PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE

S.I.E.S College of Arts, Science and Commerce(Autonomous)

Sion(W), Mumbai – 400 022.

Sikshantra

Smarter Management for Smarter Coaching

For the Partial Fulfilment for the Degree of Bachelor of

Science (Computer Science) 2025 – 2026

SUBMITTED BY

Mr. Ajay Gautam Irle -- TCS2526020

Dr. MANOJ SINGH
HEAD OF DEPARTEMNT

Dr. MANOJ SINGH
PROJECT GUIDE

DECLARATION

I, Ajay Gautam Irle, Roll No: TCS2526020, hereby declare that the project entitled “Sikshantra: Smarter Management for Smarter Coaching” submitted to the Department of Computer Science, SIES College, is my original work carried out under the guidance of Dr. Manoj Singh. This report has not been submitted elsewhere for any degree or diploma and all sources have been duly acknowledged.

I further declare that this project has not been previously submitted to any other university or institution for the award of any degree, diploma, or certificate, and that I bear full responsibility for the authenticity of the contents of this report.

Name:AJAY GAUTAM IRLE

Roll No:TCS2526020

ACKNOWLEDGEMENT

I take this opportunity to express my deep sense of gratitude to all those who have contributed in various ways to the successful completion of this project titled **"Sikshantra: Smarter Management for Smarter Coaching."**

First and foremost, I would like to convey my heartfelt thanks to my project guide, **Dr. Manoj Singh**, for his valuable guidance, constant encouragement, and insightful suggestions throughout the development of this project. His expertise, motivation, and constructive feedback have played a vital role in shaping the direction and quality of this work.

My deepest gratitude extends to my **family members**, who have been my constant source of strength, understanding, and motivation throughout this endeavor. Their support and encouragement helped me stay focused and committed.

Finally, I wish to acknowledge my **friends and classmates**, who provided valuable insights, cooperation, and moral support at every stage of the project. Their teamwork, discussions, and companionship have made this journey both productive and memorable

ABSTRACT

The increasing digitization of educational activities has transformed the way institutions manage their academic operations. However, many small and medium-sized coaching centers still rely on manual processes for attendance tracking, assignment submission, and performance monitoring. These conventional methods often lead to inefficiencies, errors, data redundancy, and communication gaps between students and teachers.

Sikshantra: Smarter Management for Smarter Coaching is a web-based academic management system developed to address these challenges by offering a unified, automated, and user-friendly platform. The system integrates multiple academic functions, including **attendance management, assignment submission, feedback collection, timetable scheduling, quiz conduction, and study material distribution**. Each module is designed to enhance transparency, accessibility, and real-time interaction within the institution.

The project follows a **three-tier architecture**—presentation layer, application layer, and database layer—ensuring scalability, maintainability, and ease of deployment. Technologies such as **HTML, CSS, PHP, JavaScript, and MySQL** are used to create a dynamic, responsive, and secure environment. The system also provides **role-based access control**, allowing Admins, Teachers, and Students to perform distinct tasks efficiently.

By automating repetitive academic operations, Sikshantra minimizes human intervention, reduces the chances of error, and improves data-driven decision-making for administrators and educators. The system's modular design enables easy customization according to the requirements of different institutions. Overall, Sikshantra demonstrates how technology can transform the traditional coaching model into a **smarter, more efficient, and collaborative academic ecosystem** suitable for modern education needs.

TABLE OF CONTENTS

LIST OF TABLES

CHAPTER 1: INTRODUCTION

In today's fast-paced and technology-driven educational environment, institutions face increasing challenges in managing academic and administrative tasks efficiently. With growing student strength and the diversification of academic activities, manual record-keeping methods have become outdated, time-consuming, and error-prone. Tasks such as maintaining attendance registers, collecting assignments, tracking submissions, and evaluating student performance are still handled manually in many small and medium-sized coaching centers. These traditional methods often lead to inconsistencies, loss of data, duplication of work, and a lack of transparency between teachers and students.

The need for automation in academic management has become more critical than ever. While large institutions and universities have access to enterprise-level ERP (Enterprise Resource Planning) systems, most small coaching centers lack the financial and technical resources to adopt such solutions. This creates a technological gap where smaller institutions continue to rely on outdated systems that limit productivity and growth. To address this gap, **Sikshantra** was conceptualized as a **web-based academic management system** specifically designed for smaller educational setups.

