

**Tables of Contents………………………………………………………3**

**List of Figures…………………………………………………………...5**

**List of tables……………………………………………………………..6**

**Abstract………………………………………………………………….7**

**1. Introduction…………………………………………………………..9**

**2. Description of project……………………………………………….10**

**3. SRS Document……………………………………………………….11**

1. **Introduction……………………………………………………...11**
   1. Purpose
   2. Scope
   3. Definitions, Acronyms, and Abbreviations
   4. References
   5. Overview
2. **The Overall Description………………………………………….12**
   1. Product Perspective
   2. Product Functions
   3. User Characteristics
   4. Constraints
   5. Assumptions and Dependencies
3. **External interface Requirements………………………………..16**
   1. User Interfaces
   2. Hardware Interfaces
   3. Software Interfaces
   4. Communications Interfaces
4. **System Features……………………………………………………22**
5. **Other Non-Functional Requirements…………………………….23**
   1. **Performance Requirements**
      1. Capacity
      2. Dynamic Requirements
      3. Quality
   2. **Software System Attributes 12**
      1. Reliability 12
      2. Availability 12
      3. Security 13
      4. Maintainability
   3. **Business Rules**

**4. Functional and nonFunctional requirements…………………………………..47**

**5. Technology used………………………………………………………………4**

**6. Description of Front-end and Back-end………………………………………**

**7. Execution Screenshots…………………………………………………………**

**8. Data flow diagrams……………………………………………………..**

**9. Data Dictionary…………………………………………**

**10. Use case Diagram…………………………………………**

**11. ER Diagram…………………………………………………………**

**12. Class Diagram…………………………………………………………55**

**13. Activity Diagram……………………………………………….57**

**14. State chart Diagram………………………………………………………..59**

**15. Collaboration Diagram…………………………………………..61**

**16. Testing(Path coverage , unit testing)......................................................63**

**17. Conclusion and Limitations……………………………………………65**

**18. References(Github links)...........................................................................67**

**List of Figures**

1. Data flow diagrams………………………………………………….46
2. Use case Diagram…………………………………………………..51
3. ER Diagram…………………………………………………………52
4. Class Diagram……………………………………………………….55
5. Activity Diagram…………………………………………………….58
6. State Chart Diagram…………………………………………………60
7. Collaboration Diagram……………………………………………….62

**List of Tables**  
  
1. Data Dictionary ………………………………………………………………………..50

**Abstract**

In the dynamic business landscape, companies in diverse sectors face unique challenges and pursue distinct goals. To efficiently manage operations, organizations divide tasks into departments, requiring seamless collaboration and timely task completion. Information flow between departments is critical for effective workflow. Enterprise Resource Planning (ERP) facilitates information integration, enabling transparent and instant reporting on company operations. This improves decision-making, reduces process time and costs. ERP, organized into modules for specific business processes, connects to the organization's systems, providing insights into each department. As a single source of truth, ERP ensures accurate and shared data, lowering risks and enhancing productivity. While modern ERP systems offer extensive functions, they require integration with other applications and data sources. This integration allows companies to have a cohesive view of information, leading to improved business processes, enhanced customer experiences, and better collaboration. The benefits of implementing ERP are, it keeps the data safe and regulated with built in protections, it automates all business operations cross departmentally and also provides faster response time to customer queries. This project endeavors to develop a comprehensive Enterprise Resource Planning (ERP) system tailored for effective business management. Leveraging a full-stack technology approach, the software aims to integrate essential business modules, offering a unified platform to streamline and optimize various aspects of organizational operations.

**Keywords**: Enterprise Resource Planning, Full Stack Technology, Business Modules, Enhanced Productivity, Unified Platform, Information Integration, Process Time Reduction, Information Flow, Transparent & Instant Reporting

**Technologies used:**

**Front-End**: HTML, CSS, JS, ReactJs

**Back-End**: NodeJs, PostgreSQL

**1. Introduction**

In the modern educational landscape, the integration of technology is indispensable for enhancing the efficiency and effectiveness of academic administration. Our project, an Enterprise Resource Planning (ERP) system designed specifically for educational institutions, addresses this need by providing a robust, user-friendly platform through which both students and teachers can seamlessly manage academic records and resources. This system not only simplifies administrative tasks but also fosters a more engaging and transparent educational environment.

The primary goal of our ERP system is to streamline the management of essential academic processes including attendance tracking, grade management, and access to educational content. For students, the system offers real-time access to their academic data, such as attendance records and grades, which is crucial for fostering academic responsibility and enhancing student engagement with their learning process. For teachers, the system provides an intuitive interface for managing student records, editing attendance, and updating grades, thus reducing administrative burdens and allowing more time for pedagogical activities.

Our project stands out by integrating all these features into a single, cohesive interface that is accessible to all stakeholders in the educational process—students, teachers, administrative staff, and management. By ensuring data accuracy and real-time updates, our ERP system supports informed decision-making and facilitates a more responsive educational environment.

Through the development and deployment of this ERP system, we aim to demonstrate how technology can be leveraged to improve the organizational efficiency of educational institutions, thereby enhancing both teaching and learning experiences. This project not only showcases our technical capabilities but also our commitment to improving the educational sector through innovative solutions.

**2. Description of project**

ERP, also known as the Enterprise resource planning, is a comprehensive software solution aimed at improving and automating the administrative and academic functions within educational institutions. It seeks to enhance operational efficiency, improve communication among stakeholders, and provide insights for informed decision-making. By automating essential tasks such as student admissions, fee management, timetable scheduling, and attendance tracking, ERP significantly reduces the administrative workload, enabling staff to focus more on quality education.

The system includes a wide range of modules that support different facets of school management, from student information management, which covers enrollment and record-keeping, to academic management with tools for curriculum planning and grading. Financial and staff management modules help handle fee processing and teacher evaluations efficiently. These components are integrated to offer a seamless management experience.

ERP also integrates with external databases and applications, including Student Information Systems (SIS) for real-time data updates, financial systems for accurate accounting, and APIs such as payment gateways for online fee payments and SMS gateways for communications. Additional integration with third-party platforms extends capabilities into online learning environments.

Functions of ERP include registration, user login, profile management, doubt resolution, fee payments, and examination management, with tailored interfaces for students, faculty, and administrators to meet their specific needs.

In summary, ERP is designed to streamline educational management processes, ensuring that administrative and faculty efforts are more focused on student development. With its robust features and integrations, ERP is an essential tool for modern educational institutions looking to enhance their management practice.

**3. SRS Document**

# Introduction

## Purpose

* The purpose of the ERP portal is to streamline and enhance the administrative processes within the school environment.
* It serves as a comprehensive platform that integrates with the Student Information System (SIS) and Financial Management System databases.
* The primary objectives are:
  1. Efficient retrieval of student information, including demographic data, enrolment status, academic records, and attendance records.
  2. Seamless interaction with the financial management system for managing student fees, budget allocations, and expenditure tracking.
  3. Facilitation of key functions such as registration, user login, user profiles, library access, attendance tracking, doubt solving, fee payment, examination marks, finance department management, student analysis, and feedback.

## Scope

* The ERP portal is designed to mainly cater to the needs of students, few needs of faculty, and administration, providing them with a user-friendly interface to access and manage relevant information.
* The scope encompasses:
  1. Student management: Demographic details, enrolment status, academic records, and attendance tracking.
  2. Financial Management: Student Fees.
  3. Functionalities: Registration, user authentication, user profiles, library access, attendance tracking, doubt-solving, fee payment, examination marks, finance department tools, student analysis, and feedback.

## Definitions, Acronyms, and Abbreviations

* ERP: Enterprise Resource Planning
* SIS: Student Information System

## References

What is an ERP & overview of it's functionality

Introduction to ERP

Essentials of an ERP

How to implement ERP

## Overview

* The ERP portal is a comprehensive solution designed to enhance efficiency and transparency in school administration. By integrating with the SIS and Financial Management System databases, it provides a unified platform for managing student information, financial data, and various administrative functions. The user-friendly interface ensures that students, faculty, and administration can easily navigate and utilize the features tailored to their specific needs.

# The Overall Description

## Product Perspective

* + 1. **Overview**
* The School Management System (SMS), also known as "EduSys," is a comprehensive software solution designed to streamline and automate various administrative and academic processes within educational institutions.
  + - * ERP Sys aims to enhance operational efficiency, improve communication between stakeholders, and provide valuable insights for informed decision-making.
      * By automating processes such as student admissions, fee management, timetable scheduling, and attendance tracking, EduSys reduces the administrative burden on school staff, allowing them to focus more on delivering quality education.

## Scope

* + - * ERP system encompasses a wide range of modules and functionalities to support various aspects of school management, including:
      * Student information management: Enrolment, admissions, student records, and demographic data management.
      * Academic management: Curriculum planning, timetable generation, lesson planning, and grading.
      * Financial management: Fee collection, billing, budgeting, and financial reporting.
      * Staff management: Teacher profiles, attendance tracking, professional development tracking, and performance evaluation.

## Databases

## The ERP system would interact with the Student Information System (SIS) database to retrieve student information, such as demographic data, enrolment status, academic records, and attendance records. This integration ensures that the ERP system has access to up-to-date student information for administrative and academic purposes.

* + - * Financial Management System Database: The ERP system interacts with the financial management system database to retrieve financial data related to student fees, budget allocations, and expenditure tracking. This integration enables seamless financial management within the ERP system, including fee collection, billing, and budgeting.

## APIs (Application Programming Interfaces)

* + - * The ERP system integrates with a payment gateway API to facilitate online fee payments from students and parents. This integration enables secure payment processing within the ERP system, allowing users to make fee payments conveniently.
      * The ERP system might integrate with an SMS gateway API to send automated notifications and reminders to students, parents, and staff. This integration supports communication functionalities within the ERP system, such as attendance alerts, event reminders, and emergency notifications.

## Third-party Software

* + - * Communication Platforms: The ERP system integrates with communication platforms such as email servers to facilitate the doubt solving for students. This integration supports features such as email notifications within the ERP system.
      * The school utilizes a third-party resource for delivering online courses and assignments, the ERP system integrates this API to synchronize course enrolments, grades, and assignments between the two systems. This integration ensures seamless academic management across platforms.

## Product Functions

* + 1. **Registration**
       - According to vacancies present in the school, administrators allow the students to register into school for that academic year, and also can add the details of faculty, send verification mail to them.
       - Administration gives the basic details, few documents of the student or faculty.
       - The details will be stored in their school database.

## User Login

* + - * Students, Faculty and school administration can login into the system. According to their login credentials.
      * Username and password
      * Redirect into their respective profile pages

## User Profile

* + - * Profile consists of the basic details of the user. Like,
        1. Role
        2. Username
        3. Personal details like DOB (date of birth), residence, parents or guardian names, etc.

## Attendance tracking

* + - * Students can check their attendance in ERP. Faculty can enter attendance of their respective class students. Administrators can check the attendance of all the students.
      * This also depends on database connectivity and accessing the database, updating the details in the database.
      * This also depends on database connectivity and class schedule.

## Doubt solving

* + - * Students raise their doubt in their portal that they will share to their respective instructor through email and students get the response mail from the instructor. After the student gets the mail from the instructor, the student can delete the doubt from their portal.
      * Students enter their doubts in a clear and concise manner in text format.
      * Email will be sent to their instructor along with student mail Id.

## Fee payment

* + - * Students can pay their academic fees through a 3rd party online payment gateway.
      * Students need to select the payment gateway and need to login.
      * Payment details are reflected in their ERP.

## Examination marks

* + - * Students can check their exam results.
      * Faculty need to enter the student marks.
      * Marks will be updated in the students ERP.

## Finance department

* + - * Administration cross check the fee payments manually based on the payment details in the database.

## Student management & analysis

* + - * Administration can add and delete the student records from the database.

## Feedback

* + - * Students and faculty can provide the feedback related to UI & UX of their ERP.

## User Characteristics

* + 1. **Student**
       - Check their attendance, exam results, raise doubts, fee payment, feedback, library access

## Faculty

* + - * Enter student attendance, exam marks, get mails related to student doubts, can check their respective class students performance, library access.

## Administration

## Validation of registered users, check all students' performance, add and remove the materials from the library.

## Constraints

* Constraints on hardware resources, such as server capacity and network bandwidth, impacts system performance and scalability. Incomplete or inconsistent data available from legacy systems or manual records constrains data migration efforts, affecting the accuracy and completeness of data in the new ERP system.

## Assumptions and Dependencies

* + 1. **Functional Dependencies**
* Student Management: There is a functional dependency between the registration of students and their enrolment in classes. The process of registering students depends on the availability of enrolment slots in specific classes. Students can view their result, attendance, fee to be paid and balance, schedule in the ERP. Students can do transactions through the gateway and doubt clarification questions posed in the ERP will take the page to Gmail of that teacher to ask doubts.
* Staff Management: The assignment of teachers to courses depends on their qualifications and availability. This functional dependency ensures that only qualified teachers are assigned to teach specific subjects. The particular subject teacher can edit and view the marks, attendance of the student. Teachers can view the schedule to teach the students.
* Attendance Tracking: The process of recording student attendance depends on the class schedule and the presence of students in those classes. There is a functional dependency between the class timetable and the attendance tracking system. It is Done by teachers.
* School Management: The Management can keep track of all the information related to School. It can analyze fee dues, performance analysis of students, can view feedback of students about teachers and update the schedule.

## Non-functional Dependencies

* Performance: There is a non-functional dependency on system performance, where the ERP system handles a large volume of data efficiently, especially during peak times such as enrolment periods or report generation.
* Security: The system adheres to strict security requirements to protect sensitive student and staff information. This includes role-based access control, regular security audits.
* Scalability: The ERP system is scalable to less student enrolment and staff size. This non-functional dependency ensures that the system can adapt to changing needs over time.
* Usability: The usability of the ERP system is crucial for its adoption and effectiveness. There is a non-functional dependency on user-friendly interfaces, intuitive navigation, and comprehensive documentation to support users in performing their tasks efficiently.

## External Interface Requirement

* 1. **User Interface**

Our user interface prioritizes accessibility and efficiency, ensuring that students, faculty, and administrators can effortlessly navigate through the portal to access the myriad of features and services it offers. Through careful consideration of user experience principles, we have implemented a clean and responsive design that adapts seamlessly across various devices, providing a consistent and engaging interaction regardless of the platform.

To enhance user engagement, we have incorporated modern design elements, employing a cohesive color palette and intuitive layout. The navigation system has been meticulously structured to streamline access to essential features, minimizing any potential learning curve for users. Through extensive usability testing and feedback iterations, we have refined the interface to prioritize clarity and functionality, resulting in an interface that is not only aesthetically pleasing but also highly practical.

Additionally, our user interface design aligns closely with the overall branding and ethos of our educational institution, creating a sense of familiarity and coherence for users. Customizable elements have been integrated to allow for personalization, ensuring that individuals can tailor their experience within the portal to suit their preferences and requirements.

In conclusion, the user interface component of our student ERP portal reflects a collaborative effort that emphasizes both form and function. The design choices made not only prioritize the user's journey through the portal but also contribute to an overall positive and engaging experience. We are confident that our user interface design will play a pivotal role in facilitating seamless interactions and optimizing the overall usability of our student ERP portal.

## Hardware Interface Requirements

* **Device Compatibility**
  1. Interacts with other devices and exchanges the data, to perform their functions without errors and malfunctions.
  2. Software is compatible with devices like desktops, tablets, smartphones. Most of the processors in the devices.

## Resource Optimization

* 1. Optimizing hardware requirements by efficiently allocating resources like CPU, RAM, and storage based on the anticipated workload, ensuring a balance between performance and cost.
  2. Energy-efficient hardware components like reduced power consumption, optimizing data structures and algorithms contribute to environmental sustainability.

## Software Interfaces

* + 1. **Admin Interface**

The administrative interface in a school management system is a crucial component that allows administrators to efficiently manage various aspects of the system.

* + - * User Accounts Management: This feature enables administrators to create, update, and delete user accounts within the system. Administrators can add new users, such as teachers, students, and parents, and assign them appropriate roles and permissions. They can also manage existing user accounts, reset passwords, and deactivate accounts as needed.
      * Permissions Management: Administrators have the authority to define and manage permissions for different user roles within the system. This includes specifying which features, data, and functionalities each user role can access and modify. For example, administrators may grant teachers permission to enter grades and attendance but restrict access to administrative settings.
      * System Configurations: Administrators can manage system configurations related to security, data management, and integration with other systems. This includes configuring backup and recovery settings, specifying data retention policies, and managing integration with third-party applications or services such as student information systems (SIS).

## Teacher Interface

The Teacher Interface is a critical component of the School Management System, providing teachers with the tools necessary to efficiently perform various tasks related to classroom management, grading, attendance tracking, and communication with students and parents.

* + - * Class Schedule Management: The interface allows teachers to view, create, and manage class schedules for different subjects and grade levels. Teachers should be able to specify the date, time, duration, and location of each class session.
      * Grade Entry and Management: Teachers are able to enter, update, and manage grades for exams and other assessments. The interface provides a gradebook where teachers can view student grades, calculate averages, and generate progress reports. In this teacher interface, Teachers should be able to assign weights to different types of assessments (e.g., quizzes) and calculate overall course grades based on weighted averages.
      * Attendance Tracking: This interface allows teachers to take attendance for each class session, marking students as present and absent. Teachers should be able to view attendance records for individual students and track attendance trends over time. The interface should support the generation of attendance reports and notifications for students with frequent absences.
      * Resource Sharing and Course Materials: The interface provides a platform for sharing course materials, lecture notes, presentations, and supplementary resources with students. Teachers will be able to upload and organize files into folders, categorize resources by subject or topic, and grant access permissions to students and parents.

## Student Interface

The Student Interface is a crucial component of the School Management System, providing students with a user-friendly platform to access course materials, submit assignments, view grades, check attendance, and access other relevant information related to their academic progress and school activities.

* + - * Course Materials Access: The interface should provide students with access to course materials such as lecture notes, presentations, readings, and multimedia resources. Students can download or view course materials online through this interface.
      * Grade Viewing: Students can view their grades for assignments, quizzes, and examinations through the interface. Grades will be displayed in a clear and organized manner, allowing students to track their progress over time. The interface should support the display of detailed feedback and comments provided by teachers for each graded assignment.
      * Attendance Checking: Students will have access to their attendance records through the interface, allowing them to view their attendance history for each class session.

## Communications Interfaces

* **Frontend to Backend API Communication:**
  + Frontend interfaces with backend APIs to fetch and manipulate data.
  + APIs serve as the bridge between frontend user interfaces and backend databases or services.
  + Technologies such as RESTful API utilized for this communication.

## User Interface (UI) Components:

* + Interfaces between different UI components within the frontend application.
  + Facilitates communication for data sharing and user interaction.
  + Components may communicate through event listeners, state management libraries, or custom messaging systems.

## Database Interfaces:

* + Backend communicates with one or multiple databases to store and retrieve data.
  + This interface involves database management systems and database query languages (e.g., SQL).
  + Object-Relational Mapping (ORM) frameworks serve as interfaces between application code and databases, abstracting away low-level database interactions.

## External Service Integration:

* + Interfaces with external services such as payment gateways, messaging services, storage.
  + Communication protocols and APIs provided by these services are utilized for integration.
  + This ensures features like online payments, notifications, and data backups.

## Authentication and Authorization Interfaces:

* + Handles communication related to user authentication and authorization.
  + Involves login/logout mechanisms, token-based authentication, and permission management.

## Real-Time Communication:

* + Interfaces for real-time communication between users or system components.
  + Utilizes technologies like Web Sockets or server-sent events (SSE) for instant messaging, notifications, or collaborative features.

## Error Handling and Logging:

* + Interfaces for logging and reporting errors and system events.
  + Logs will be stored locally or in centralized logging systems.
  + Error handling mechanisms ensure graceful degradation and proper reporting of errors to users and administrators.

## Testing and Debugging Interfaces:

* + Interfaces for testing the application components and debugging issues.
  + Includes unit testing frameworks, debugging tools, and logging mechanisms for development and troubleshooting purposes.

## Version Control and Collaboration Interfaces:

* + Interfaces for version control systems like Git for collaborative development.
  + Enables developers to manage codebase changes, merge code, and resolve conflicts efficiently.

# System Features:

## Student Management:

* Enrolment management
* Student profiles and records
* Attendance tracking
* Academic performance monitoring

## Teacher Management:

* Teacher profiles and records
* Assignment and grade management
* curriculum management
* Communication tools for teachers Email

## Administrative Tools:

* Staff management
* Timetable/schedule management
* Study resource allocation and management
* Student analytics

## Attendance Management:

* Automated attendance tracking
* Notifications for absenteeism

## Academic Management:

* Course management
* Examination and assessment management
* Grading and transcript generation
* Curriculum planning and mapping

## Communication Tools:

* Internal messaging system
* Announcements and alerts
* Parent-teacher communication channels
* Integration with email

## Fee Management:

* Fee collection and invoicing
* Online payment gateway integration
* Fee tracking

## Security and Access Control:

* User authentication and authorization
* Role-based access control
* Audit trails for system activities.

1. **Other Non-Functional Requirements**
   1. **Performance Requirements**
      1. **Capacity**
         * Database Capacity: The database is capable of handling a large volume of data, including student records, course information, attendance logs, grades, and administrative data. It is also scalable to accommodate future growth in the number of users, courses, and data volumes without compromising performance.
         * Server Capacity: The server infrastructure hosting the application has sufficient processing power, memory, and storage capacity to handle concurrent user requests and system operations.

## Dynamic Requirements

* + - * Response Time: The system responds to user requests promptly, with minimal latency, to provide a smooth and seamless user experience. Web pages and application interfaces should load quickly, with response times ideally within a few seconds, even during peak usage periods.
      * Scalability: Scalability mechanisms such as auto-scaling, sharding, and clustering are implemented to dynamically allocate resources and maintain optimal performance.
      * Concurrency: The system supports multiple concurrent users accessing and interacting with the application simultaneously without degradation in performance.

## Quality

* + - * Reliability: The system will be reliable and available for use 24/7, with minimal downtime and interruptions. We implement Fault tolerance mechanisms to handle software errors and unexpected exceptions gracefully.
      * Security: The system adheres to industry best practices for security, including data encryption, access controls, and user authentication mechanisms. Sensitive data such as student records, grades, and financial information will be protected from unauthorized access, disclosure, or modification.
      * Maintainability: We use Version control systems to track changes to the codebase, and automated testing and deployment pipelines to streamline development workflows.

## Software System Attributes

* + 1. **Reliability**

1. Error handling will be implemented by us to ensure the system quickly handles errors to maintain smooth performance.
2. Fault tolerance is implemented so that the system keeps working even if unexpected issues occur.

## Availability

1. System availability standards, specify goals for consistent system availability with minimal downtime.
2. Backup strategy, include backup components to prevent service interruptions.

## Security

a. Access controls only allow authorized users to access and modify system resources.

## Maintainability

1. Modularity and Documentation makes the system designing with easy to-understand modules and comprehensive documentation for straightforward maintenance.
2. Code Maintainability helps to follow coding standards and best practices for code that is easy to understand and maintain over time.

## Business Rules

Within the framework of our student ERP project, the integration of essential business rules stands as a cornerstone in ensuring the system's efficacy and coherence. These business rules serve as the governing principles that dictate how data is processed, accessed, and managed within the portal. Through meticulous analysis of institutional requirements and collaboration with stakeholders, we have identified and implemented a robust set of business rules designed to uphold the integrity of information, maintain consistency in processes, and align with the overarching goals of the educational institution.