Sikshantra: Smarter Management for Smarter Coaching offers a comprehensive solution that automates and integrates key academic activities. The system enables teachers and administrators to record attendance, distribute and evaluate assignments, manage study materials, conduct quizzes, and gather structured feedback from students — all through a centralized platform. By eliminating paper-based workflows, Sikshantra promotes digital efficiency and ensures better accuracy and accountability.

The project emphasizes **affordability, usability, and maintainability**, making it suitable even for institutions with limited technical expertise. Its modular design ensures that new features can be added or modified as per institutional needs without disrupting existing functionalities. Furthermore, the user-friendly interface allows teachers, students, and administrators to adapt quickly with minimal training.

The **main objectives** of Sikshantra are:

- To automate and simplify routine academic management tasks such as attendance marking, assignment submission, and feedback collection.
- To provide a secure and structured access mechanism for **Admin**, **Teacher**, and **Student** roles, ensuring that users can only access authorized modules.
- To generate analytical reports and performance charts that help institutions make data-driven decisions.

- To minimize manual errors, reduce administrative workload, and improve overall communication within the institution.

The **scope** of Sikshantra extends beyond basic automation. It lays the foundation for a scalable digital ecosystem that can integrate features like cloud backup, real-time notifications, and mobile app support in the future. The platform's design allows easy deployment both on local servers and cloud environments, making it a versatile solution for diverse academic setups.

In summary, Sikshantra serves as a stepping stone toward digital transformation for small and medium-sized coaching institutions. It not only bridges the gap between traditional and modern academic management but also demonstrates how technology can enhance the quality and efficiency of education delivery.

CHAPTER 2: LITERATURE REVIEW

Over the past decade, digital platforms have revolutionized academic management. Several systems are widely adopted in educational institutions, each with unique features:

1. **Google Classroom:**
    - Enables teachers to create classes, assign homework, and grade submissions.
    - Supports collaboration through shared documents and discussion posts.
    - Limitations: Requires continuous internet access, and all users must have Google accounts. It provides limited offline functionality.
2. **Moodle (Modular Object-Oriented Dynamic Learning Environment):**
    - Open-source LMS widely used in universities.
    - Features include course management, discussion forums, quizzes, attendance tracking, and grade books.
    - Advantages: Highly customizable, free to use, supports multiple plugins.
    - Limitations: Requires hosting knowledge, technical maintenance, and sometimes a dedicated IT team.
3. **Enterprise ERPs (e.g., SAP, Oracle, Microsoft Dynamics):**
    - Comprehensive solutions for large educational institutions.
    - Include modules for student enrollment, attendance, finance, grading, and reporting.
    - Advantages: Fully integrated, professional analytics, and scalability.
    - Limitations: High cost, complex setup, not suitable for small coaching centers with limited IT staff.

These systems represent the **current state of digital academic management**, but they are generally designed for **large-scale institutions**, assuming stable infrastructure and dedicated technical personnel.

2.2 Limitations of Existing Systems

While these platforms offer several benefits, they have significant drawbacks for smaller coaching centers:

- **Internet Dependency:**
  Platforms like Google Classroom and Moodle require **stable internet access**. In regions with unstable connectivity, daily operations like marking attendance or submitting assignments can be disrupted.
- **High Maintenance and Operational Costs:**
  Enterprise ERPs require **regular software updates, server management, and technical support**, which increase operational costs. Small institutions may not have dedicated IT staff to handle these issues.
- **Complexity and Overhead:**
  Many features in large platforms are **irrelevant for small coaching centers**, causing unnecessary complexity for users. Teachers and administrators often spend extra time navigating features they do not need.
- **Training Requirements:**
  Users may require **training sessions** to effectively operate these platforms, adding time and cost overhead.
- **Recurring Subscription Costs:**
  Cloud-based LMSs and ERPs often require ongoing subscriptions. Small institutions with tight budgets may find these solutions financially unviable.
- **Limited Offline Support:**
  Offline attendance tracking, assignment submission, and student feedback are **rarely supported**, limiting usability in low-connectivity environments.

2.3 Research Gap

Analyzing existing platforms reveals several gaps for smaller institutions:

1. **Affordable Solutions:**
   Most commercial platforms are **designed for large institutions** and are expensive for small coaching centers.
2. **Offline Functionality:**
   Few systems allow users to work offline and later sync data to the cloud, which is crucial in areas with **intermittent internet access**.

3. **Modular Architecture:**
   Current platforms force institutions to adopt **all features**, even when not needed. A modular system that lets users enable only essential modules is lacking.