The business rules within our ERP system encompass a diverse range of functionalities, including but not limited to student enrolment, course registration, grading policies, and administrative workflows. For instance, enrolment rules govern eligibility criteria, ensuring that only qualified students are admitted, while course registration rules facilitate a smooth and organized process for students to select and enroll in their desired classes. Additionally, grading rules have been established to standardize the evaluation process, promoting fairness and transparency.

To enhance data security and privacy, stringent access control rules have been implemented, defining user roles and permissions based on the principle of least privilege. This ensures that individuals within the institution only have access to the information necessary for their specific roles, safeguarding sensitive data and maintaining compliance with privacy regulations.

Continuous refinement and adaptation of these business rules are integral components of our project development strategy. Regular consultations with stakeholders and end-users enable us to identify evolving needs and potential enhancements to existing rules, ensuring that the ERP system remains flexible and responsive to the dynamic nature of educational processes.

In essence, our diligent attention to the formulation and implementation of business rules establishes a solid foundation for the functionality, reliability, and security of our student ERP portal. These rules not only serve as the guiding principles for data governance but also contribute to the overall efficiency and effectiveness of the system, aligning seamlessly with the strategic objectives of our educational institution.

**4. Functional and nonFunctional requirements**

**Functional requirements** define what the system should do.

**1. User Management:**

1. Secure registration and login processes for students, faculty, and administrative staff.
2. Role-based access control to ensure users can only access information pertinent to their roles.

**2. Student Information Management:**

1. Enrollment and admissions processing.
2. Maintenance and retrieval of student records, including personal details, academic performance, and attendance.

**3. Academic Management:**

1. Curriculum and lesson planning tools.
2. Timetable generation and management.
3. Assignment distribution and grade recording.

**4. Financial Management:**

1. Processing and recording of fee payments.
2. Generation of billing statements for students.
3. Budgeting and financial reporting functionalities.

**5. Attendance Tracking:**

1. Digital recording of student attendance.
2. Ability for faculty to update attendance records.
3. Reports on attendance for parents and administrators.

**6. Communication Tools:**

1. Integration with SMS and email for sending notifications to students and parents.
2. Platforms for students to submit queries and receive responses from faculty.

**7. Fee Payment System:**

1. Integration with external payment gateways.
2. Secure transaction processing.
3. Real-time updates of payment statuses in student profiles.

**8. Examination and Assessment Management:**

1. Scheduling of examinations.
2. Entry and update of student marks.
3. Generation and dissemination of progress reports and transcripts.

**9. Reporting Tools:**

1. Generation of reports on academic performance, financial status, and other administrative metrics.
2. Customizable report features to meet the specific needs of the institution.

**Non-functional requirements** define how the system should behave on its functionality:

**1. Performance:**

1. The system should be capable of handling a large number of simultaneous users without degradation of performance.
2. Quick response times for user interactions and data processing.

**2. Reliability:**

1. High availability with minimal downtime.
2. Ability to recover quickly from failures without data loss.

**3. Usability:**

1. Intuitive user interfaces that are easy to navigate for all user groups, including students, faculty, and administrators.
2. Comprehensive help documentation and support for users.

**4. Scalability:**

1. The ability to scale resources up or down based on the number of users and data volume.
2. System design should accommodate growth in user base and data without the need for a complete redesign.

**5. Security:**

1. Strong authentication and authorization to protect sensitive data.
2. Encryption of data in transit and at rest.
3. Regular security audits to identify and mitigate vulnerabilities.

**6. Maintainability:**

1. The system should be easy to update and maintain without significant downtime.
2. Modular architecture to facilitate updates and maintenance.

**7. Compliance:**

1. Compliance with educational standards and regulations.
2. Adherence to data protection regulations such as GDPR for handling personal information.

**5. Technology used**

**Express.js:** Express.js is a web application framework for Node.js. It simplifies the process of building web applications and APIs by providing a robust set of features for routing, middleware, handling HTTP requests, and more. In this project, Express.js is used to create the server-side application.

**pg (node-postgres)**: pg is a PostgreSQL client for Node.js. It allows Node.js applications to interact with PostgreSQL databases by providing a straightforward API for executing SQL queries and managing database connections.

**EJS (Embedded JavaScript):** EJS is a simple templating language that lets you generate HTML markup with plain JavaScript. It enables embedding JavaScript code directly into HTML files, allowing for dynamic content generation on the server-side. In this project, EJS templates are used to generate HTML pages dynamically based on the application's state and data retrieved from the database.

**Bootstrap**: Bootstrap is a popular front-end framework for building responsive and mobile-first websites. It provides pre-designed UI components and layout utilities that make it easy to create visually appealing web applications with minimal effort. In this project, Bootstrap is used for styling and layout purposes.

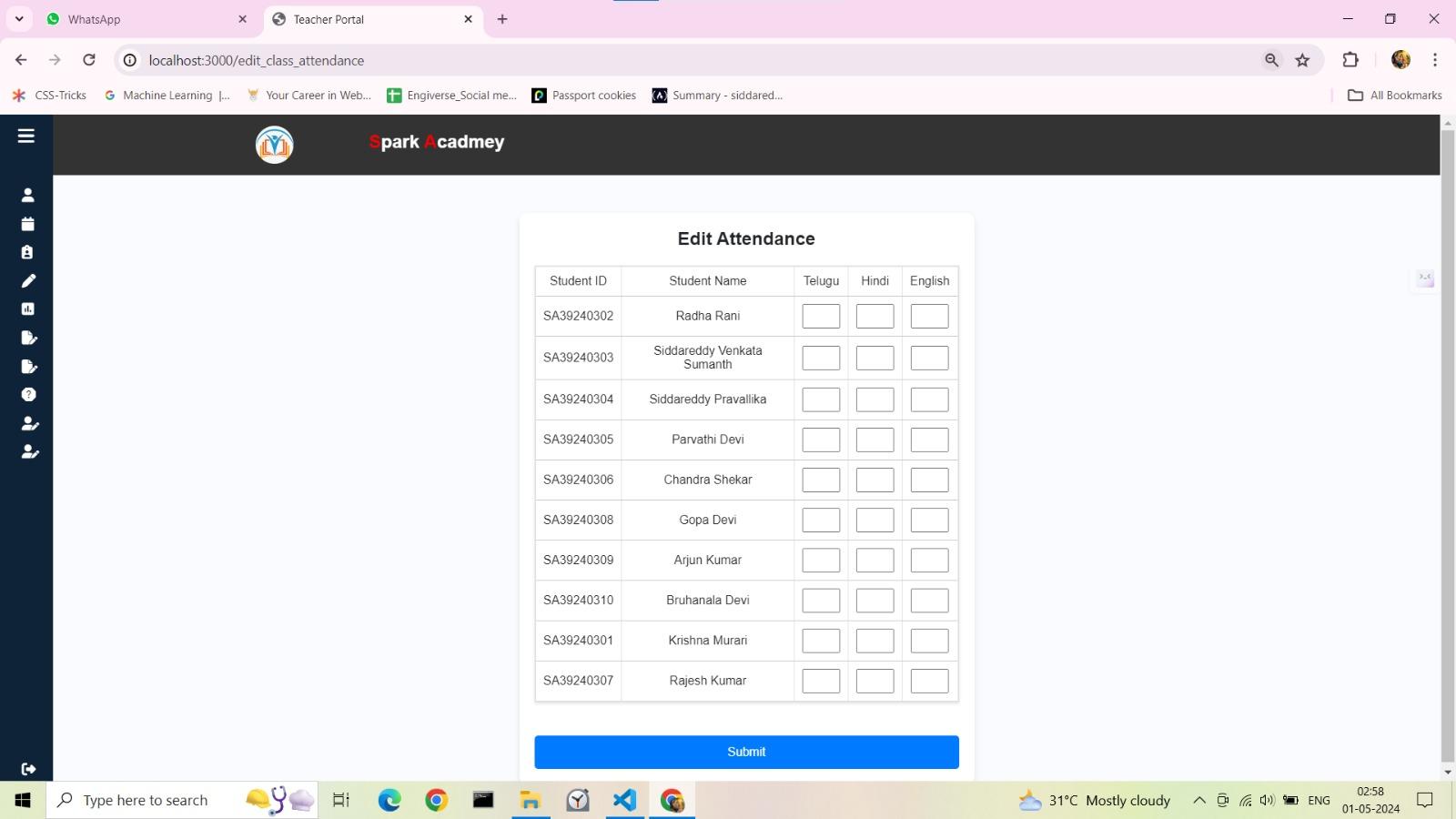
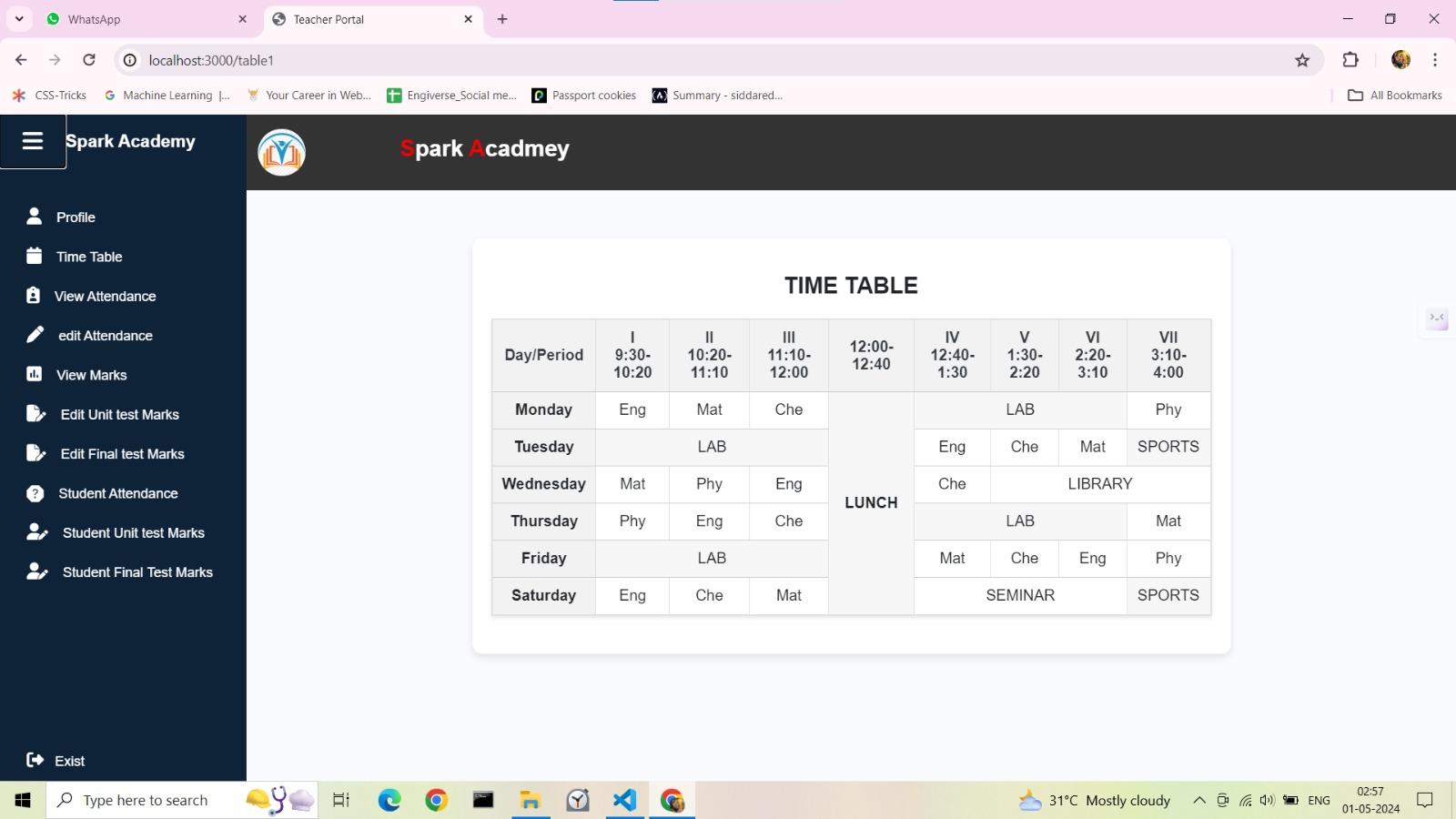
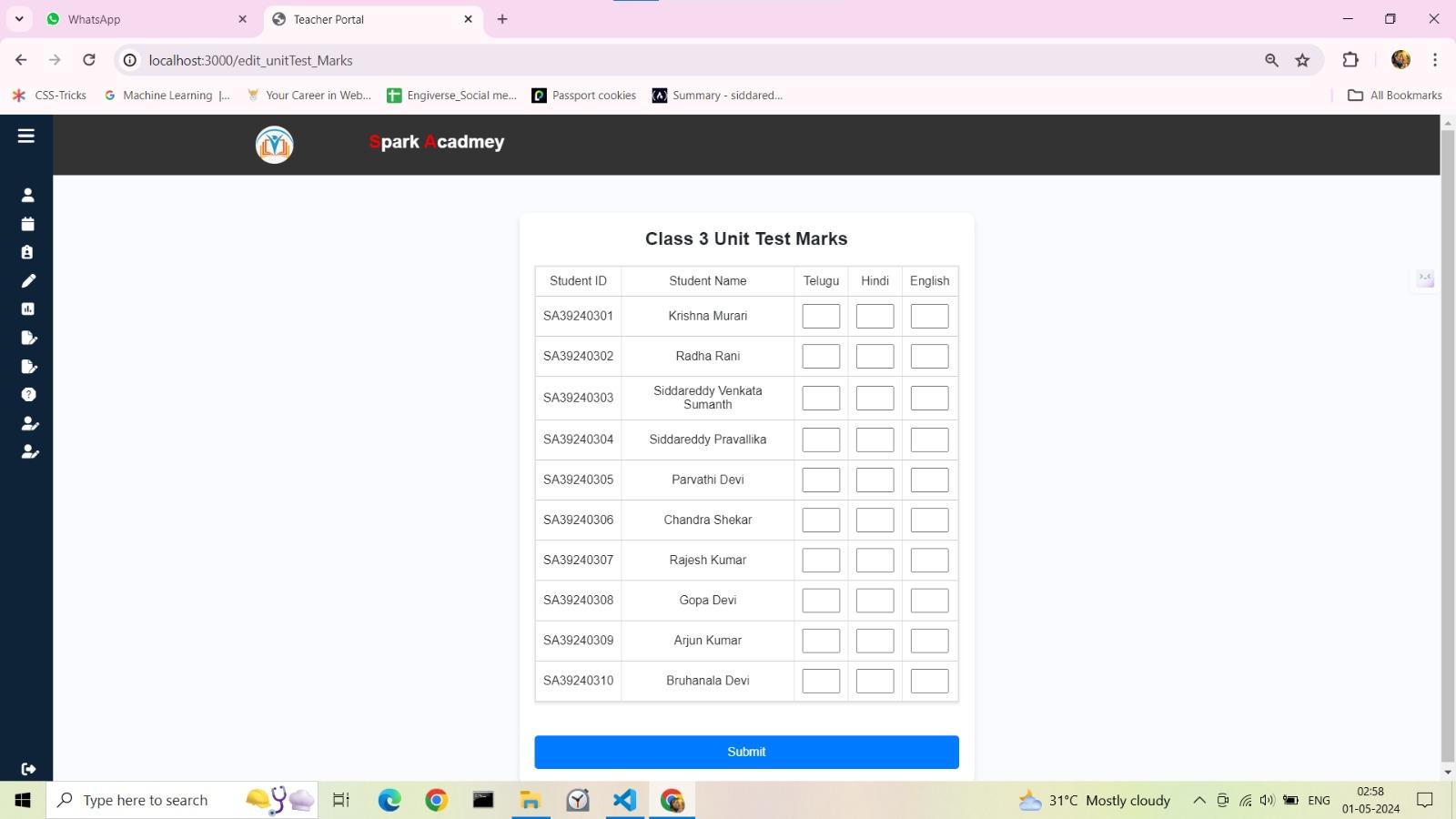
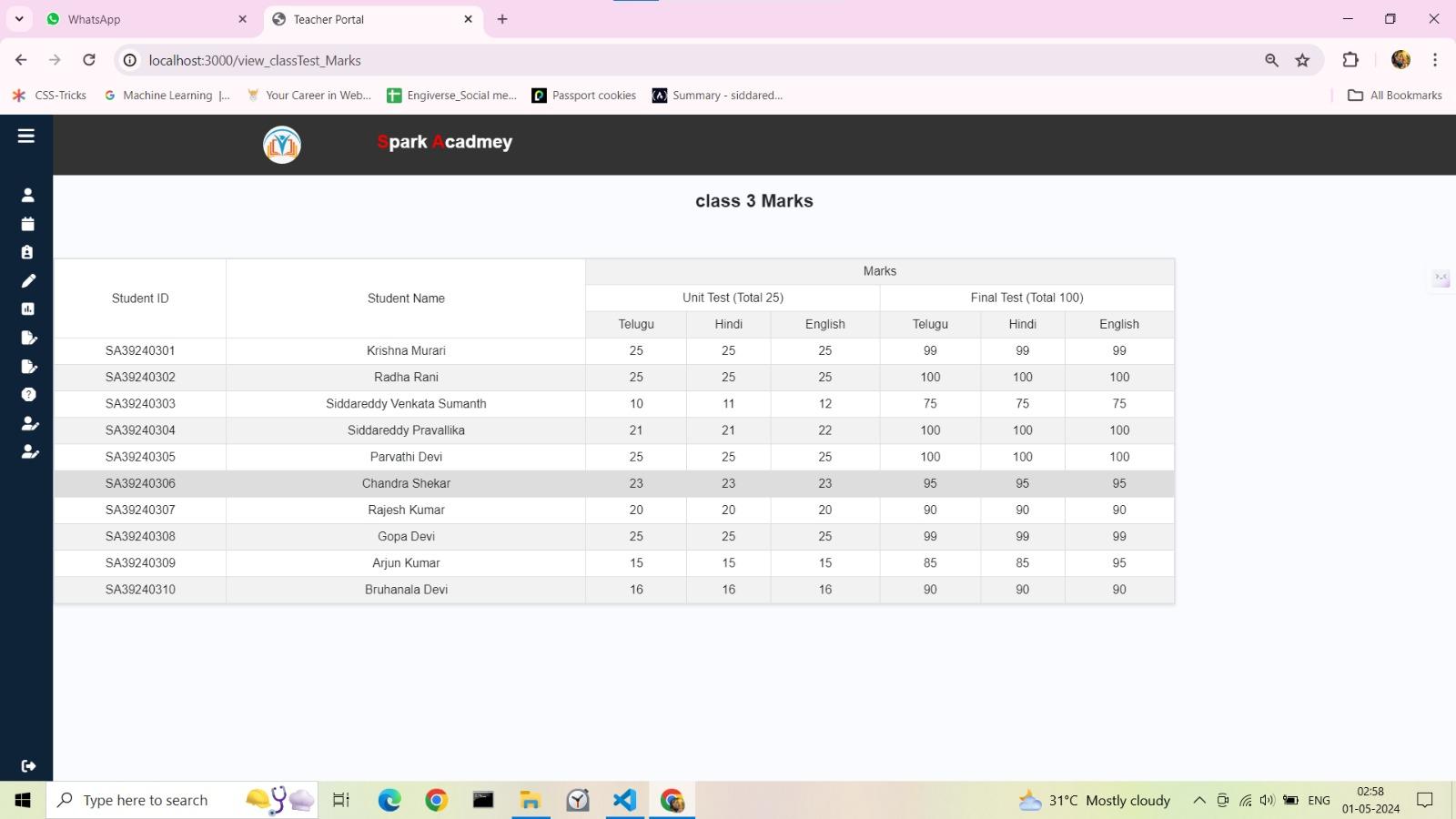
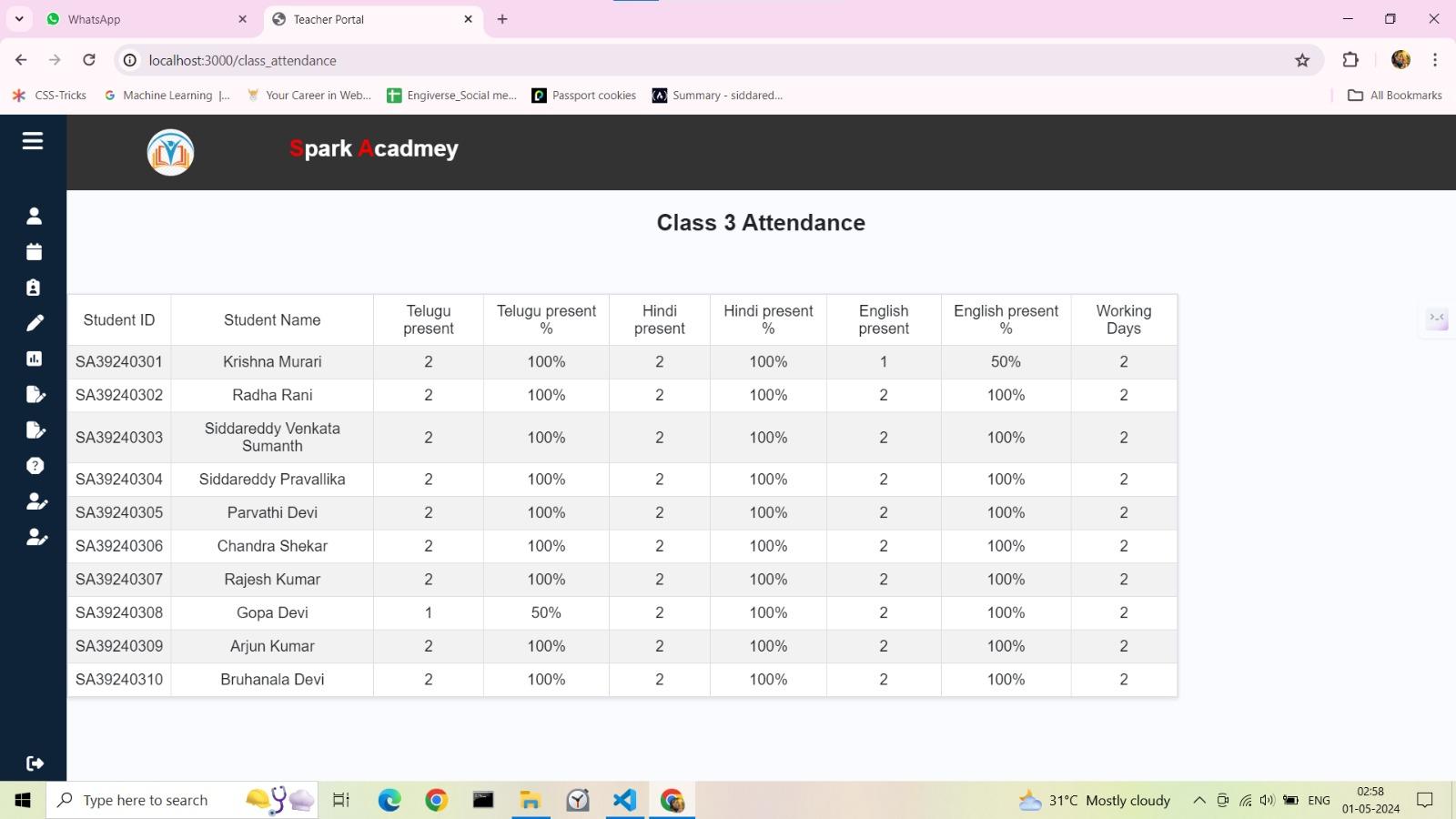
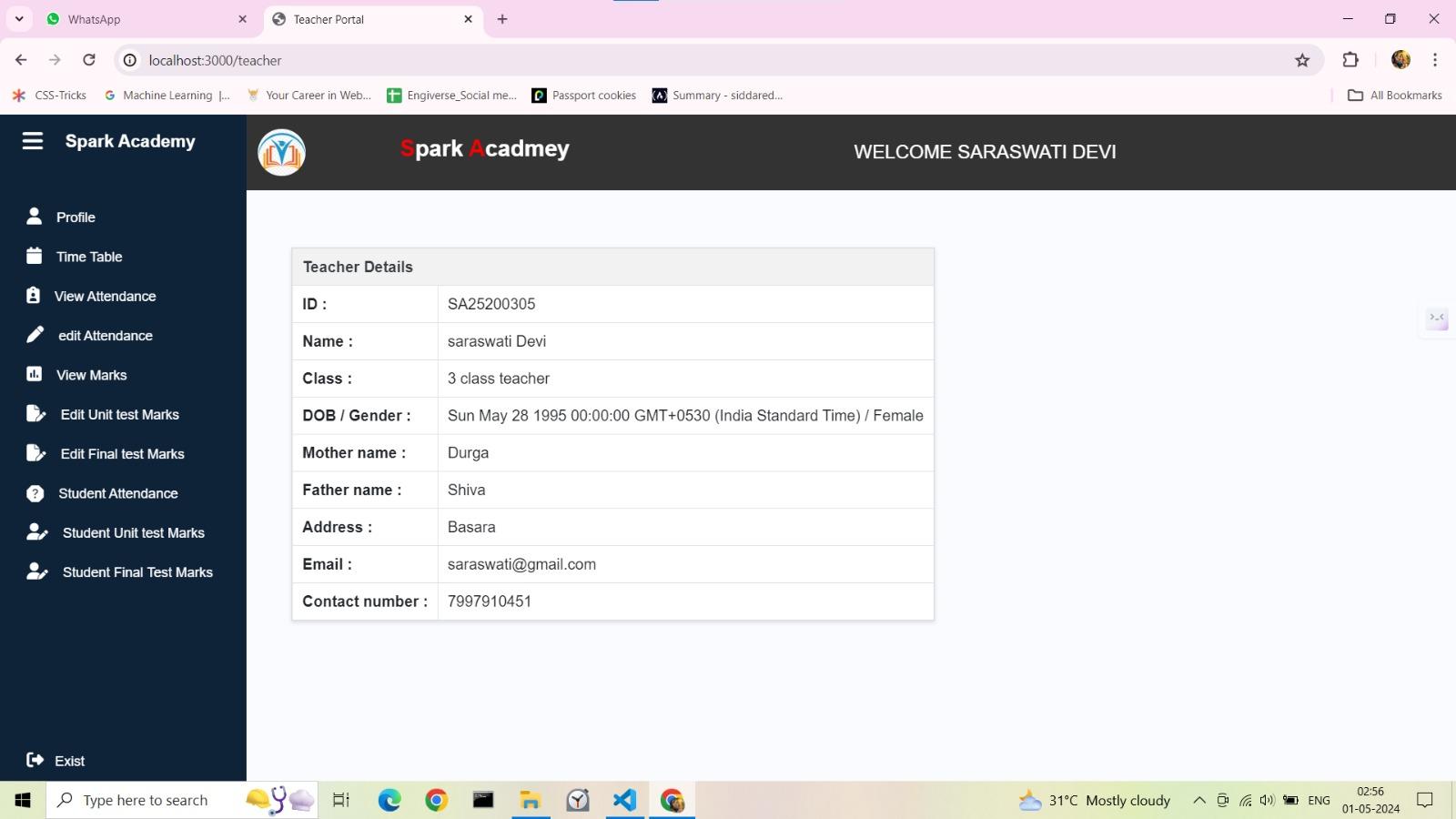
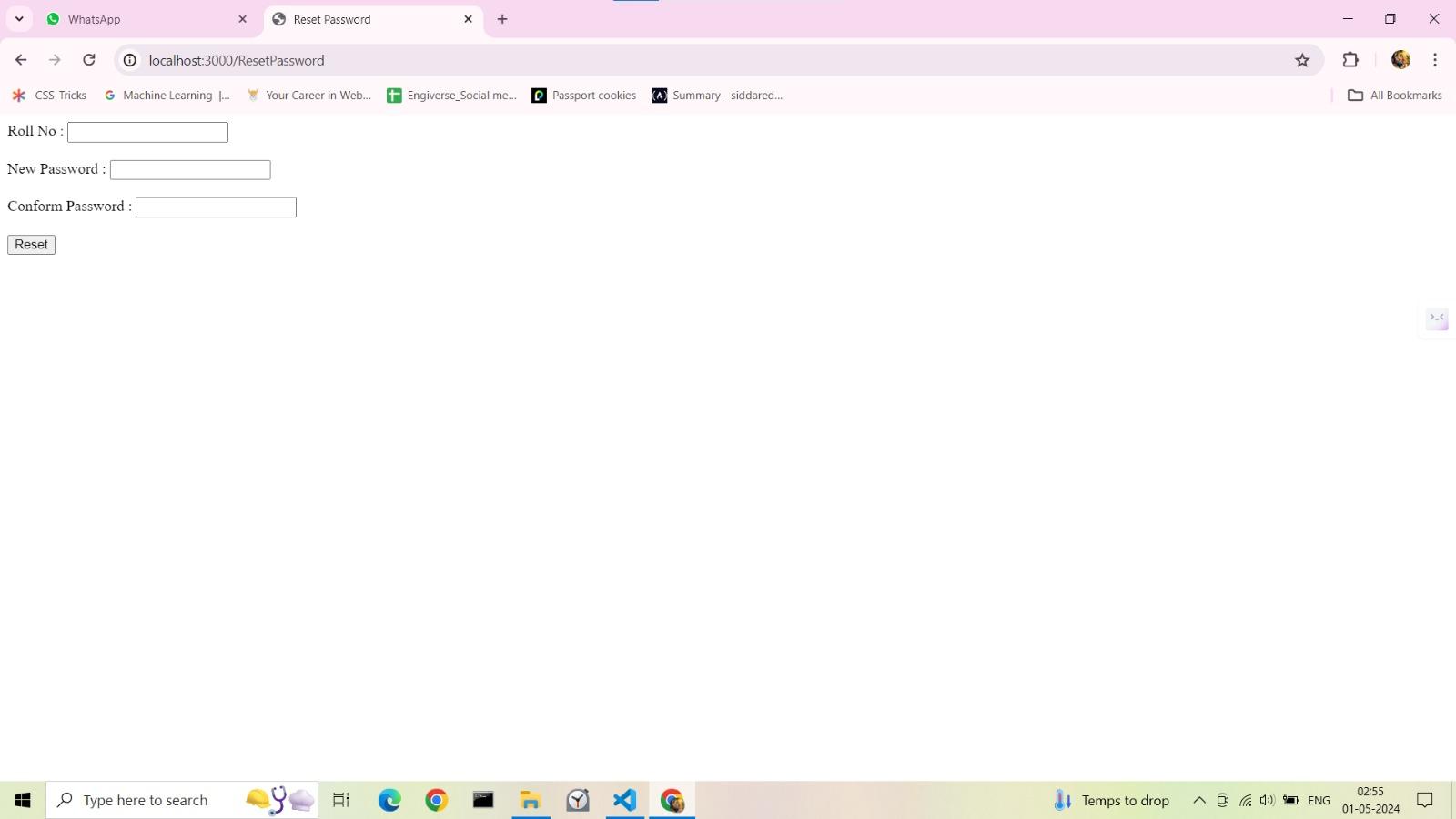
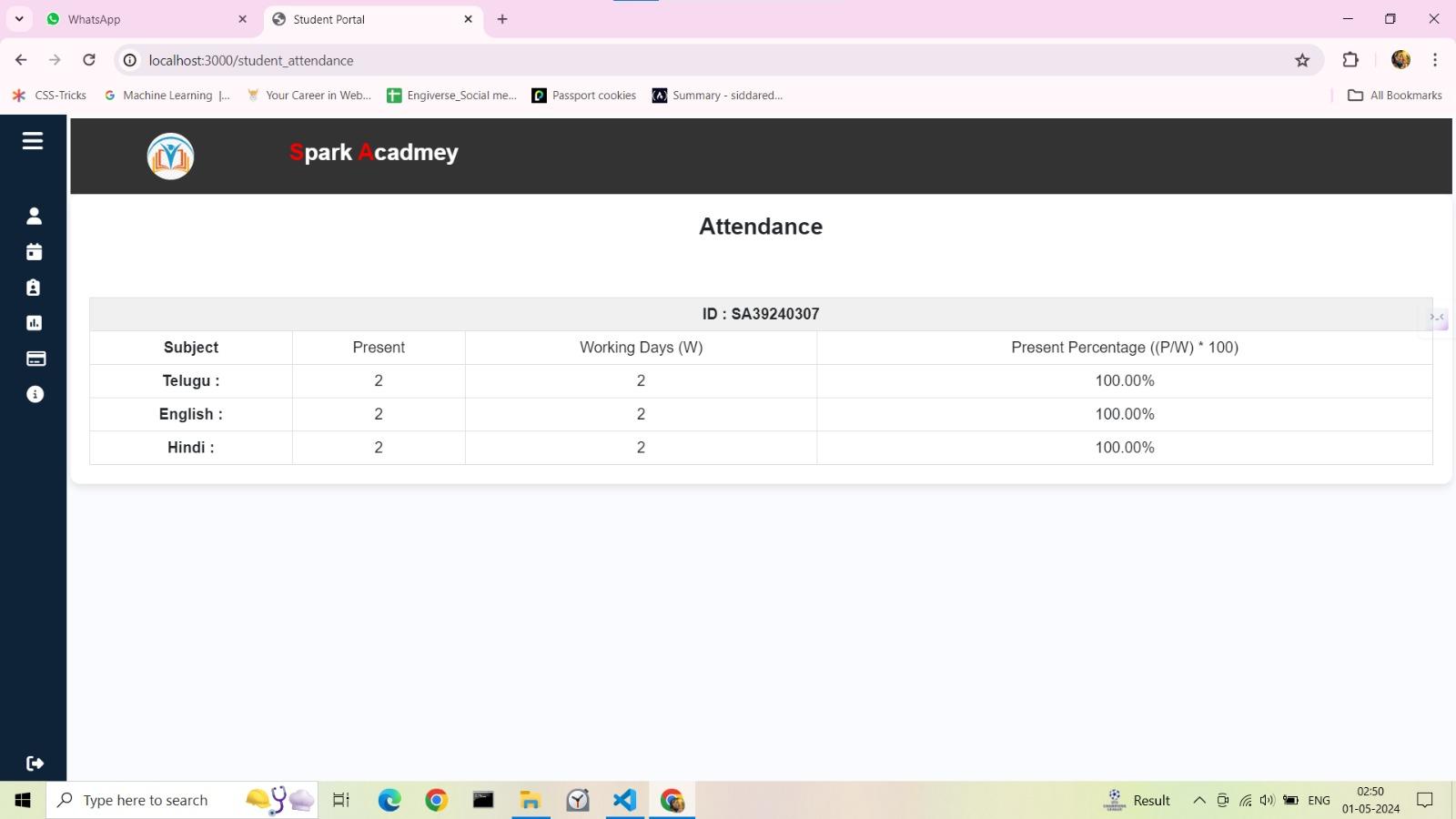
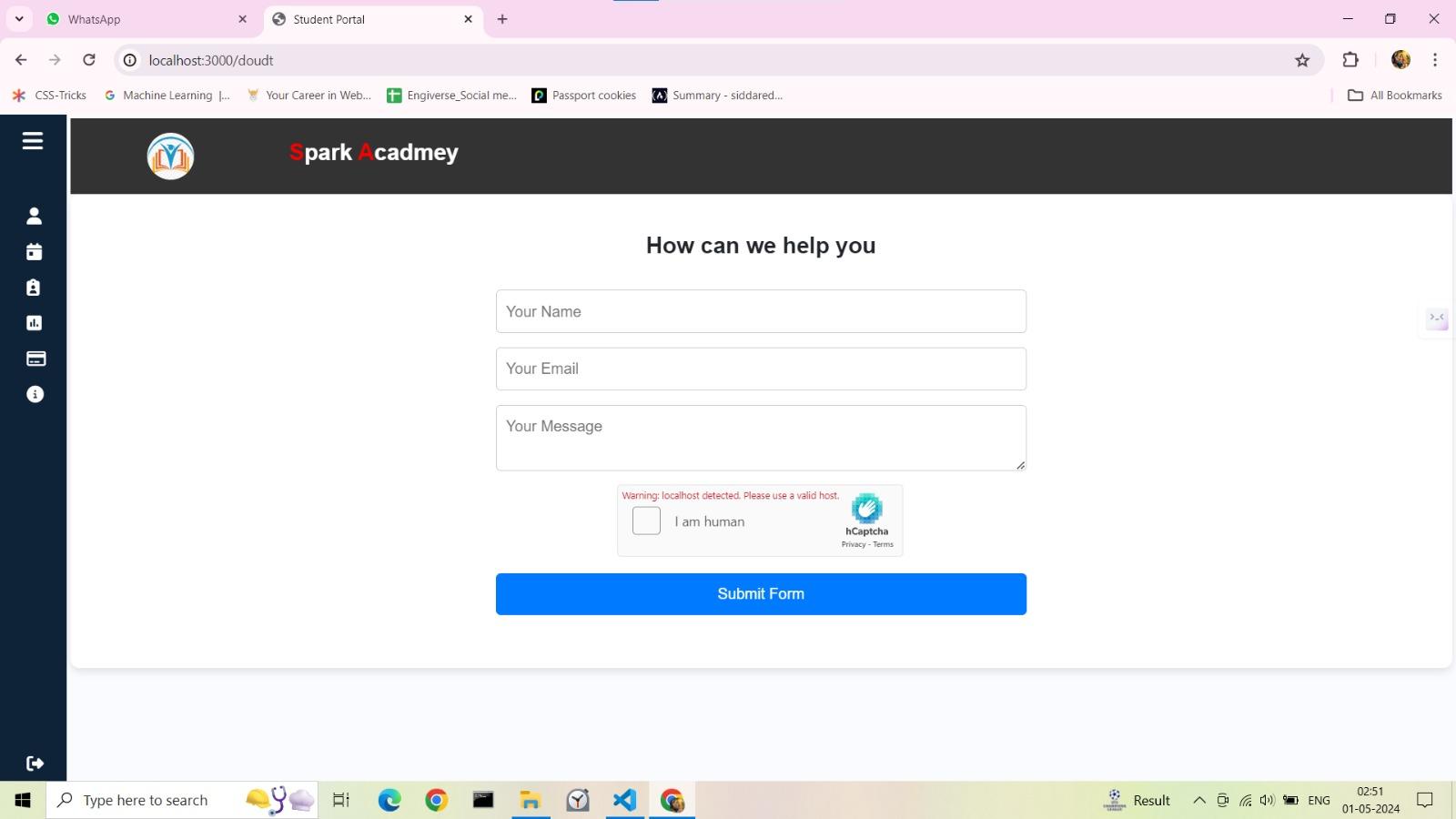
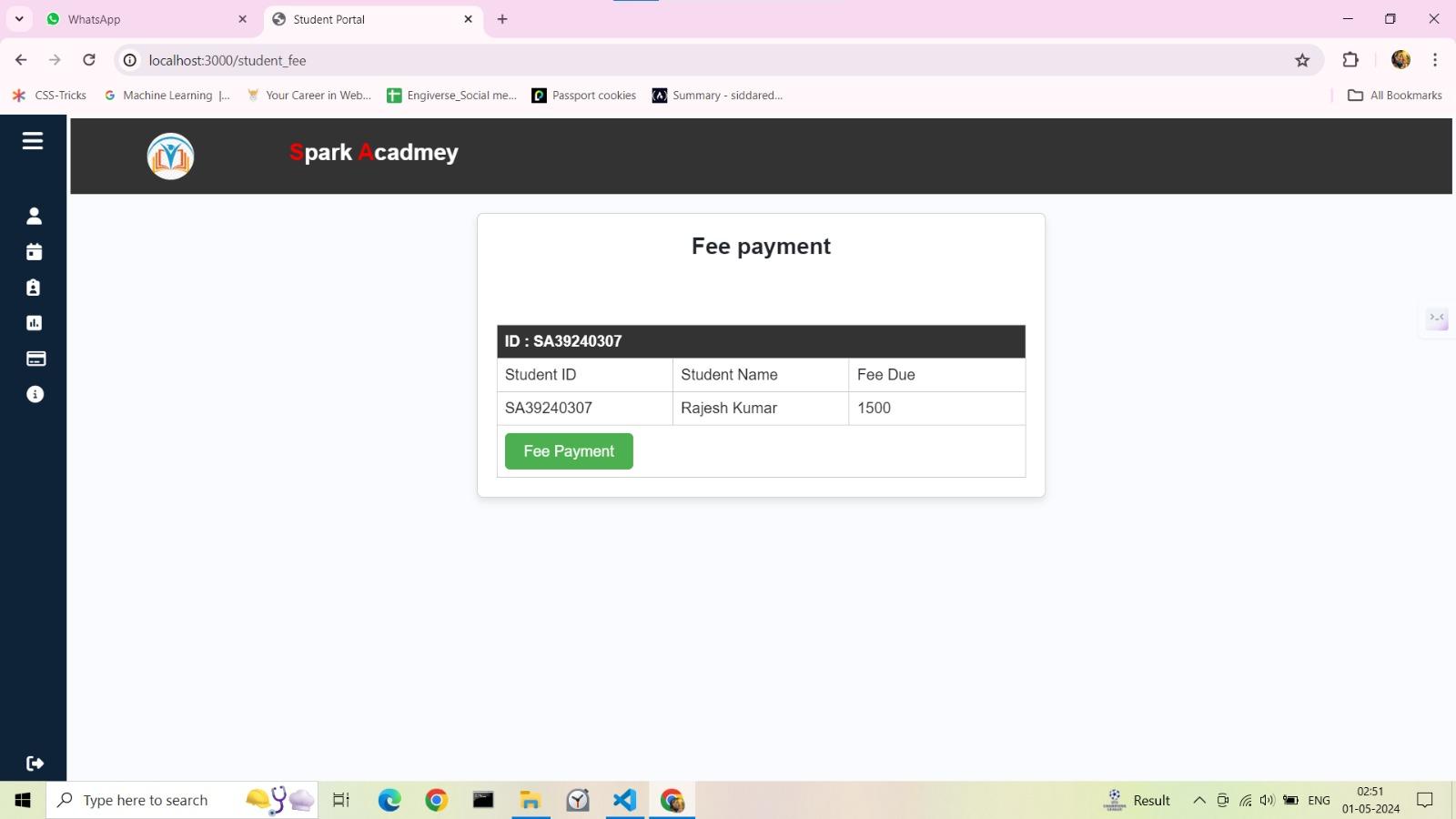
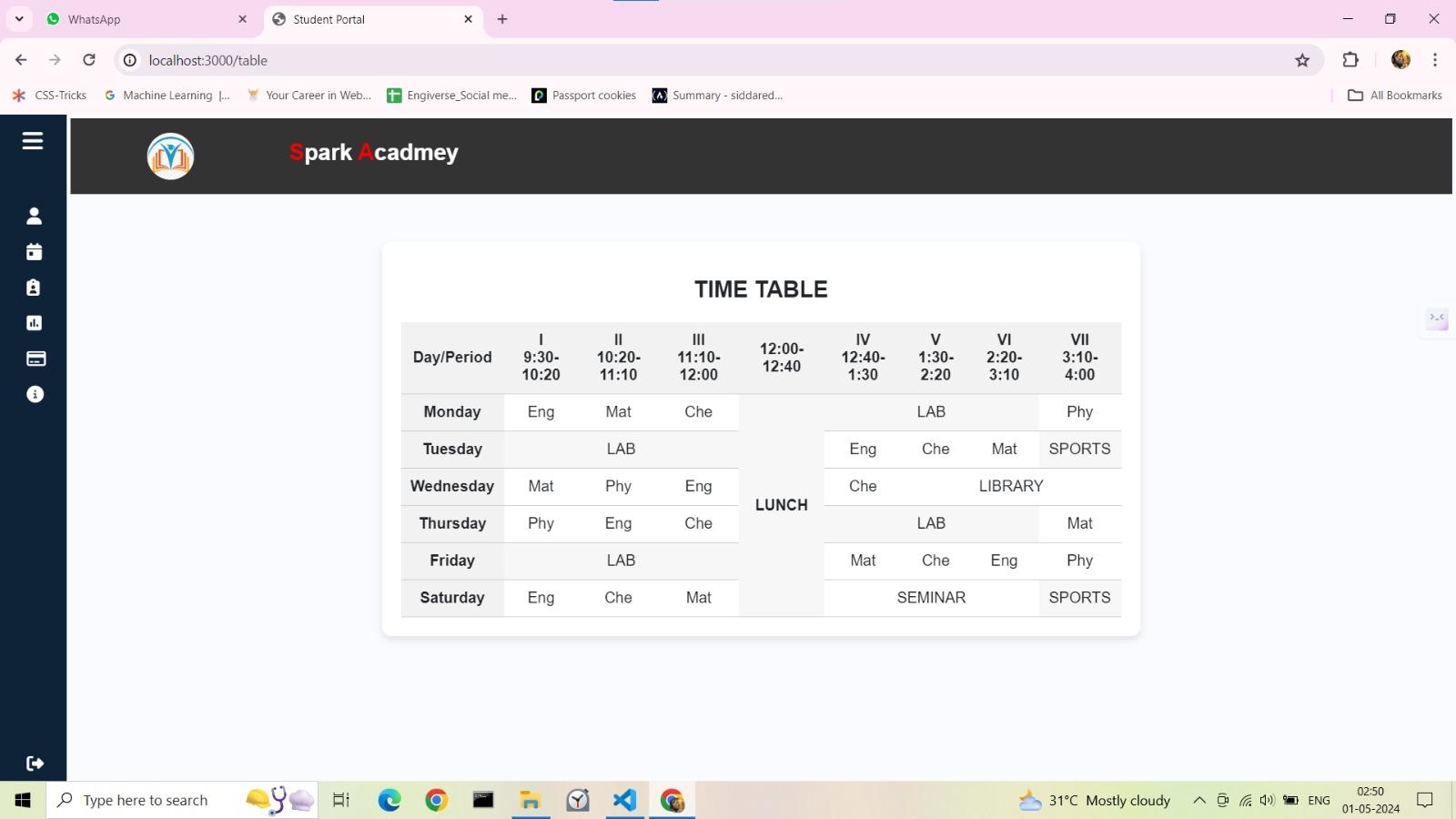
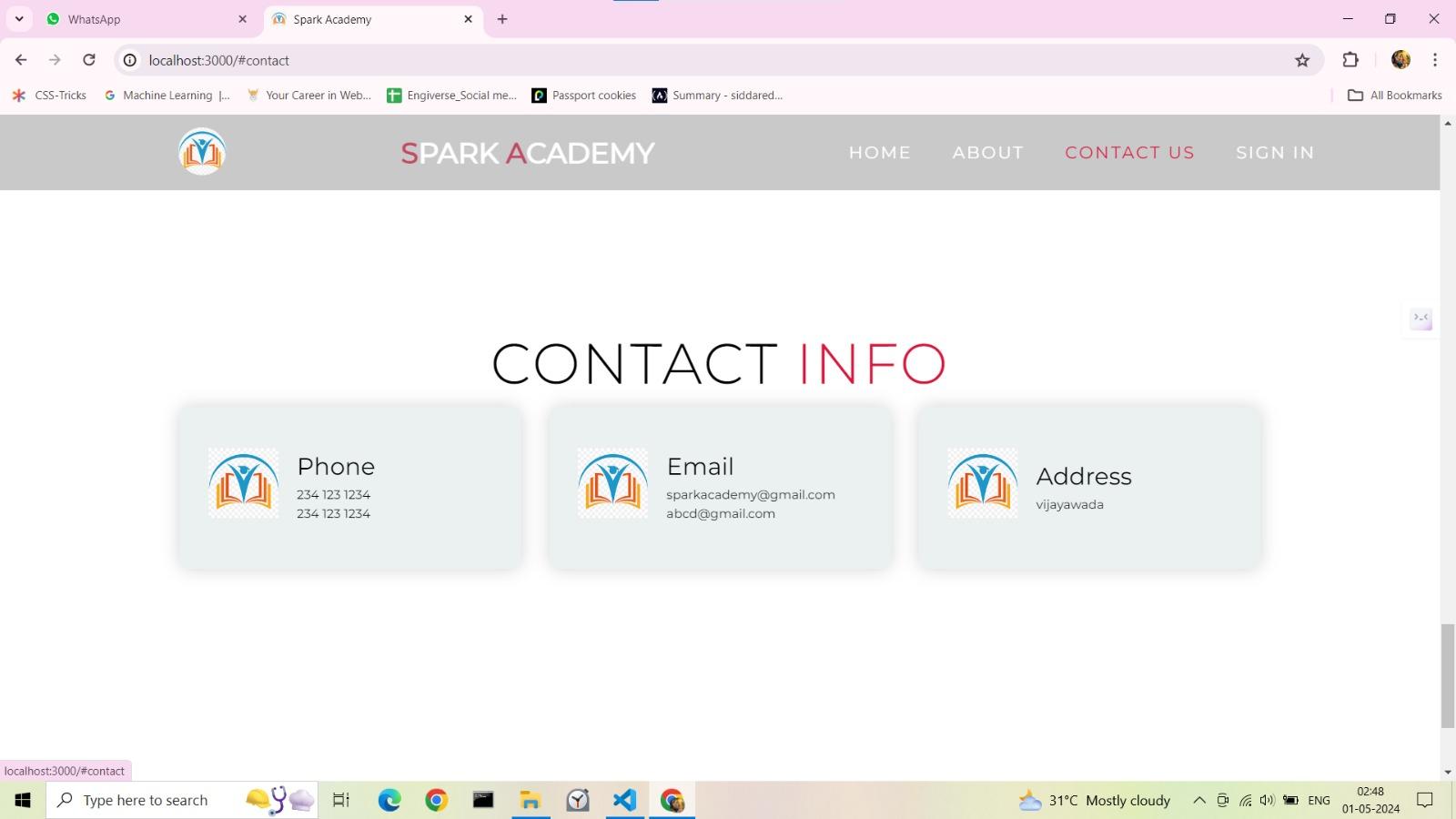
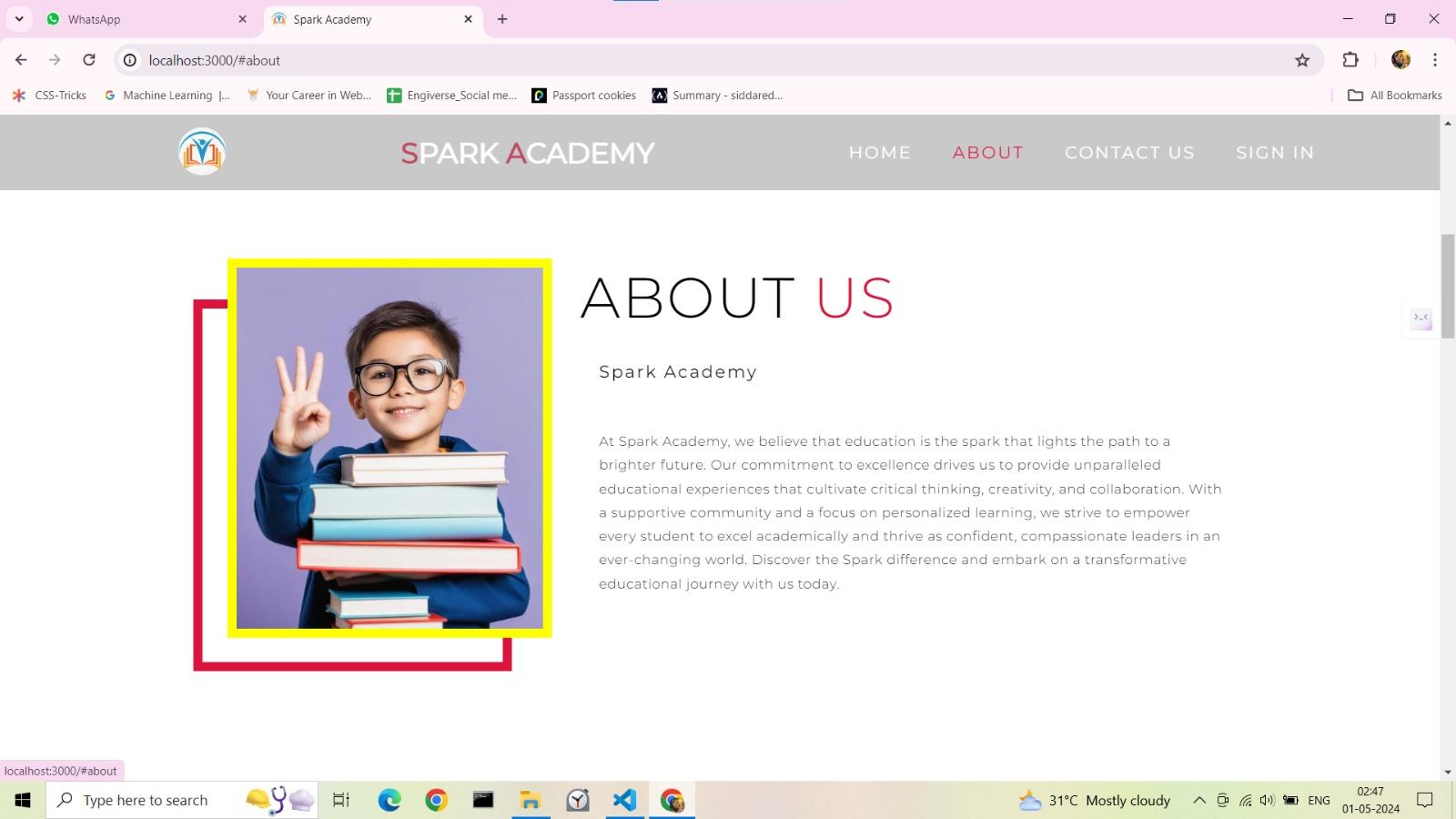
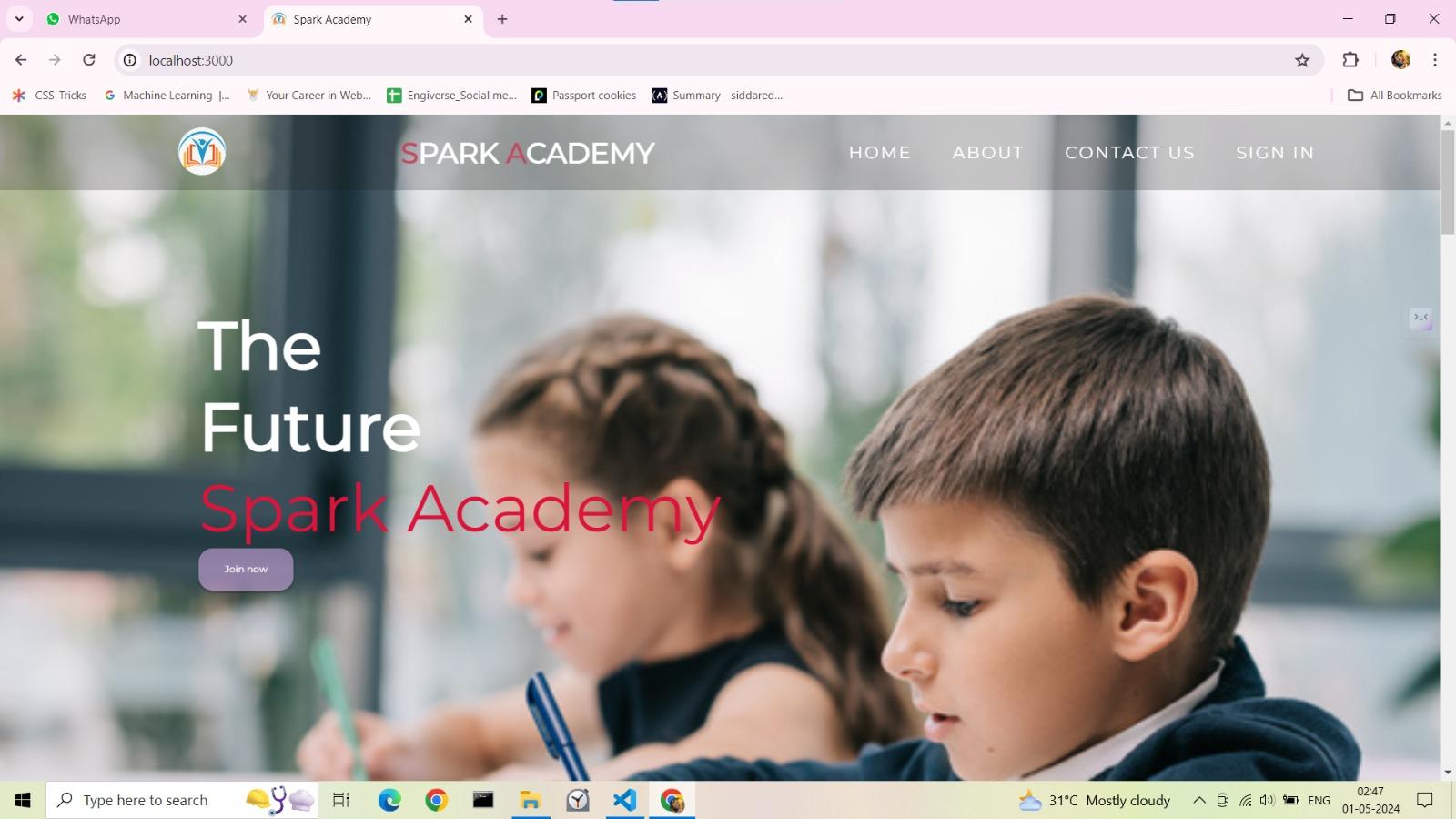
**6. Description of Front-end and Back-end**