4. **Ease of Deployment and Maintenance:**
   Installation, setup, and ongoing maintenance of existing solutions often require technical expertise, which small institutions usually lack.

5. **User-Friendliness:**
   Many LMSs and ERPs are **feature-rich but complex**, leading to a steep learning curve for teachers and students.

These gaps highlight the need for a **lightweight, low-maintenance, and cost-effective system** tailored specifically to **small coaching centers**.

2.4 Proposed Approach

Sikshantra is designed to address the limitations and gaps identified in current systems:

- **Modular Design:**
  Institutions can select and enable only the required modules:
  - Attendance management
  - Assignment submission
  - Quiz creation
  - Feedback collection
  - Study material upload
  - Timetable management
- **Local Hosting with Optional Cloud Sync:**
  Reduces dependency on internet connectivity. Optional cloud deployment allows **remote access and scalability**.
- **User-Friendly Interface:**
  Intuitive dashboards for **Admin, Teachers, and Students** reduce the need for extensive training.

- **Offline Operation:**
  Teachers can mark attendance and manage assignments offline, syncing the data once internet connectivity is available.
- **Cost-Effective and Low Maintenance:**
  Designed for **small institutions with limited budgets**, reducing reliance on IT support or expensive subscriptions.
- **Scalable and Extensible:**
  The system's architecture allows **future upgrades** and addition of new modules without replacing the existing system.

2.4.1 Advantages Over Existing Platforms:

Feature	**Existing Platforms**	**Sikshantra**
Internet Dependency	High	Optional (local hosting)
Cost	High	Low-cost, one-time setup
Maintenance	Requires IT staff	Minimal maintenance
Modular Usage	Limited	Fully modular
Offline Support	Rare	Supported
User-Friendliness	Moderate to low	High

CHAPTER 3: SYSTEM ANALYSIS

3.1 Introduction

System Analysis is a crucial phase in the Software Development Life Cycle (SDLC), where the existing system is studied, user requirements are gathered, and the new system's design is conceptualized.
In the case of **Sikshantra: Smarter Management for Smarter Coaching**, system analysis involves understanding the challenges faced by small and medium-sized coaching institutions in managing academic activities such as attendance, assignments, feedback, and timetable scheduling.

The purpose of this analysis is to identify the operational inefficiencies of manual systems and to design an automated, centralized solution that enhances productivity, reduces redundancy, and ensures data integrity. Through this phase, both the **functional** and **non-functional** requirements are clearly defined to guide subsequent design and implementation stages.

3.2 Existing System

Most small educational institutions still rely on manual processes for managing student data, assignments, and feedback. These manual operations involve maintaining registers, printed sheets, and physical records. Such systems are prone to data loss, duplication, and human error. The lack of a centralized digital platform also limits the ability to generate reports or analyze academic performance. Communication between teachers and students is often delayed, and monitoring attendance or assignment submissions requires additional administrative effort.

Key drawbacks of the existing manual system include:

- Time-consuming data handling and storage
- High chances of human error and data inconsistency
- Difficulty in tracking attendance and submissions
- No centralized repository for feedback or reports
- Inefficient communication between stakeholders

3.3 Proposed System

To overcome these limitations, **Sikshantra** proposes a fully automated, web-based system that provides role-based access for Admin, Teacher, and Student users.

The system ensures smooth management of daily academic operations through a centralized interface. Teachers can upload assignments, mark attendance, and view student feedback, while students can access study materials, submit assignments, and attempt quizzes.
The Admin can monitor overall performance, manage user accounts, and generate analytical reports.

The proposed system focuses on:

- Automation of attendance and assignment management
- Improved transparency and communication
- Secure, role-based access control
- Real-time reporting and analytics
- Easy deployment on local or hosted servers

3.4 Feasibility Study

Feasibility analysis evaluates whether the project is **technically**, **economically**, and **operationally** practical. This ensures that resources are utilized effectively and the system delivers measurable benefits.

3.4.1 Technical Feasibility

The system is built using **open-source technologies** such as PHP, MySQL, HTML, and CSS, which are compatible with multiple operating systems.
It requires no special hardware beyond basic computing devices and can run easily on local servers using tools like **XAMPP**.
Hence, **Sikshantra** is technically feasible for small institutions with limited IT infrastructure.

3.4.2 Economic Feasibility

The solution is cost-effective since it uses free and open-source software. There are no licensing or subscription fees, and the setup cost is minimal.
Only basic training for staff is required, making it economically viable for small and medium institutions.