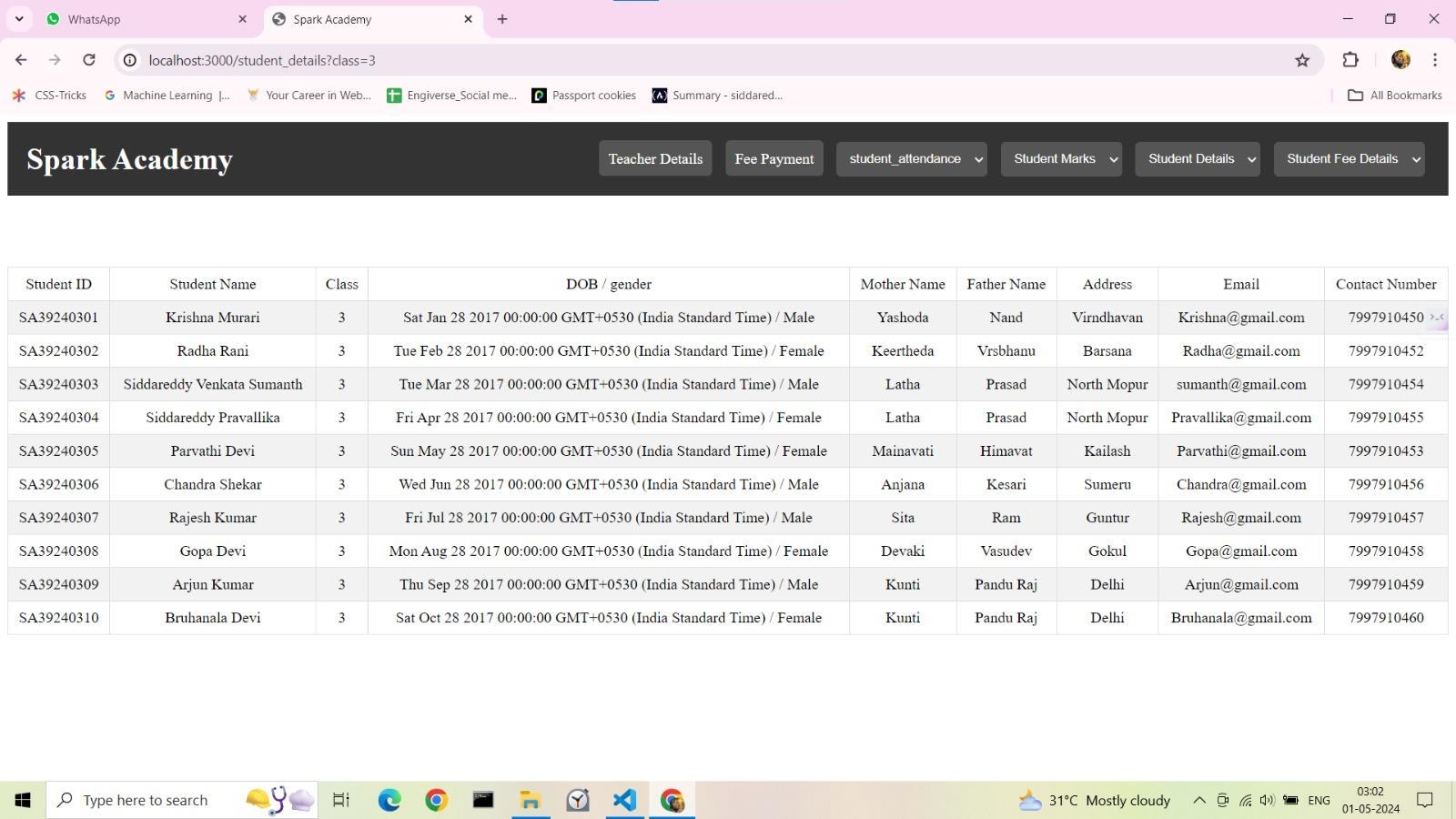
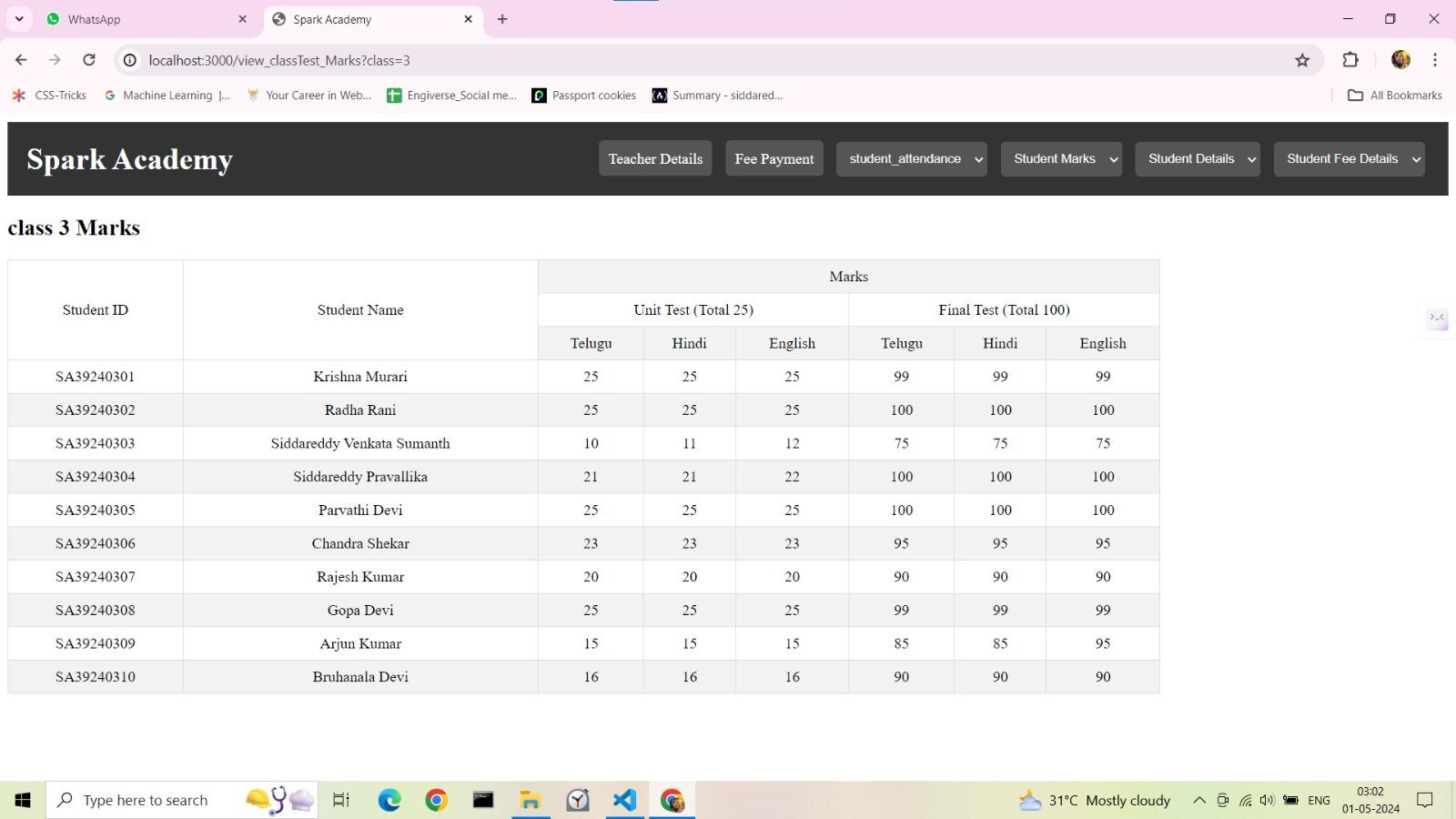
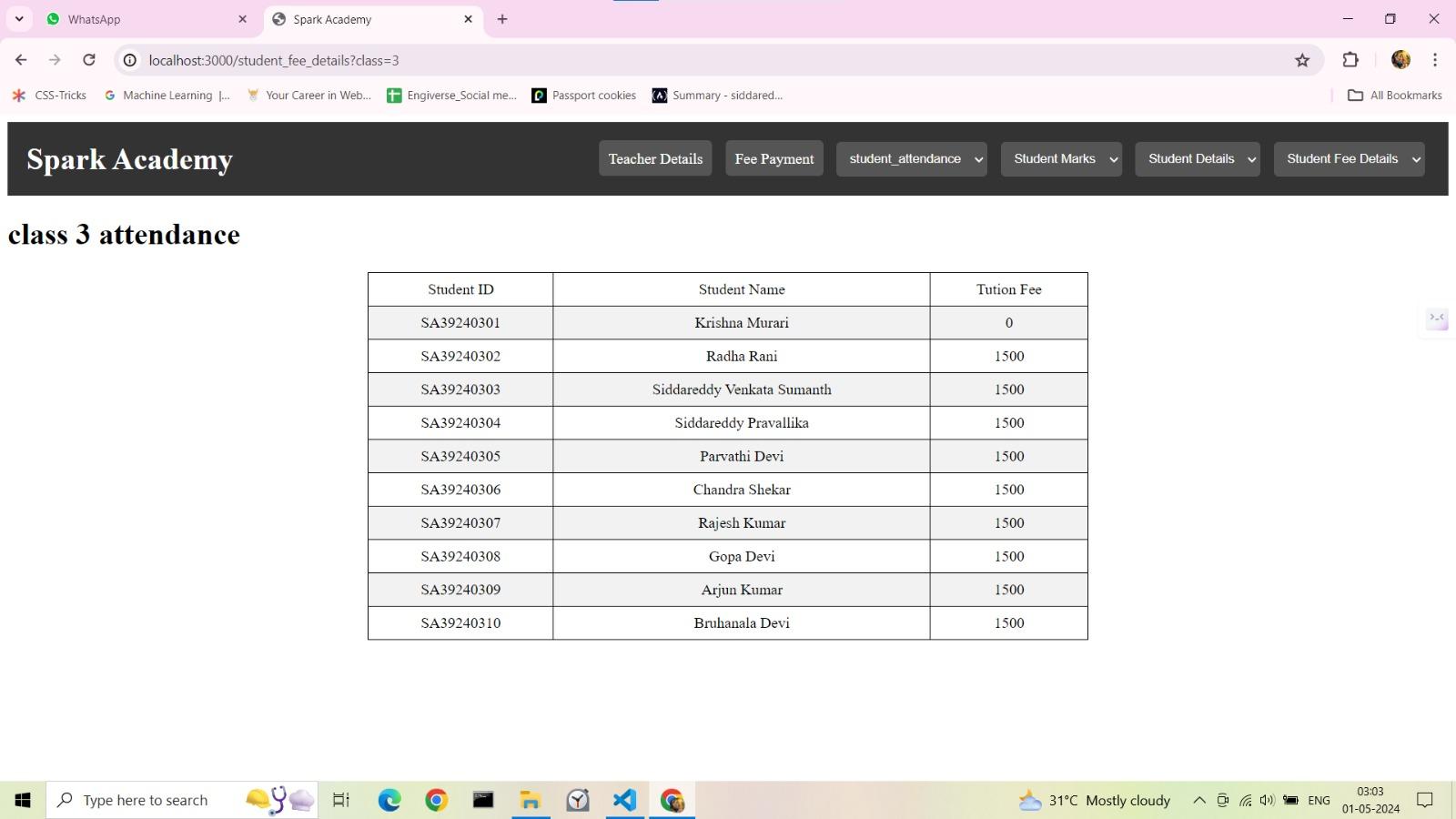
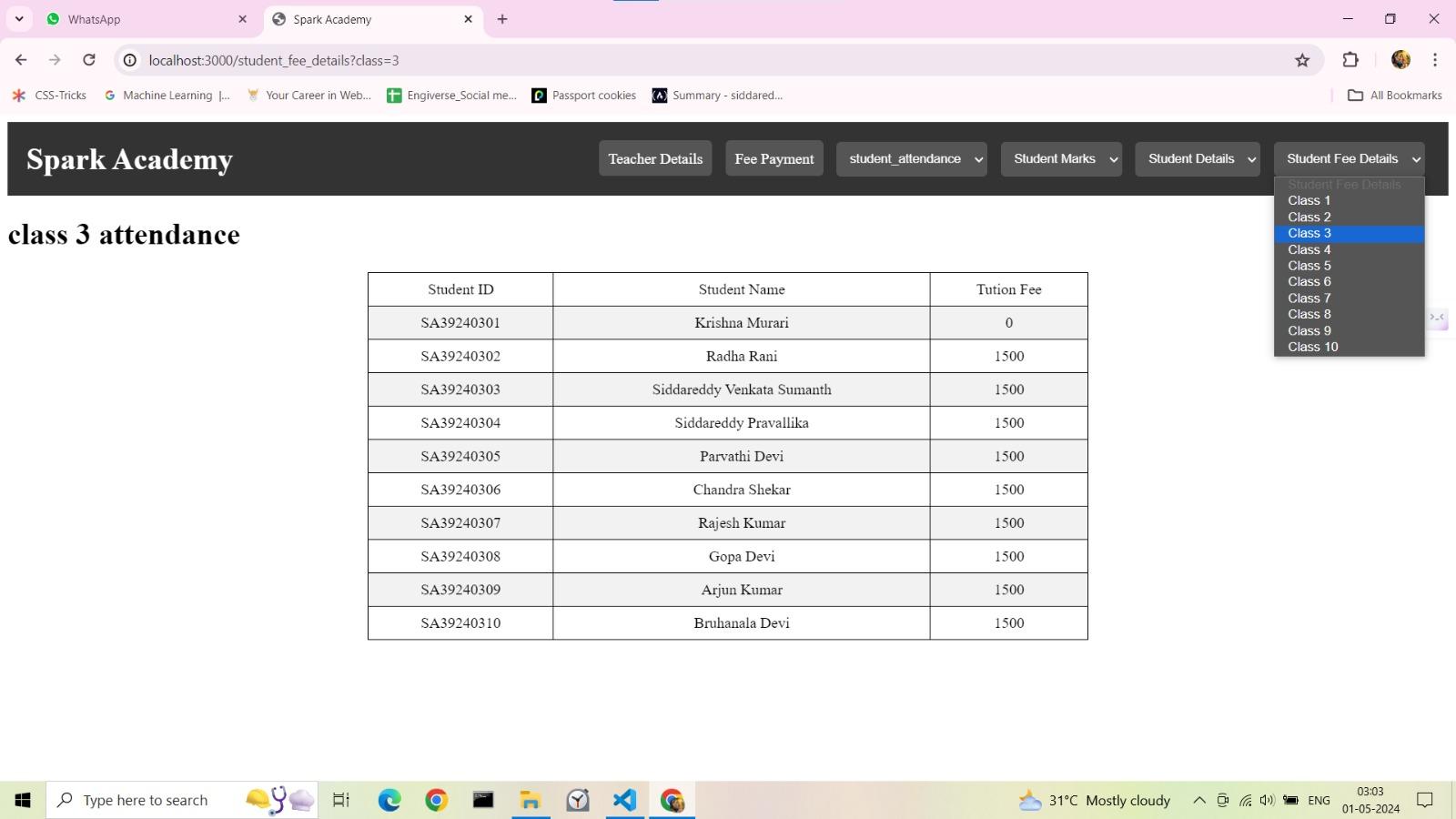
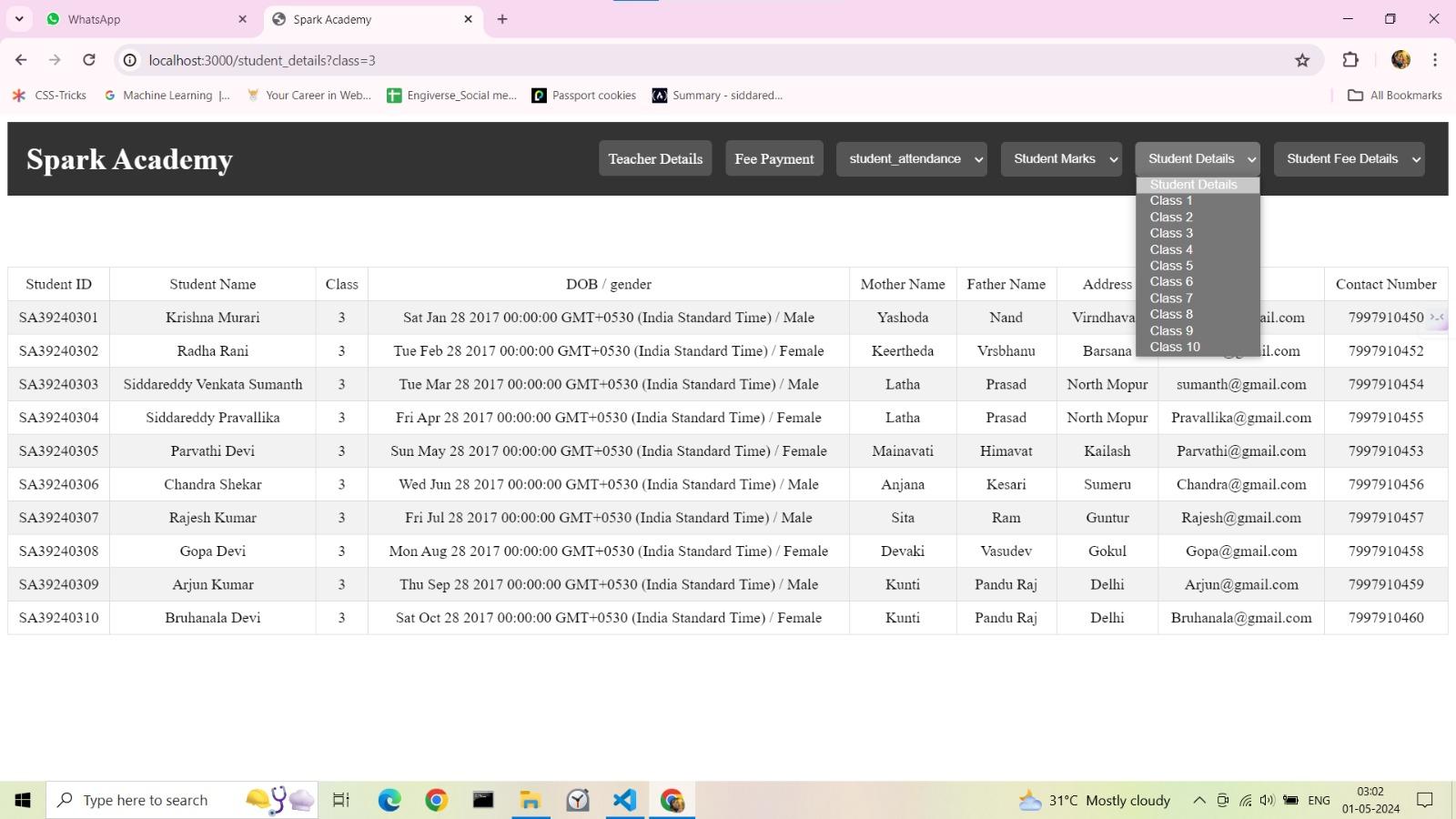
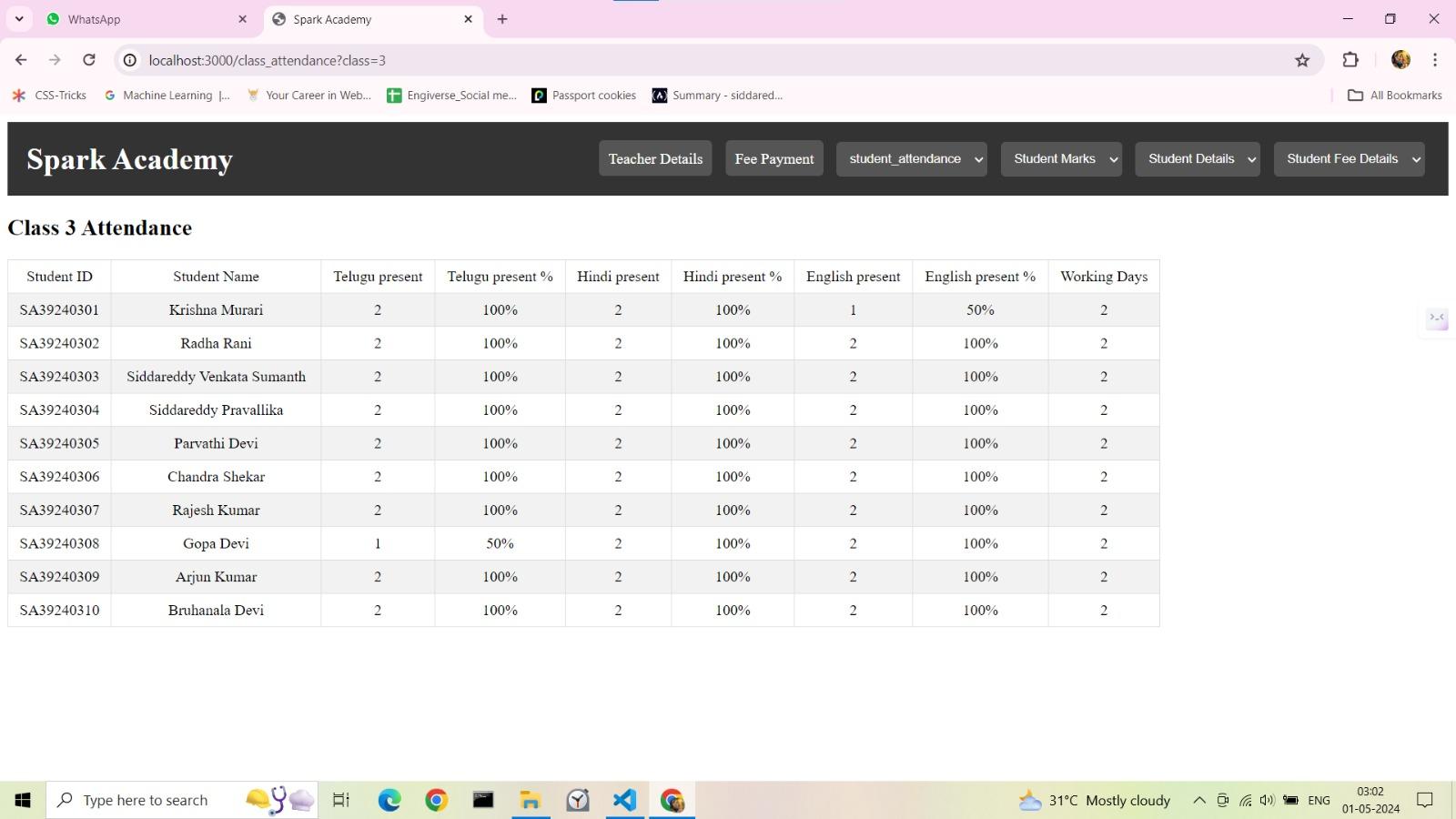
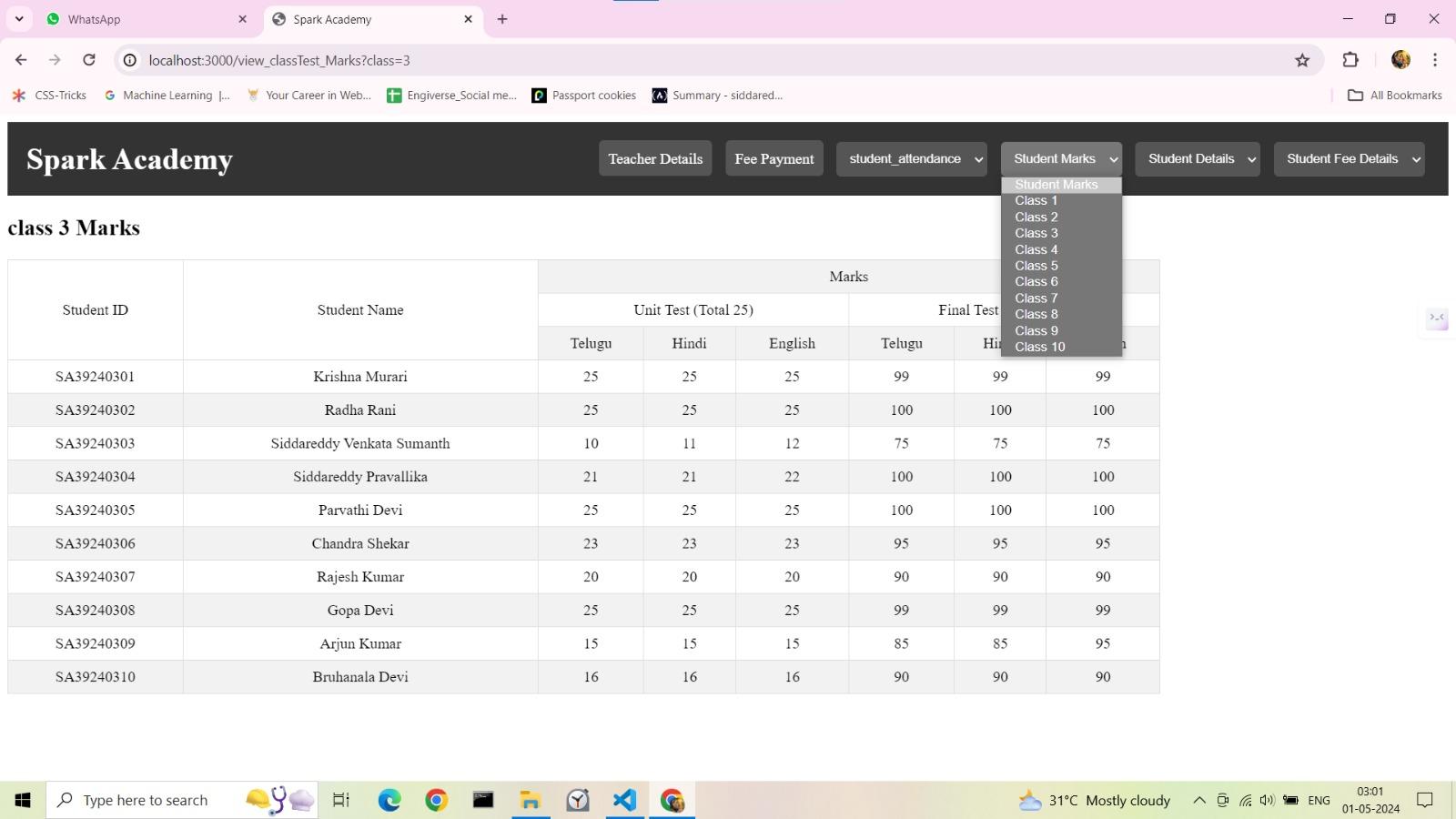
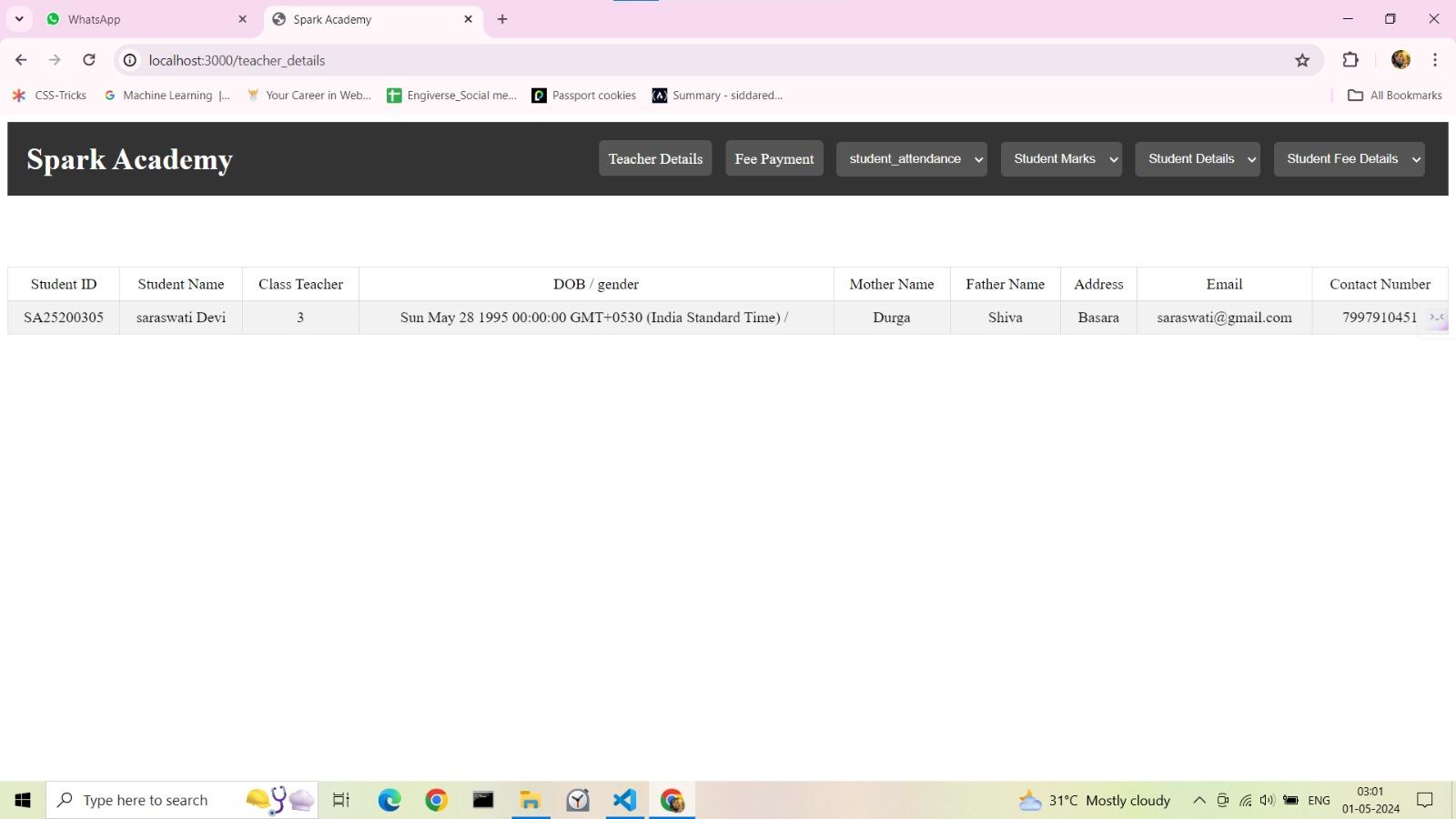
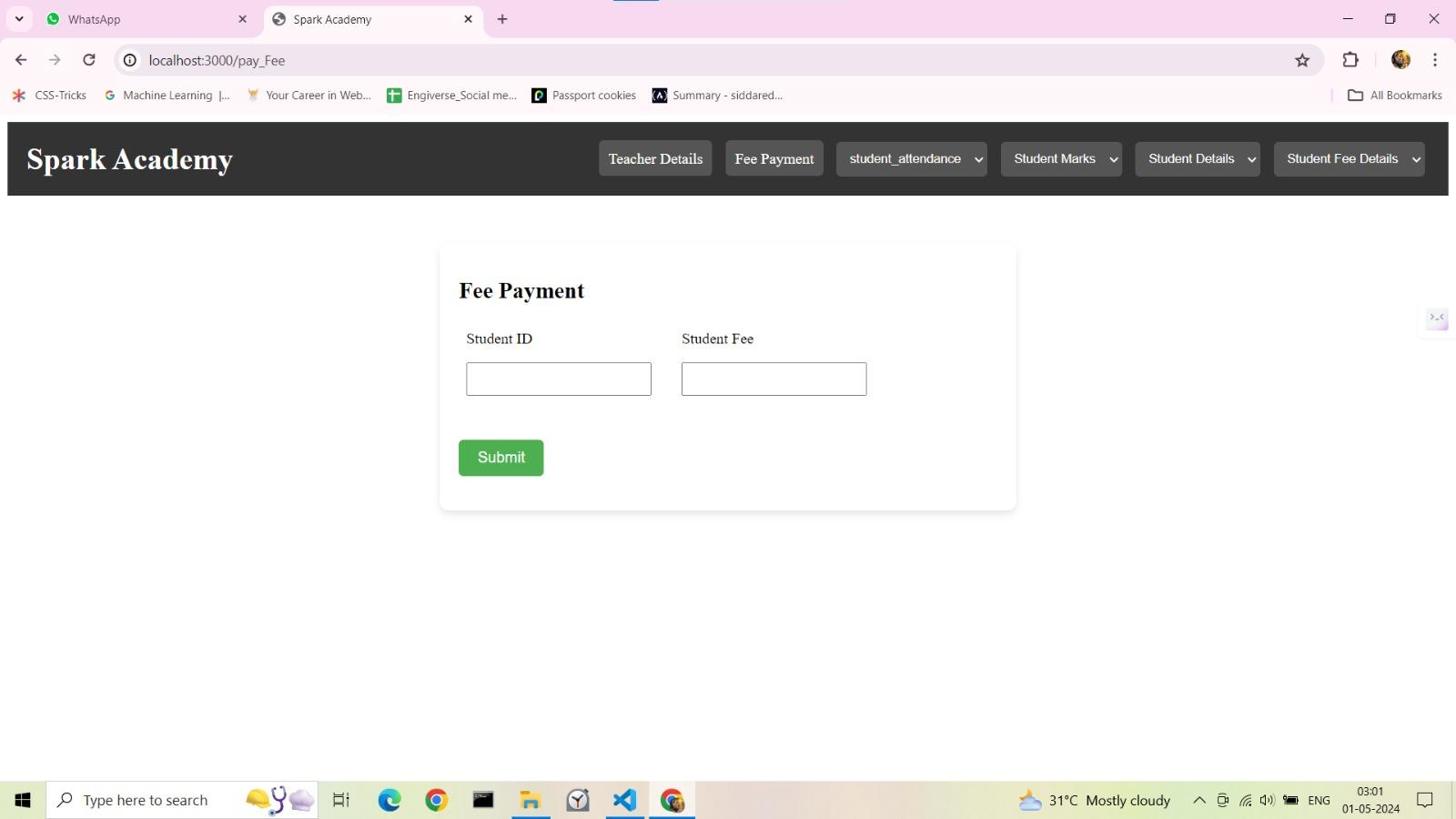
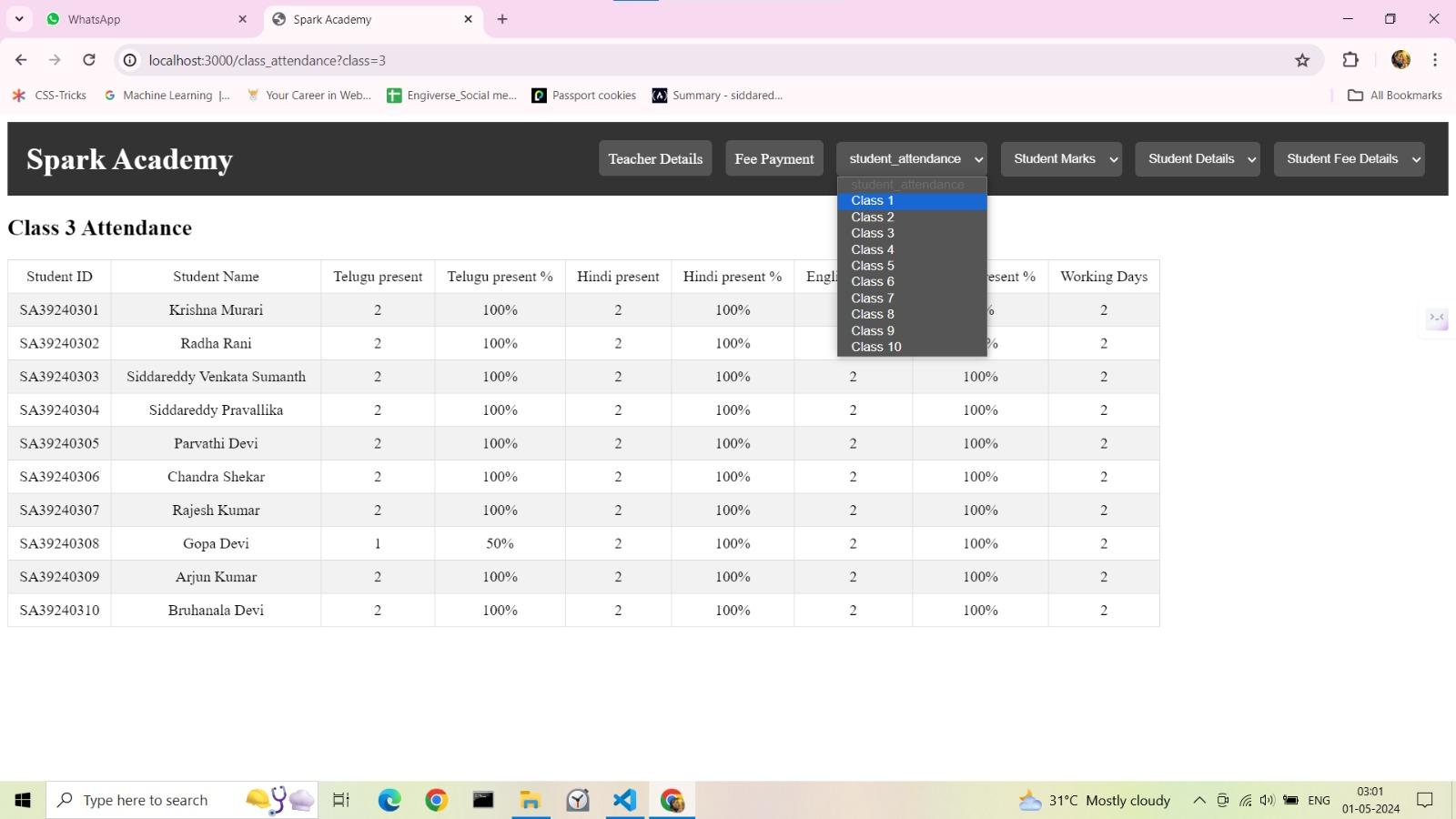
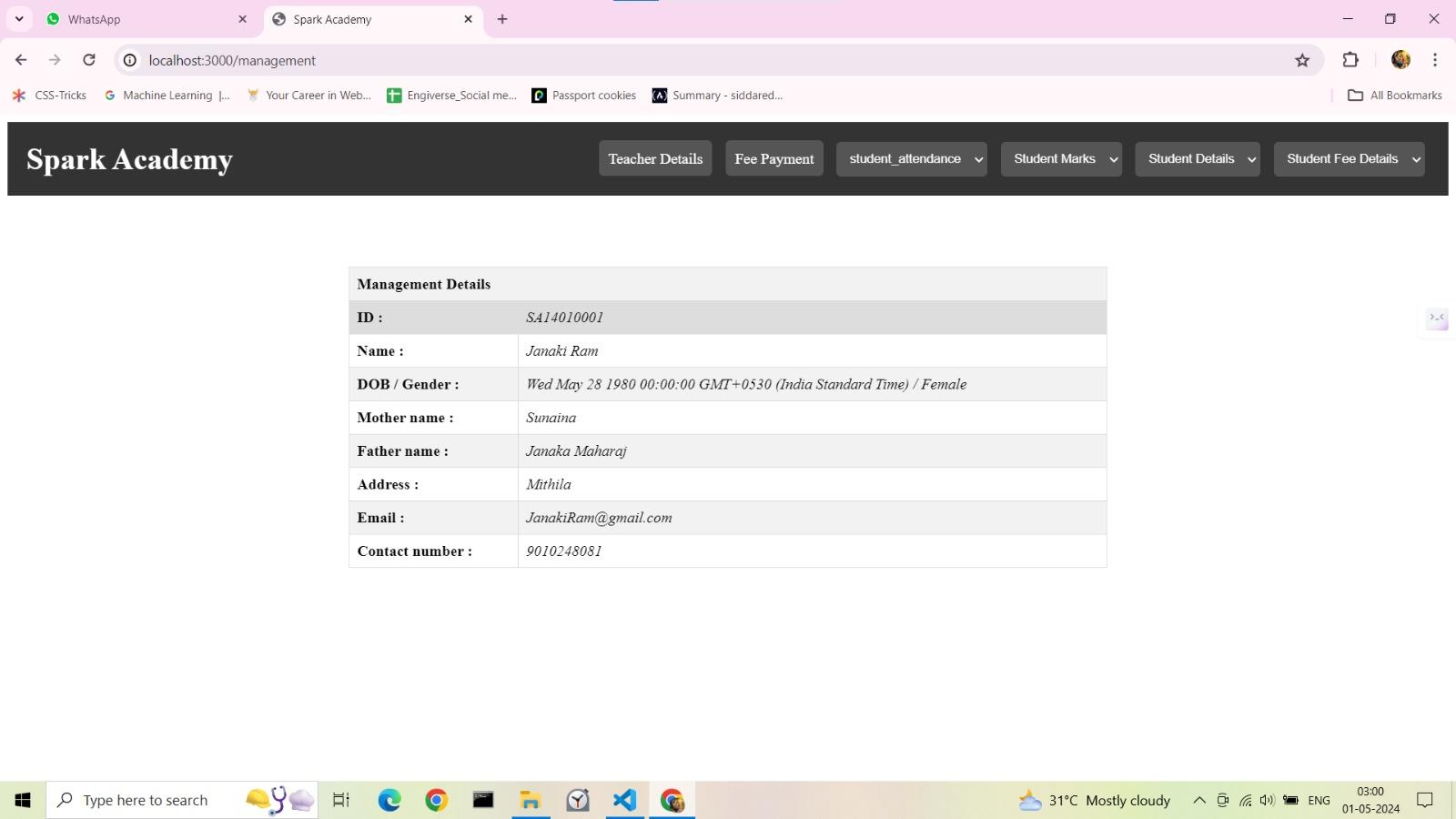
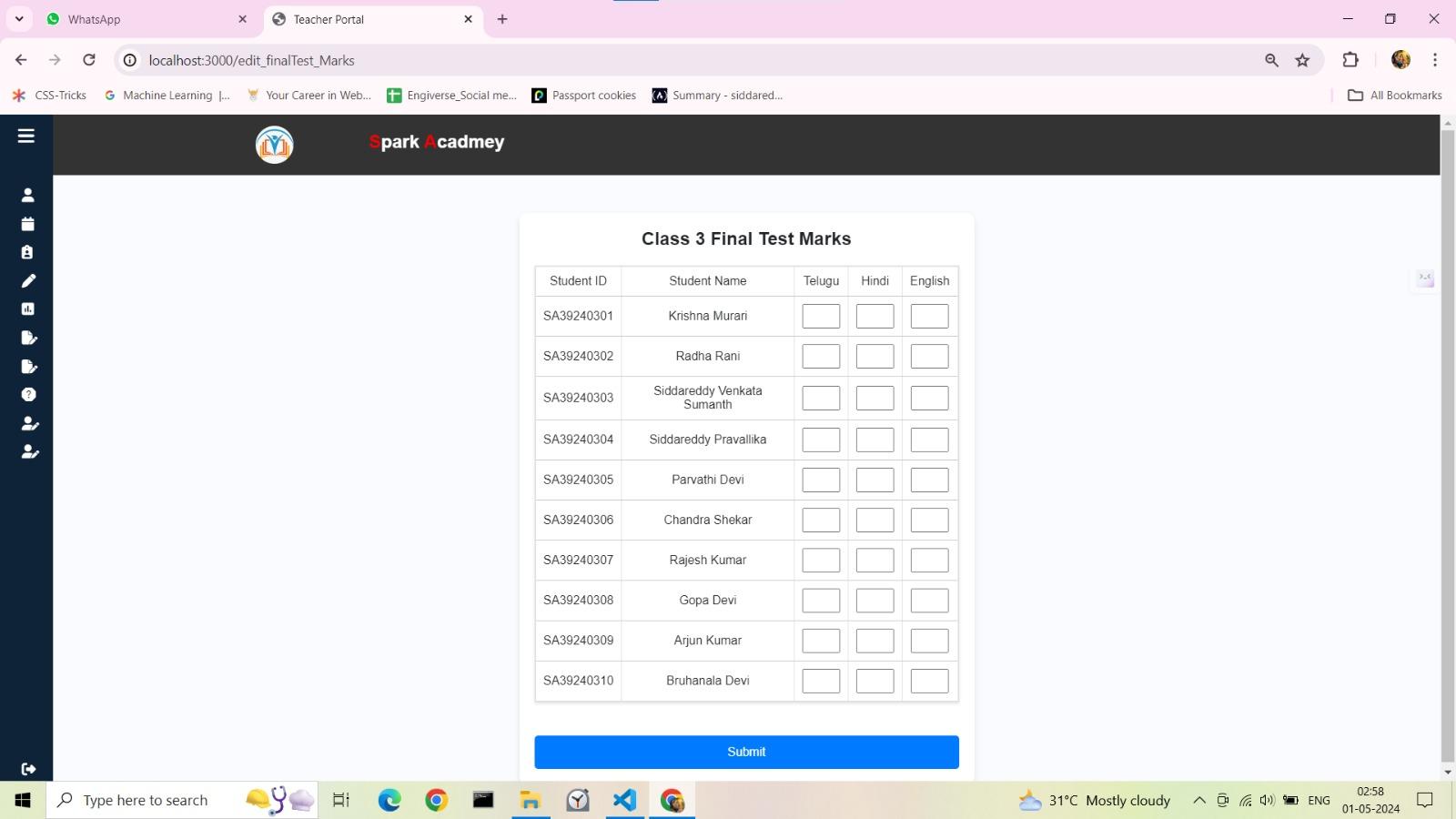
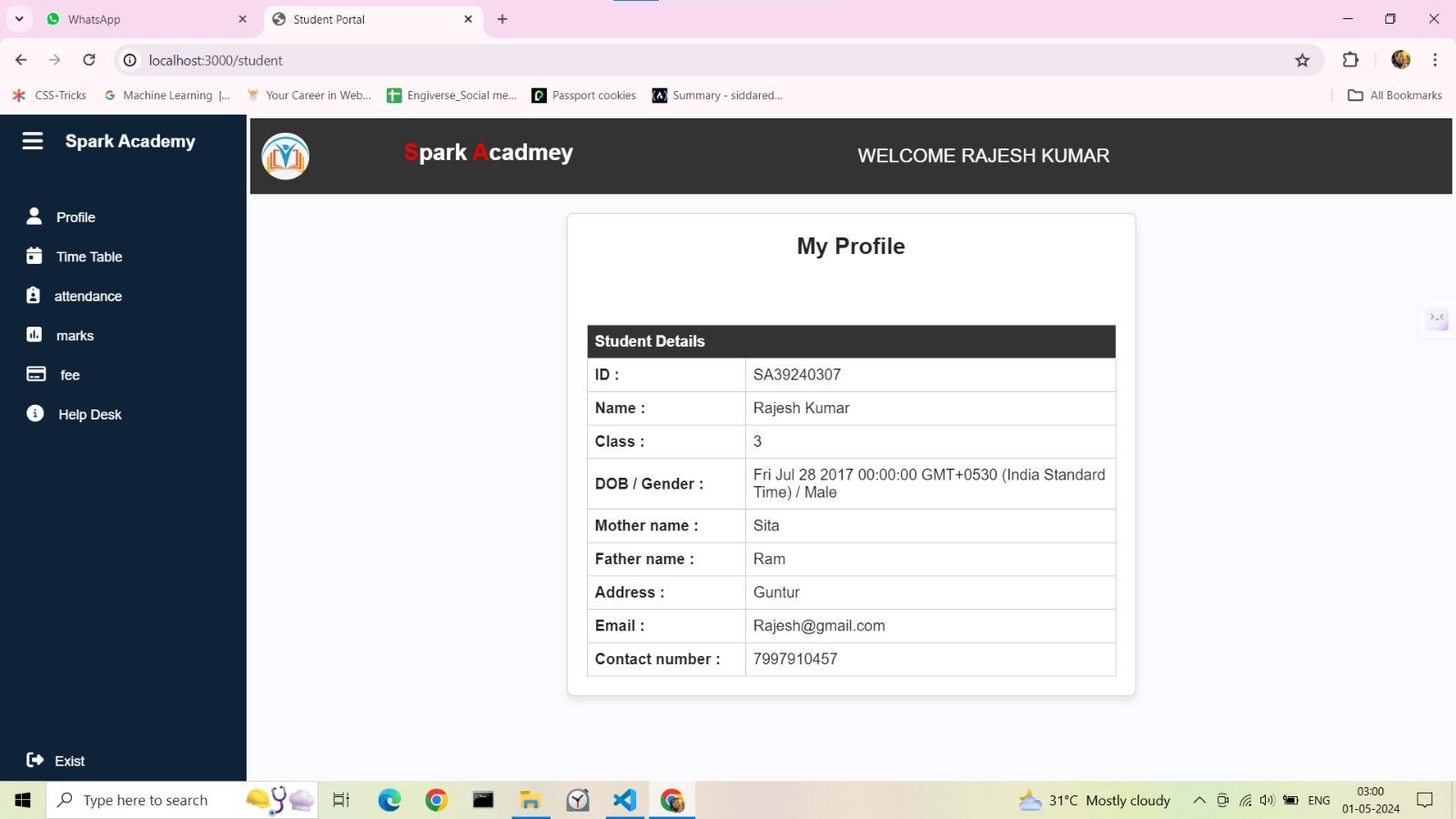
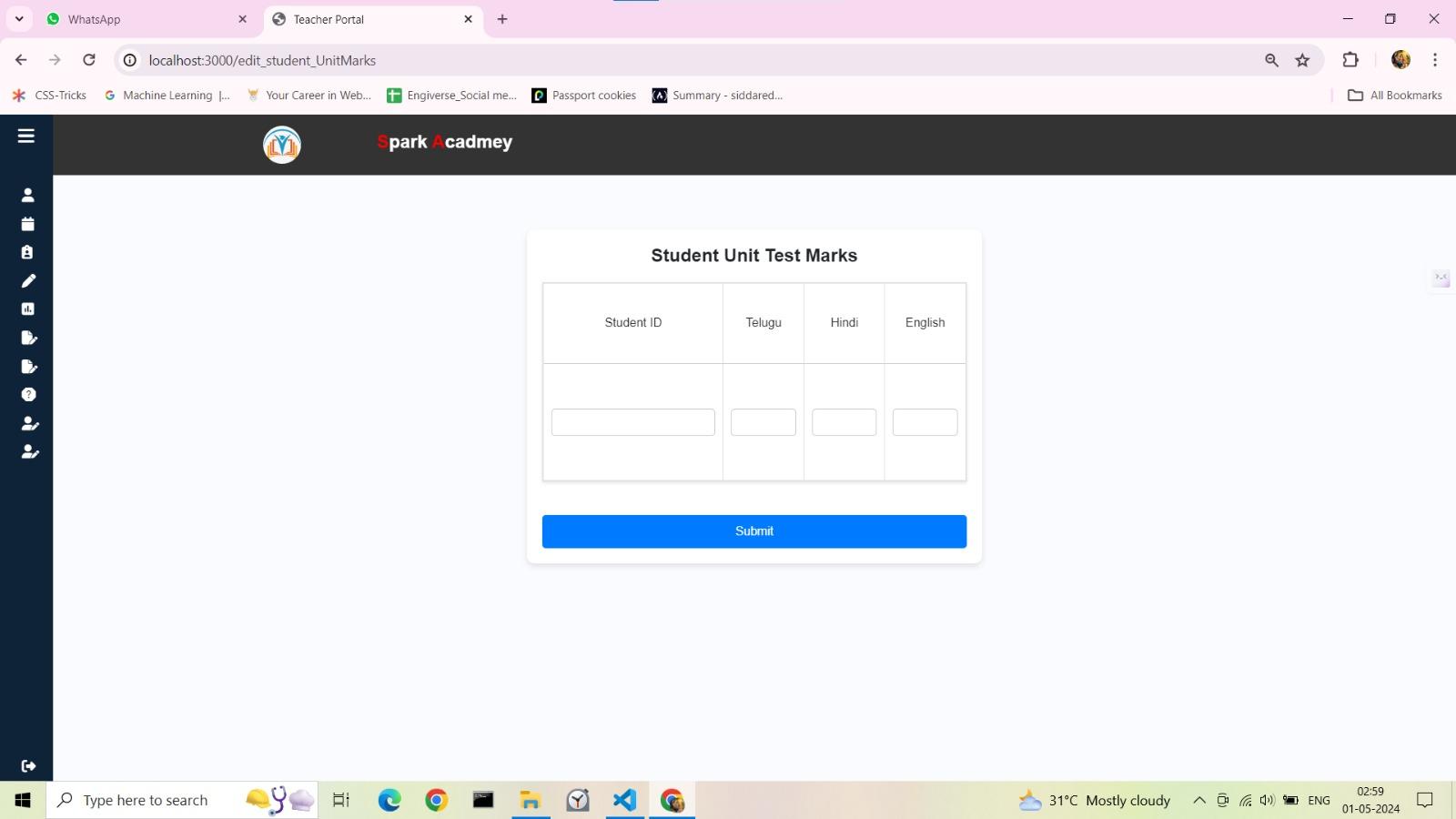
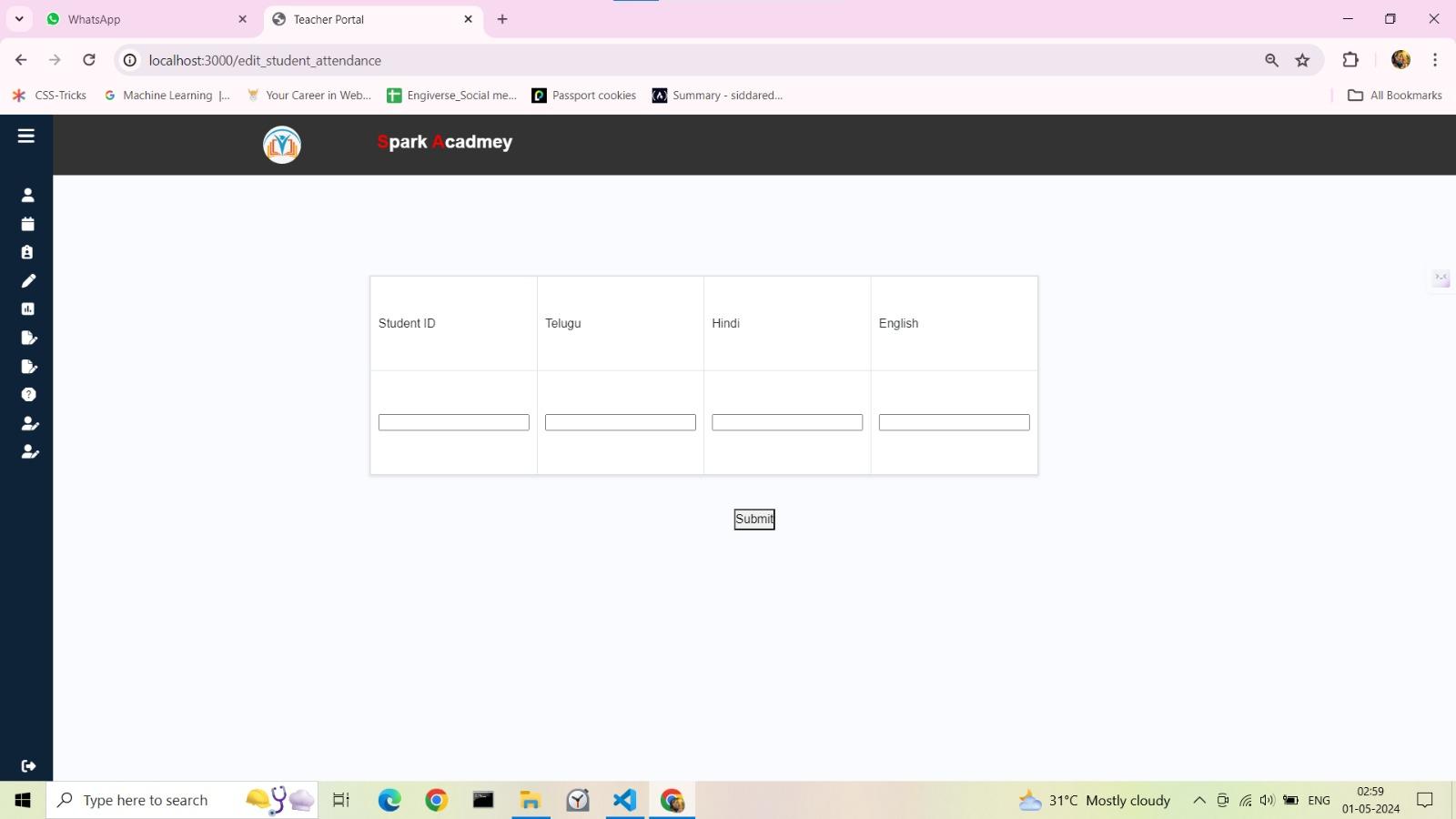
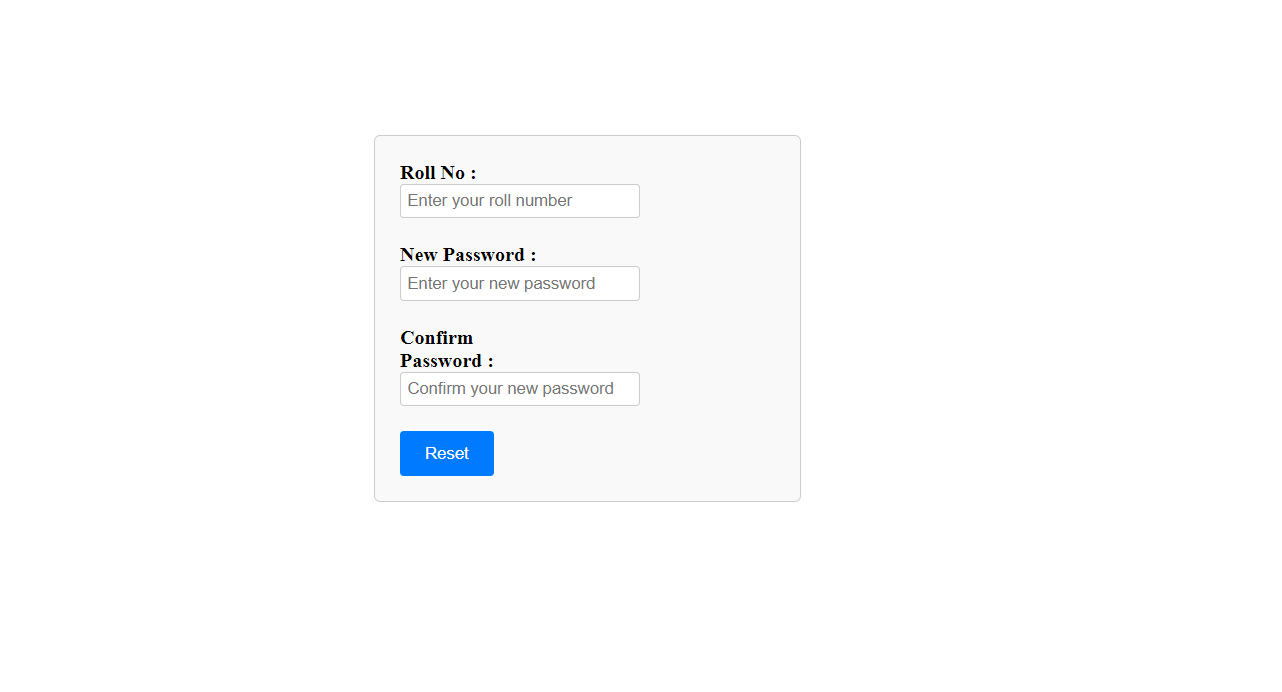
**Frontend:**