3.4.3 Operational Feasibility

Operational feasibility ensures that users can adapt to the system with minimal training.
Sikshantra’s interface is designed to be intuitive and easy to navigate, even for users with limited technical experience.

Reports, feedback, and attendance summaries can be generated with a few clicks, enhancing day-to-day operations.

3.5 System Requirements

Table 3.5.1: Hardware Requirements

Hardware Requirements	Specification
Processor	Dual Core or higher
RAM	4 GB minimum
Storage	1 GB
Network	Local LAN / Internet

Table 3.5.2: Software Requirements

Software Requirements	Specification
Operating System	Windows / Linux
Server	Apache (XAMPP)
Language	PHP 7.4+
Database	MySQL 5.7+

3.6 Methodology Used

The development of *Sikshantra: Smarter Management for Smarter Coaching* followed the **Iterative Development Model**, which emphasizes progressive refinement of the system through repeated cycles of design, implementation, and testing.

In this approach, the project is divided into small, manageable modules, each of which is developed and tested independently before being integrated into the main system. This method allows for incorporating user feedback, improving design flexibility, and reducing risk through early testing.

The iterative model was chosen because the project requirements evolved during development. Initially, core modules such as **User Management** and **Assignment Submission** were implemented. In later iterations, additional features such as **Attendance Tracking**, **Feedback System**, and **Timetable Management** were integrated based on project goals and testing feedback.

Each cycle of development consisted of the following steps:

1. **Requirement Analysis:** Gathering and refining the needs of Admin, Teacher, and Student users.
2. **System Design:** Creating UML diagrams and defining database structures for each module.
3. **Implementation:** Developing each module using PHP, MySQL, HTML, and CSS.
4. **Testing:** Verifying module functionality individually and after integration.
5. **Feedback and Improvement:** Adjusting features and fixing bugs before starting the next cycle.

The iterative process ensured that every new component added to *Sikshantra* was fully functional and compatible with the existing modules. This model provided flexibility, adaptability, and a smooth development experience, which is ideal for student-level web-based projects.

3.7 System Objectives

The main objectives of **Sikshantra** are:

- To automate repetitive academic processes like attendance and assignment tracking.
- To provide an easy-to-use, role-based interface for Admins, Teachers, and Students.

- To enhance data accuracy, transparency, and communication within the institution.
- To enable secure storage and retrieval of academic data through a centralized system.
- To generate reports and analytical insights to support better academic decisions.

3.8 Summary

The **System Analysis** phase confirms that Sikshantra is both **feasible and sustainable** for smaller educational setups.
The system's reliance on open-source technologies makes it economical, while its intuitive design ensures smooth adoption.
Through careful study of existing problems, requirements, and feasibility aspects, Sikshantra is designed to provide a robust, scalable, and efficient digital academic management platform.

3.9 Gantt Chart

The project planning for *Sikshantra: Smarter Management for Smarter Coaching* was carried out systematically using a Gantt Chart to visualize the overall development process and timeline. The chart illustrates the sequence of activities, their start and end dates, duration, and overlapping periods between tasks. Project work was scheduled from **1st July 2025 to 10th October 2025**, covering all major stages of development — from requirement analysis to final submission.

The first phase, **Requirement Gathering and System Study**, was conducted between **1st July and 15th July 2025**, focusing on understanding the functional and non-functional requirements of the system, analyzing existing manual operations, and identifying challenges faced by small coaching institutions.

This was followed by **System Design and UML Diagram Preparation** (16th July – 31st July 2025), during which use case, class, and data flow diagrams were developed to represent system behavior and workflow.