1. **HTML (Hypertext Markup Language):** HTML is the standard markup language for creating web pages. It provides the structure and content of web pages using elements such as <div>, <p>, <h1>, etc.
2. **EJS (Embedded JavaScript):** EJS is a templating language that allows embedding JavaScript code within HTML. It enables dynamic content generation on the server-side by executing JavaScript code and embedding the results directly into HTML templates.
3. **Bootstrap:** Bootstrap is a popular frontend framework for building responsive and mobile-first websites. It provides a set of pre-designed UI components (e.g., buttons, forms, navigation bars) and CSS stylesheets that help streamline the frontend development process. Bootstrap's grid system facilitates responsive layout design, ensuring that web pages look good on various devices and screen sizes.

**Backend:**

1. **Node.js:** Node.js is a server-side JavaScript runtime built on Chrome's V8 JavaScript engine. It allows developers to use JavaScript for building scalable and high-performance server-side applications. Node.js provides an event-driven, non-blocking I/O model, making it suitable for handling asynchronous operations such as file I/O, network requests, and database interactions.
2. **Express.js**: Express.js is a minimalist web application framework for Node.js. It simplifies the process of building web servers and APIs by providing features such as routing, middleware support, request handling, and response generation. Express.js abstracts away common HTTP server functionalities, allowing developers to focus on defining routes and implementing business logic.
3. **pg (node-postgres):** pg is a Node.js module that serves as a PostgreSQL client library. It enables communication between Node.js applications and PostgreSQL databases by providing methods for executing SQL queries, managing database connections, and handling result sets. pg abstracts the underlying database interactions, making it easier for developers to work with PostgreSQL databases in Node.js applications.
4. **Bcrypt:** Bcrypt is a cryptographic hashing library used for securely hashing passwords in Node.js applications. It employs a one-way hashing algorithm (bcrypt) to convert plaintext passwords into irreversible hash values. This ensures that even if the hashed passwords are compromised, they cannot be easily reversed or decrypted. Bcrypt enhances security by protecting user passwords from brute-force attacks and unauthorized access.
5. **Session Management:** Express.js Session Middleware: Express.js session middleware is used for managing user sessions in Node.js applications. It enables the creation and management of session objects that store user-specific data across multiple HTTP requests. Session management is essential for maintaining user authentication state, tracking user sessions, and storing session-related data such as user IDs, authentication tokens, and user preferences.
6. **Dotenv:** Dotenv is a Node.js module used for loading environment variables from a .env file into the Node.js process environment. It allows developers to store sensitive information (e.g., database credentials, API keys) as environment variables, rather than hardcoding them directly into the codebase. By using dotenv, developers can keep sensitive information secure and separate from the application code, reducing the risk of accidental exposure or leakage.
7. **Execution Screenshots**

****

****

1. **Data flow diagrams**

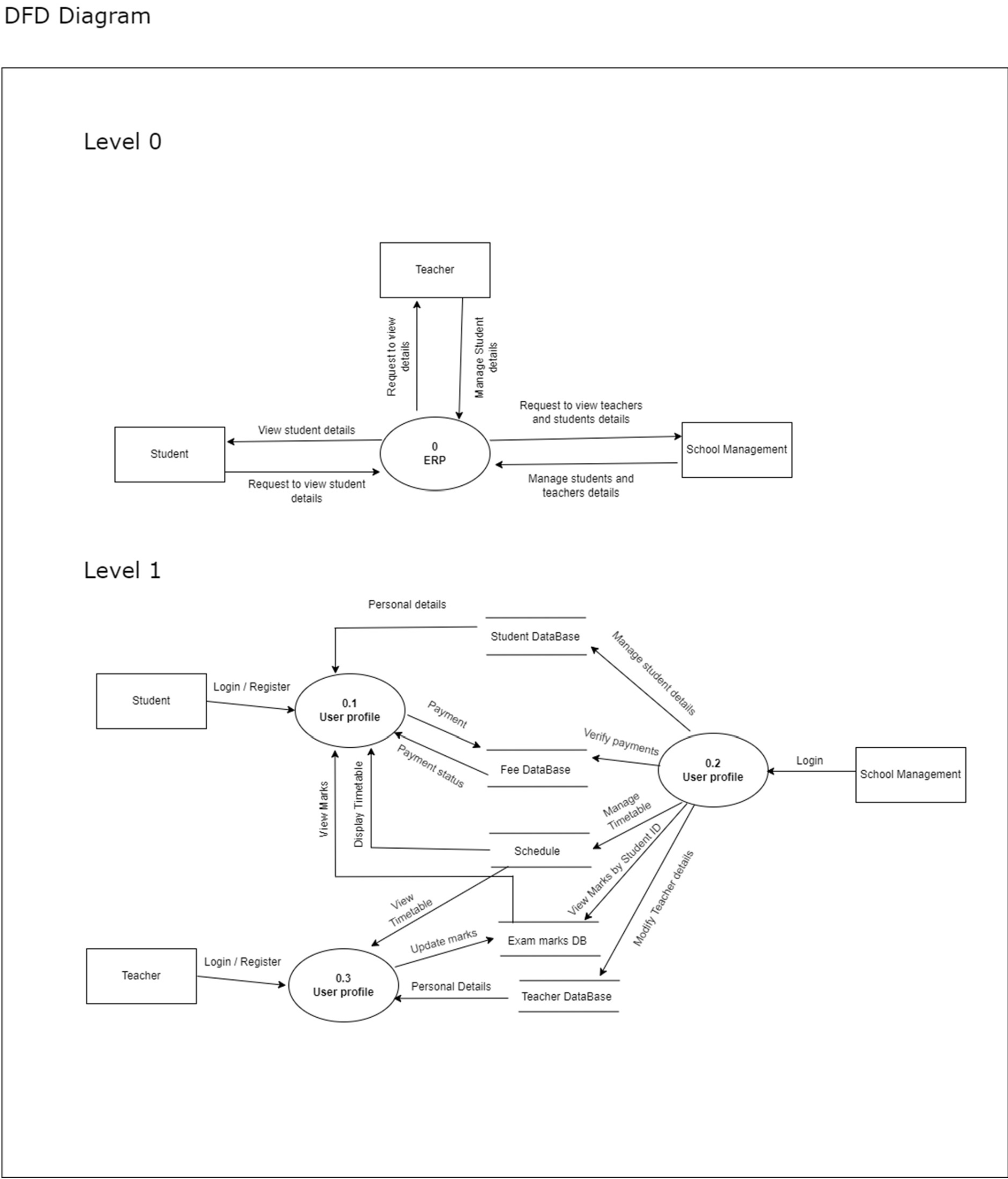
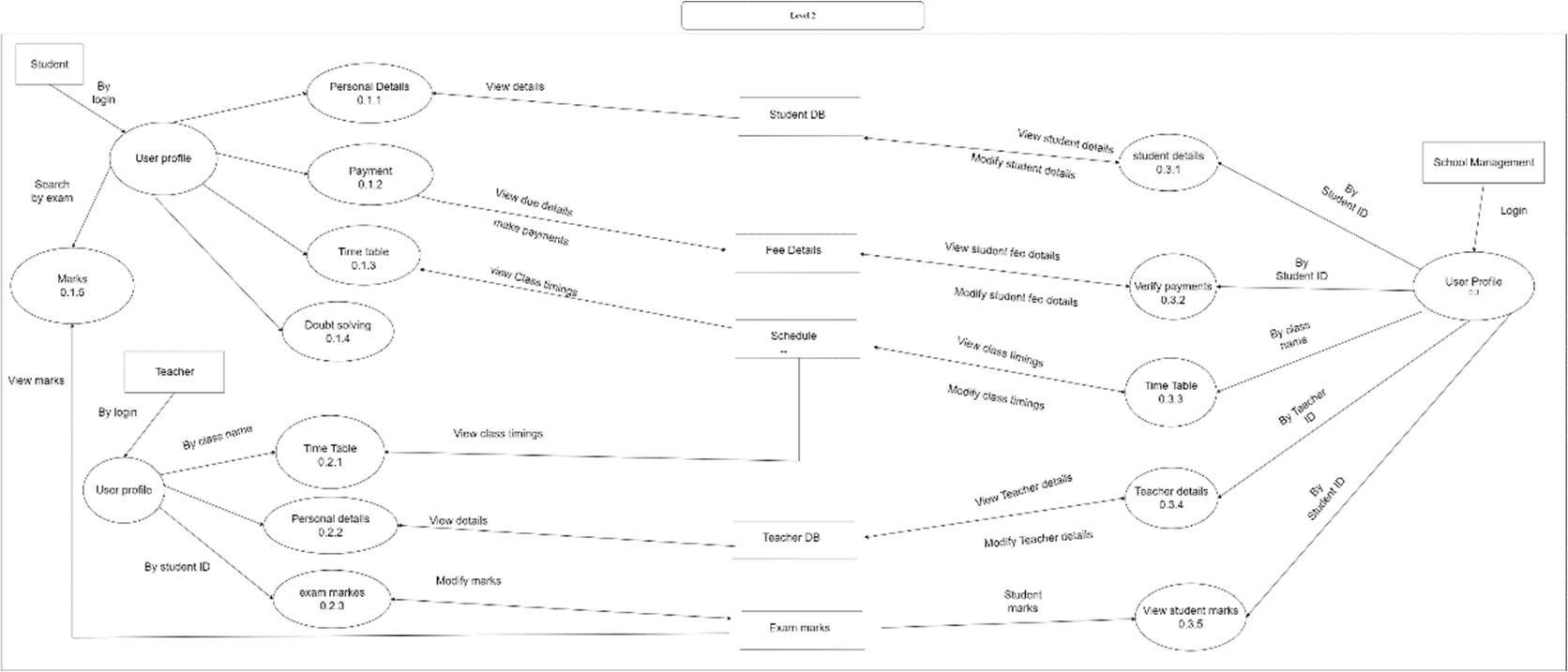


fig 1. Level 0, level 1 of Data flow Diagram for ERP



## fig 2. Level 2 Data Flow Diagram for ERP

## Level 0 - Context Diagram

The level 0 DFD provides a high-level overview of the entire system. It depicts the system as a single process entity with external entities that interact with it.

* **External Entity:** Teacher, Student, School Management
* **Process:** ERP (Enterprise Resource Planning System)
* **Data Flows**:

1. Teacher -> ERP: Request to view teachers and students details
2. Student -> ERP: Request to view student details
3. ERP -> School Management: Manage students and teachers details

## Level 1 DFD

The level 1 DFD provides more detail about the internal workings of the ERP system. It breaks down the single process in the level 0 DFD into smaller subprocesses.

* **Processes:**

1. Student Profile (0.1)
2. Teacher profile (0.2)
3. School management (0.3)

## Data Flows:

**Student Login/Register:** Students can log in or register for the system. This likely sends user profile data to the student database. (0.1)

1. **Request to view student details:** The school management system can request to view student details from the student database.
2. **Manage student details:** The school management system can manage student details, which likely involves updating the student database.
3. **Request to view teacher details:** The school management system can request to view teacher details from the teacher database.
4. **Manage teacher details:** The school management system can manage teacher details, which likely involves updating the teacher database.
5. **Teacher Login/Register:** Teachers can log in or register for the system. This likely sends user profile data to the teacher database. (0.3)
6. **View Marks by Student ID:** The system can view marks for a specific student, likely by retrieving data from the exam marks database.
7. **Update marks:** The system can update marks, likely sending data to the exam marks database.
8. **View Timetable:** The system can view the timetable, likely retrieving data from the schedule database.
9. **Manage Timetable:** The school management system can manage the timetable, likely updating the schedule database.
10. **Payment:**Admin can manually allow the students to pay the fee.
11. **Verify payments:** The school management system can verify payments, likely referencing data in the fee database.

## Data Stores:

* **Student Database:** Stores student information.
* **Teacher Database:** Stores teacher information.
* **Fee Database:** Stores fee information.
* **Exam Marks Database:** Stores exam marks.
* **Schedule Database:** Stores the class schedule.

## Level 2 DFD

**Processes**

* Student

1. Personal details (0.1.1)
2. Time table (0.1.3)
3. Doubt solving (0.1.4)
4. Marks (0.1.5)

* Teacher

1. Personal details (0.2.1)
2. Time table (0.2.2)
3. Exam marks(0.2.3)

* School management

1. Student details (0.3.1)
2. Offline fee payment(0.3.2
3. Verify payments (0.3.3)
4. Time table (0.3.4)
5. Teacher details (0.3.5)
6. View student marks (0.3.6)

**9. Data Dictionary**

| **Field name** | **Data Type** | **Description** |
| --- | --- | --- |
| Student ID | String (AlphaNumeric) | A unique number for every student |
| Teacher ID | String (AlphaNumeric) | A unique number for every teacher |
| Management ID | String (AlphaNumeric) | A unique number for management |
| DOB | date | Date of birth of user |
| Gender | String | Gender of the user |
| FirstName | String | First name of user |
| Middle\_LastName | String | Middle name and last name of user |
| Class number | Integer | Class number for students and for teachers it shows their respective classes |
| Father Name | String | Father name of user |
| Mother Name | String | Mother name of user |
| Address | String (AlphaNumeric) | House address of user |
| Email | String (AlphaNumeric) | Email ID of user |
| Telephone number | String | Contact number of user |
| Password | String (encrypted Hash) | Password to user account |
| Tuition Fee | Real | Tuition fee of students |
| Exam Marks | Integer | student unit test and final test marks in range [0, 100] |
| Attendance | Integer | student attendance |
| Working days | Integer | number of total working days |
| Telugu | Integer | Student attendance for telugu subject |
| Hindi | Integer | Student attendance for hindi subject |
| English | Integer | Student attendance for english subject |

Table 1. Data diagram for ERP

**10. Use Case Diagram**

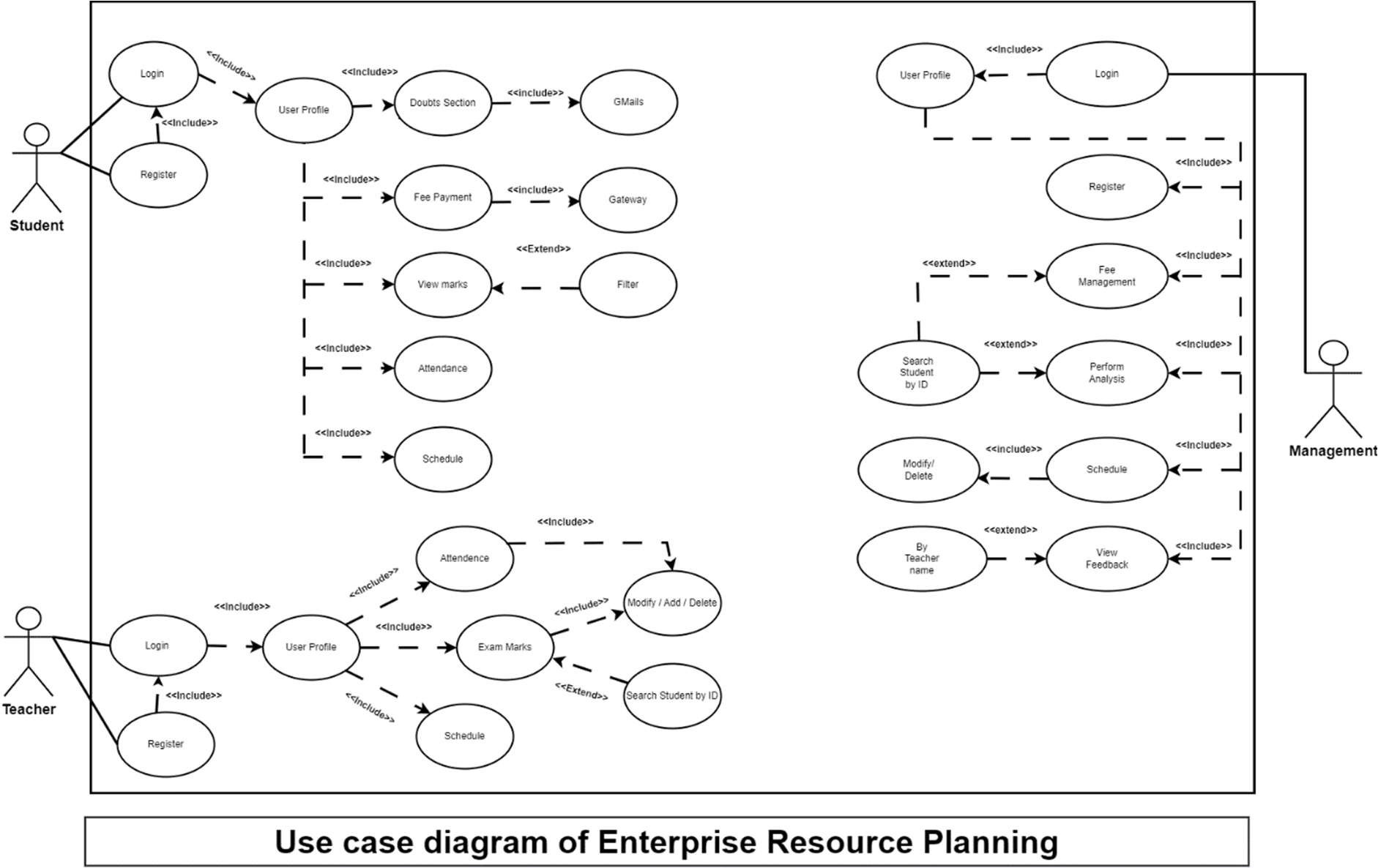
****

fig 3. Use case Diagram for ERP

**11. ER diagram**

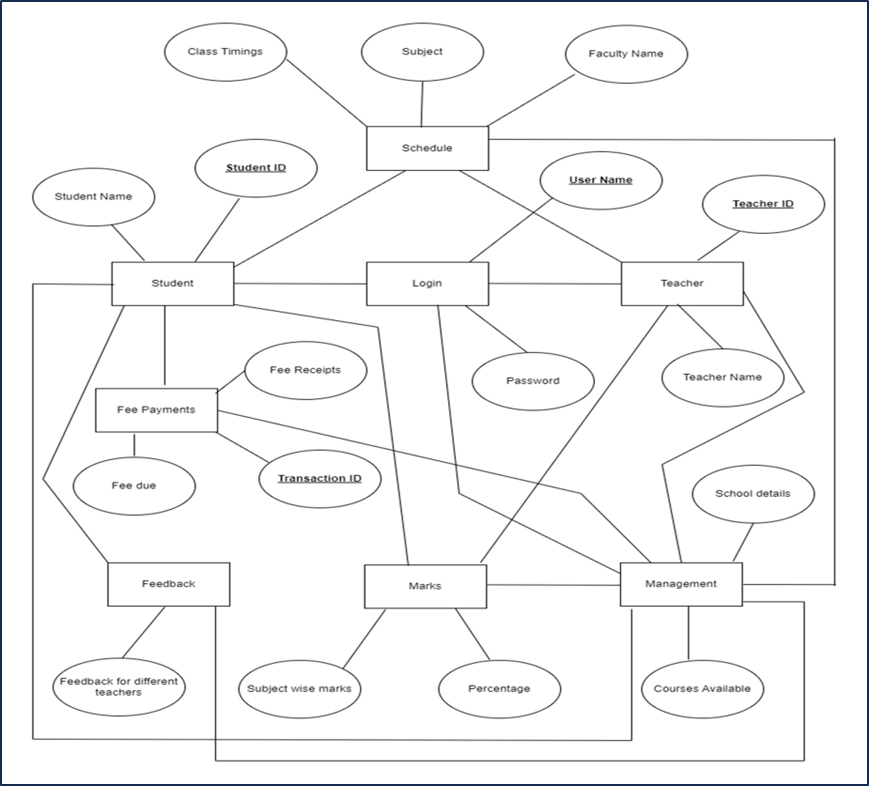


fig 4. ER Diagram for ERP

# Student

1. **Register:** This use case allows students to register for the system. Student has an attribute Student ID as primary key through which Student details can be accessed.
2. **Fee Due:** This use case allows students to check their fee due.
3. **View Marks:** This use case allows students to view their marks.
4. **View Attendance:** This use case allows students to view their attendance.
5. **Schedule:** This use case allows students to view their class schedule.

# 2. Teacher

1. **Login:** This use case allows teachers to login to the system. Teacher has an attribute Teacher ID as primary key through which Student details can be accessed.
2. **Register:** This use case allows teachers to register for the system.
3. **View Marks:** This use case allows teachers to view student marks.
4. **View Attendance**: This use case allows teachers to view student attendance.
5. **Modify/Add/Attendance**: This use case allows teachers to modify, add, or delete student attendance records.
6. **Schedule**: This use case allows teachers to view their class schedule.

# 3. Management

1. Management **can access** both student and Teacher details.
2. **Can update** all the attributes of teacher and student.
3. **Offline fee payment:** They can make changes in fee Due, which reflects the students portal and database.

**12. Class Diagram**

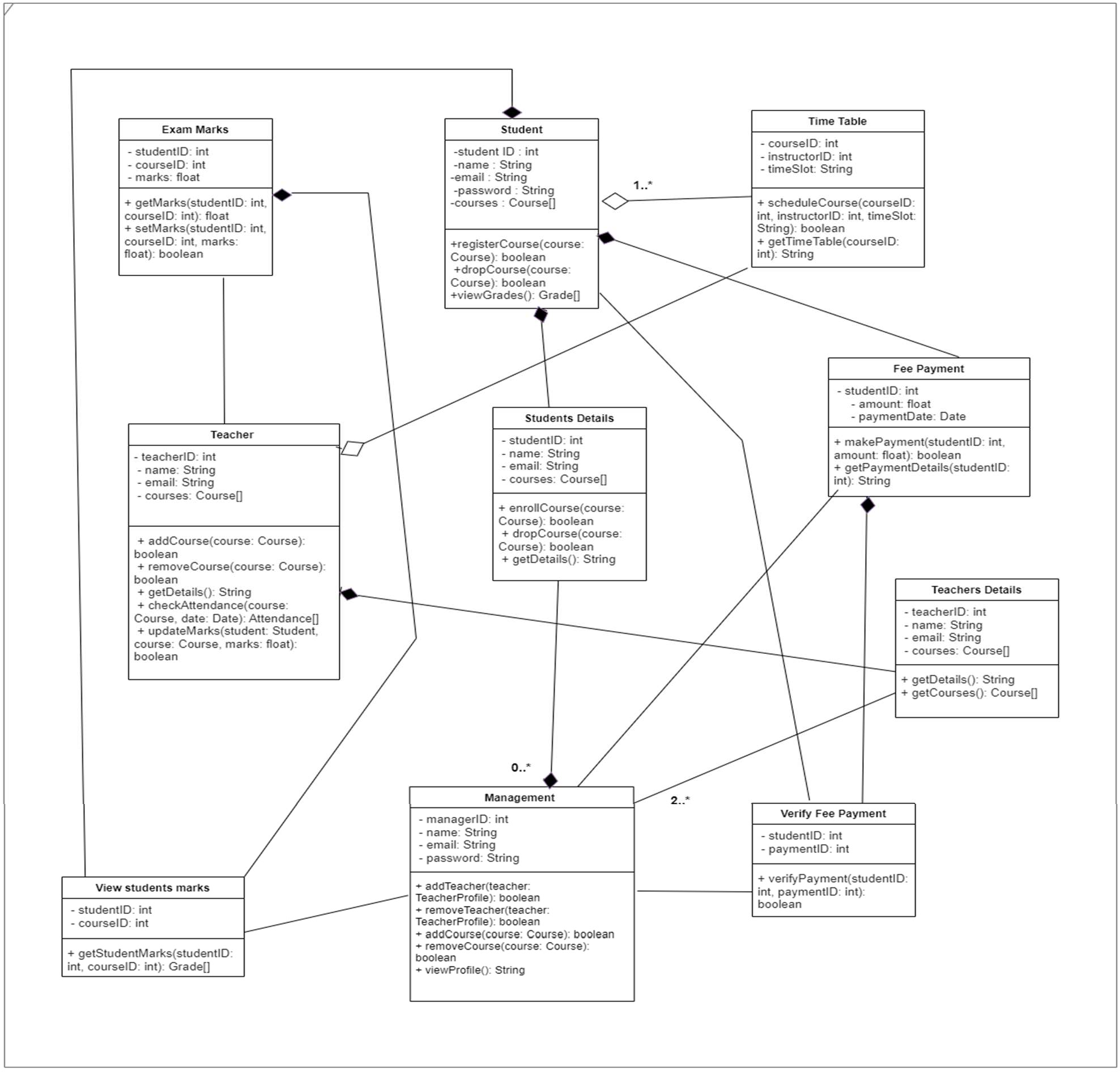


fig 5. Class Diagram for ERP

This class diagram depicts a possible class structure for the ERP system. It shows the various entities (classes) involved in the system and how they relate to each other.

# Student

**Attributes:**

1. studentID: int
2. courseID: int
3. marks: float
4. name: String
5. email: String
6. password: String
7. courses: Course[ ] (an array of Course objects)

**Methods:**

1. getMarks(studentID: int, courseID: int): float - This method likely retrieves the marks obtained by a student in a particular course based on their student ID and course ID.
2. setMarks(studentID: int, courseID: int, marks: float): boolean - This method likely allows updating a student’s marks in a particular course.
3. registerCourse(course: Course): boolean - This method allows a student to register for a course.
4. dropCourse(course: Course): boolean - This method allows a student to drop a course. viewGrades(): Grade - This method likely returns a student’s grades.

# Teacher

**Attributes:**

1. teacherID: int
2. name: String
3. email: String
4. courses: Course[ ] (an array of Course objects)

**Methods:**

1. enrollCourse(course: Course): boolean - This method allows a teacher to enroll for a course (likely to manage the courses they teach).
2. dropCourse(course: Course): boolean - This method allows a teacher to drop a course (likely to manage the courses they teach).
3. addCourse(course: Course): boolean - This method might be for adding a course to the system (possibly by an admin).
4. removeCourse(course: Course): boolean - This method might be for removing a course from the system (possibly by an admin).
5. getDetails(): String - This method likely returns details about the teacher.
6. checkAttendance(course: Course, date: Date): Attendance[] - This method allows a teacher to check the attendance of students in a course on a particular date.
7. updateMarks(student: Student, course: Course, marks: float): boolean - This method allows a teacher to update a student’s marks in a course.

# Management

**Attributes:**

1. managerID: int
2. name: String
3. email: String
4. password: String

**Methods:**

1. View students marks(studentID: int, courseID: int): Grade[] - This method retrieves the marks obtained by a student in a particular course based on their student ID and course ID (likely for the admin).
2. AddTeacher(teacher: Teacher Profile): boolean - This method allows adding a teacher to the system (likely by the admin).
3. RemoveTeacher(teacher: Teacher Profile): boolean - This method allows removing a teacher from the system (likely by the admin).
4. AddCourse(course: Course): boolean - This method might be for adding a course to the system (possibly by an admin).
5. RemoveCourse(course: Course): boolean - This method might be for removing a course from the system (possibly by an admin).
6. VerifyPayment(studentID: int, paymentID: int): boolean - This method likely verifies if a student’s payment has been made based on their student ID and payment ID.

**13. Activity Diagram**

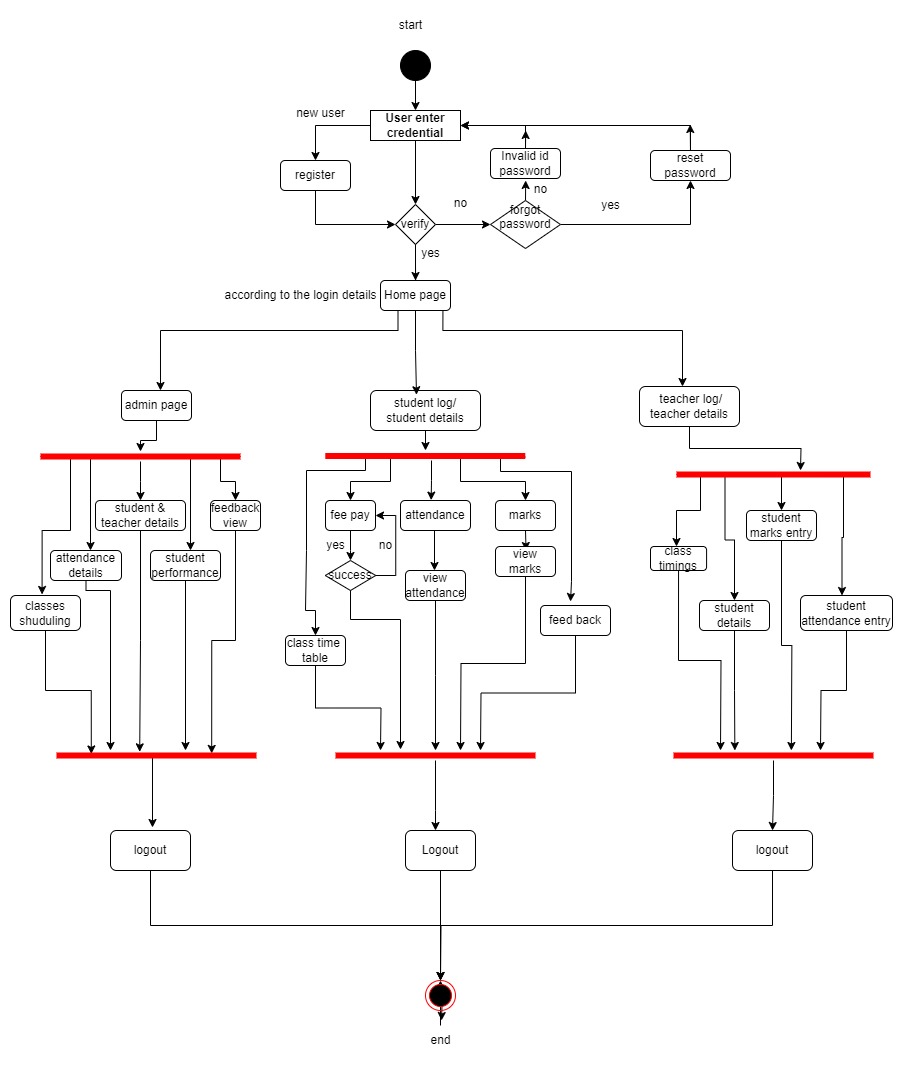
****

fig 6. Activity diagram for ERP

**User Registration**

1. The process starts with a new user entering their credentials.
2. The system verifies the credentials.
3. If the credentials are invalid, the user is prompted to reset their password.
4. If the credentials are valid but the user forgot their password, they can choose to reset it.
5. Upon successful verification, the user is directed to the homepage.

**User Types and Access**

1. The homepage provides access to different sections depending on the user type (student, teacher, admin).
2. Students and teachers can view their details and feedback.
3. Students can also view their attendance and class schedule.
4. Teachers can view attendance, marks, and student details.
5. Admins can access additional functionalities like fee payment and class schedule management.

**Additional Activities**

1. Users can view marks and attendance.
2. Students can enter feedback.
3. The system allows logging out, which ends the session.

**14. State Chart Diagram**

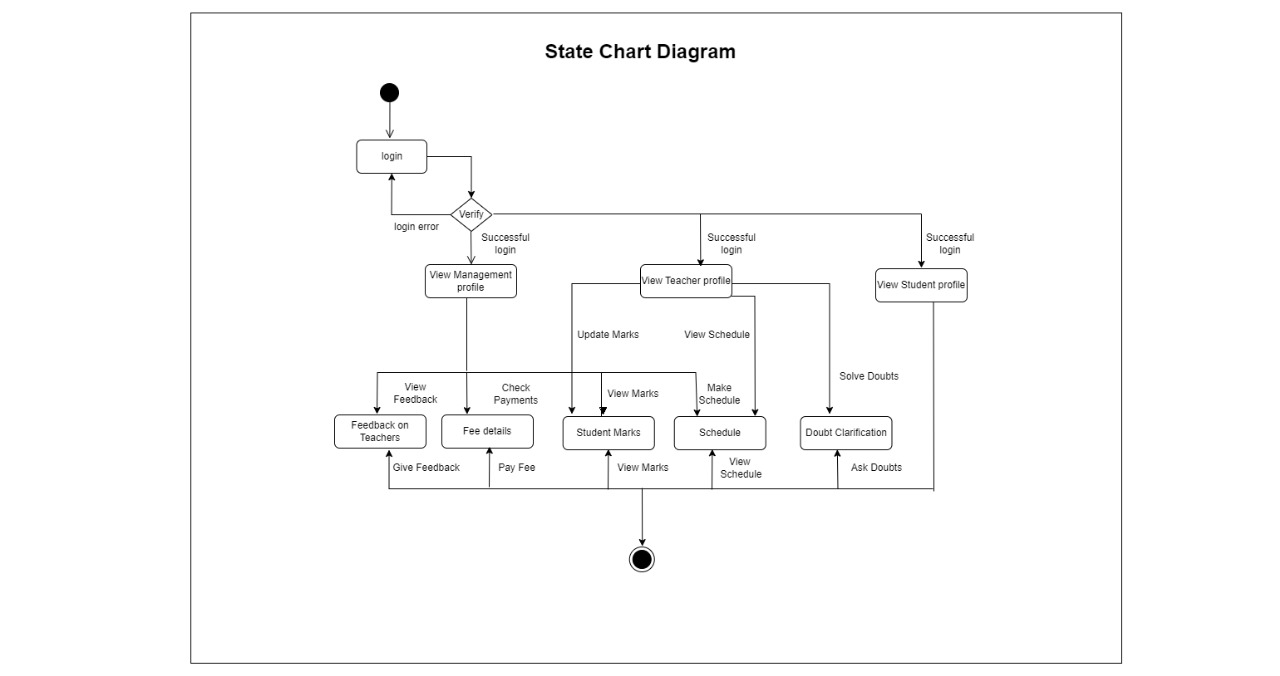
****

fig 7. State chart diagram for ERP

**States**

1. **Login**: This is the initial state where the user enters their credentials.
2. **Login Error:** This state is reached if the user enters invalid credentials.
3. **Successful Login**: This state is reached if the user enters valid credentials.
4. **View Management Profile:** This state allows admins to view their profile information.
5. **View Teacher Profile**: This state allows teachers to view their profile information.
6. **View Student Profile:** This state allows students to view their profile information.
7. **Update Marks**: This state allows teachers to update student marks.
8. **View Schedule**: This state allows users to view the class schedule.
9. **View Marks**: This state allows students to view their marks.
10. **Make Payment:** Students can make their fee in Offline mode.
11. **Solve Doubts:** This state allows students to ask questions or clarify doubts with teachers.
12. **Give Feedback**: This state allows users to provide feedback on teachers.
13. **Check Payments**: This state allows Management to check fee payments. Management can make changes in fee Due, which reflects the students portal and database.
14. **Feedback on Teachers:** This state allows students to select a teacher to provide feedback.
15. **Fee Delays:** This state allows admins to view information on fee delays.

**Transitions**

1. The user transitions from the Login state to either the Login Error state or the Successful Login state depending on whether their credentials are valid.
2. From the Successful Login state, the user can transition to different states depending on their user type (admin, teacher, student).
3. Admins can view their profile (View Management Profile), view teacher profiles (View Teacher Profile), view student profiles (View Student Profile), check fee payments (Check Payments), and view information on fee delays (Fee Delays).
4. Teachers can view their profile (View Teacher Profile), update student marks (Update Marks), view the class schedule (View Schedule), and view feedback from students (View Feedback).
5. Students can view their profile (View Student Profile), view their marks (View Marks), view the class schedule (View Schedule), ask questions or clarify doubts with teachers (Solve Doubts), and provide feedback on teachers (Give Feedback).
6. Users can log out from any state, which transitions them back to the Login state.

**15. Collaboration Diagram**

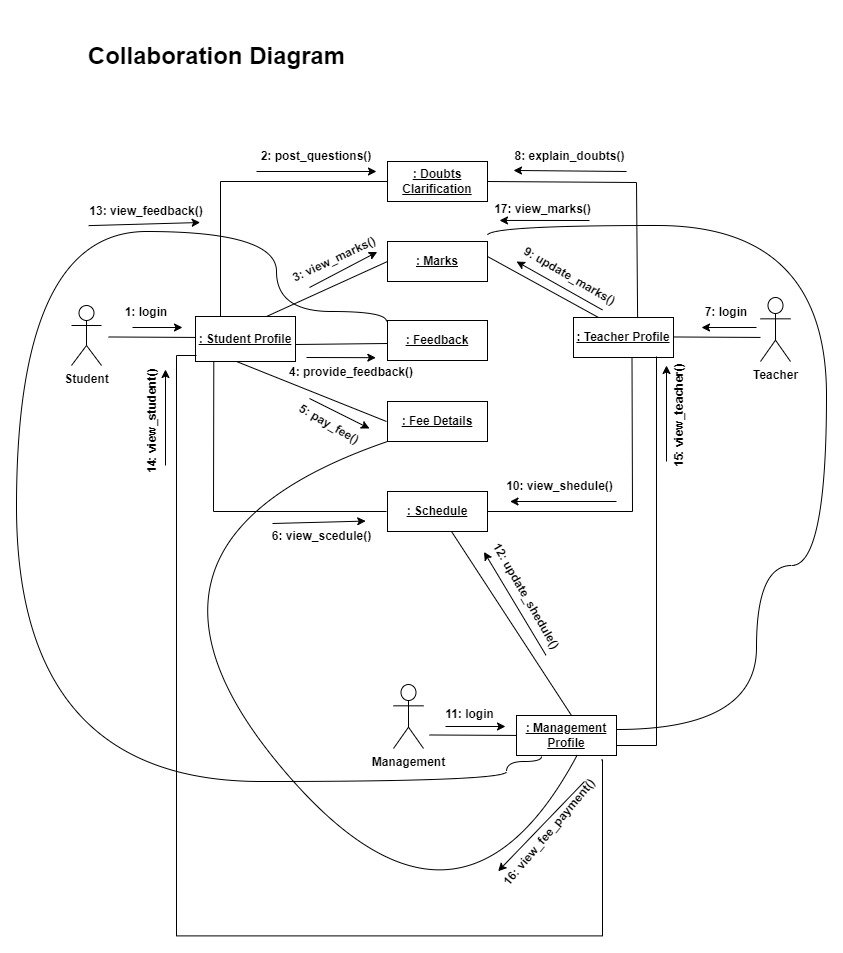
****

Fig 8: Collaboration Diagram

The interactions between different actors in the system, which include:

**Actors:**

* Student
* Teacher
* Management

**Interactions:**

1. Student Login: The student initiates the interaction by logging into the system. This message doesn't directly involve another actor, but it establishes the student's presence in the system.
2. Post Questions (Student -> Teacher): The student sends a message to the teacher, which could be a question or clarification of a doubt.
3. View Marks (Student -> System): The student interacts with the system to view their marks. The system retrieves and displays the student's marks.
4. Provide Feedback (Student -> System): The student interacts with the system to provide feedback about a teacher. The system stores the feedback.
5. View Fee (Student -> System): They can view their fee due.
6. View Schedule (Student -> System): The student interacts with the system to view their class schedule. The system retrieves and displays the student's schedule.

**Teacher Interactions:**

1. Login (Teacher -> System): Similar to the student login, the teacher establishes their presence in the system by logging in.
2. Explain Doubts (Teacher -> Student): The teacher responds to the student's questions or clarifications (from point 2) by sending an explanation message.
3. Update Marks (Teacher -> System): The teacher interacts with the system to update a student's marks. The system stores the updated marks.
4. View Schedule (Teacher -> System): Similar to the student, the teacher can view the class schedule by interacting with the system.

**Management Interactions:**

1. Login (Management -> System): The management profile logs into the system, similar to students and teachers.
2. View Student Profile (Management -> System): The management can view information about a student by interacting with the system. The system retrieves and displays the student's profile.
3. View Teacher Profile (Management -> System): Similar to student profiles, the management can view teacher information by interacting with the system.
4. View Fee Payment (Management -> System): The management interacts with the system to view information about fee payments made by students. The system retrieves and displays relevant data.

**16. Testing**

For unit testing and path coverage testing of the provided code, we'll need to break it down into smaller units and identify the paths through the code. Let's start with unit testing:

Unit Testing:

1. Test Case 1: Login with valid student credentials

- Input: Valid student username and password

- Expected Output: Redirect to /student

2. Test Case 2: Login with valid teacher credentials

- Input: Valid teacher username and password

- Expected Output: Redirect to /teacher

3. Test Case 3: Login with valid management credentials

- Input: Valid management username and password

- Expected Output: Redirect to /management

4. Test Case 4: Login with invalid credentials

- Input: Invalid username or password

- Expected Output: Stay on the login page or display an error message

5. Test Case 5: Login with non-existent user

- Input: Username that doesn't exist in the database

- Expected Output: Stay on the login page or display an error message

Path Coverage Testing:

For path coverage testing, we need to ensure that every path through the code is tested at least once. Here are some paths to consider:

1. Path 1: Login with student credentials

- Ensure the path where the user ID starts with "39" is tested.

- Check the path where the user is redirected to /student.

2. Path 2: Login with teacher credentials

- Ensure the path where the user ID starts with "25" is tested.

- Check the path where the user is redirected to /teacher.

3. Path 3: Login with management credentials

- Ensure the path where the user ID starts with "14" is tested.

- Check the path where the user is redirected to /management.

4. Path 4: Password comparison failure

- Test the path where the password comparison fails.

- Check if the user is not authenticated and an appropriate response is sent.

5. Path 5: User not found

- Test the path where the user is not found in the database.

- Ensure that an appropriate response is sent in this case.

6. Path 6: Default case

- Test the default case where the user ID doesn't match any of the predefined cases.

- Ensure that an appropriate response is sent or handled.

By designing and executing test cases for both unit testing and path coverage testing, we can ensure the reliability and correctness of the login functionality in the provided code.

**17.Conclusion and Limitations**

**Conclusion**

The ERP System stands out as a robust and comprehensive solution, designed to enhance the administrative and educational capabilities of modern educational institutions. Through its intuitively designed front-end, EduSys offers users a seamless and accessible interface, allowing students, faculty, and administrators to interact efficiently with the system across various devices. The use of modern UI/UX principles ensures that all interactions are user-friendly and visually appealing, enhancing the overall user experience and facilitating easier navigation through a multitude of functionalities.

On the back-end, EduSys demonstrates a strong foundation built on advanced server technologies and robust database management systems. This backbone supports the system’s complex data management needs and ensures reliable and secure handling of sensitive information. The back-end architecture is engineered for scalability and performance, capable of adapting to increasing demands without compromising speed or functionality. Security measures are rigorously implemented, safeguarding the system against potential threats and ensuring compliance with data protection regulations.

Moreover, the integration of both front-end and back-end elements using modern technologies and frameworks allows EduSys to offer a dynamic, responsive, and efficient educational management platform. RESTful APIs and middleware optimize the communication flow between client-side and server-side operations, further ensuring that the system operates smoothly under various operational loads.

In conclusion, EduSys is not just a tool but a strategic asset for educational institutions, streamlining operations, ensuring data integrity, and enhancing communication within the educational community. Its design and development take into consideration both current educational needs and future advancements, positioning EduSys as a pivotal solution for educational institutions looking to embrace technology and improve their educational and administrative services.

**Limitations**

While EduSys offers numerous advantages and comprehensive functionalities to educational institutions, like any complex system, it also presents certain limitations that stakeholders should consider:

**Dependence on Robust Internet Connectivity:** ERP's performance and accessibility are heavily reliant on consistent and high-speed internet connectivity. In regions where internet service is unstable or slow, users might experience delays or interruptions in accessing the system's functionalities, which can impact the efficiency of administrative tasks and academic activities.

**Initial Setup and Customization Efforts:** Implementing ERPs within an institution is not a trivial task. It involves significant setup time and customization to align with specific organizational structures and processes. This initial investment in time and resources can be a hurdle for institutions with limited IT capabilities or those that are resistant to change.

**Maintenance and Updates**: Continuous maintenance is required to ensure the system operates smoothly and remains secure against new threats. This ongoing need can strain resources, especially for institutions without a dedicated IT team. Additionally, system updates might require downtime or lead to temporary disruptions in service.

**Cost Implications**: While ERPs offer a range of essential services that justify its cost, the total expense—including license fees, customization, training, and maintenance—can be substantial. This cost barrier may be prohibitive for smaller institutions or those with limited budgets.

**Data Privacy Concerns:** Despite stringent security measures, the vast amount of sensitive data managed by ERPs makes it a potential target for data breaches. Ensuring the confidentiality, integrity, and availability of student and faculty information remains a constant concern, necessitating ongoing vigilance and investment in cybersecurity measures.

**Scalability Challenges:** While designed for scalability, the actual scaling process—whether upgrading hardware, optimizing databases, or expanding server capacity—can introduce challenges. As educational institutions grow and their needs evolve, the system must adapt quickly to avoid performance bottlenecks or service degradation.

**Integration Complexity:** Integrating ERPs with other existing platforms and systems (such as legacy databases or third-party educational tools) can be complex and time-consuming. Incompatibilities between systems may require additional workarounds or custom solutions, increasing the overall project timeline and cost.

**18. References**

1. [**Project Code (GitHub Repo)**](https://github.com/Sumanth-Siddareddy/Enterprise-Resource-Planning)
2. [**What is ERP**](https://www.youtube.com/watch?feature=shared&v=Da1hUqzoiAo)
3. [**Enterprise Resource Planning**](https://www.sap.com/india/products/erp/what-is-erp.html)
4. [**ERP Implementation**](https://www.sap.com/india/products/erp/what-is-erp/erp-implementation-best-practices.html)
5. [**Benefits of ERP**](https://www.sap.com/india/products/erp/what-is-erp/erp-benefits.html)
6. [**Web Development Tutorials**](https://www.udemy.com/course/the-complete-web-development-bootcamp)
7. [**Express session**](https://www.geeksforgeeks.org/session-management-using-express-session-module-in-node-js/)
8. [**Postgresql documentations**](https://www.postgresql.org/docs/)