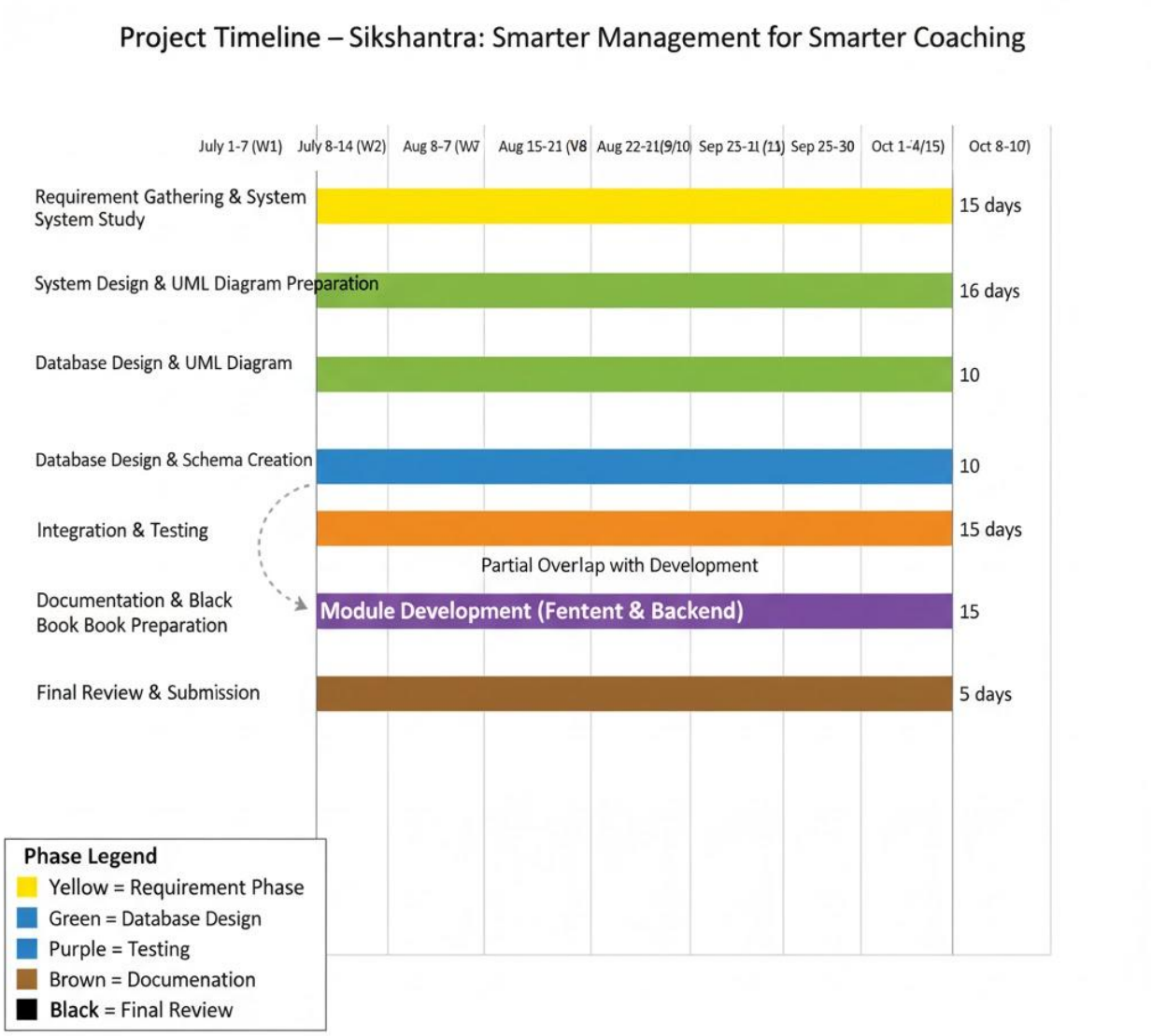
In the **Database Design and Schema Creation** phase (1st August – 10th August 2025), all database tables, relationships, and constraints were planned using MySQL. This phase slightly overlapped with the **Module Development** phase (11th August – 10th September 2025), represented by a dotted connector in the Gantt Chart. Module development included frontend creation using HTML, CSS, and PHP, and backend implementation through MySQL integration.

Subsequently, the **Integration and Testing** stage (11th September – 25th September 2025) involved merging all individual modules and performing both unit and system-level testing to ensure functionality, accuracy, and stability. After testing, the focus shifted to **Documentation**

and Black Book Preparation (26th September – 5th October 2025), where all findings, diagrams, and results were compiled according to academic guidelines.

Finally, the project concluded with the **Final Review and Submission** phase (6th October – 10th October 2025), which included a comprehensive review of system performance, final corrections, and preparation for submission.

The Gantt Chart clearly demonstrates how each task was planned and executed in an organized, time-bound manner, ensuring smooth coordination among various development activities. It highlights the structured project management approach adopted for *Sikshantra*, emphasizing timely progress, modular development, and systematic documentation.



CHAPTER 4: SYSTEM DESIGN

4.1 System Architecture

The Sikshantra system follows a **three-tier architecture**, which divides the application into three separate layers to ensure modularity, scalability, and ease of maintenance.

Presentation Layer:

The presentation layer acts as the **interface between the system and its users**. It is responsible for displaying information and receiving inputs from Admins, Teachers, and Students. Implemented using **HTML, CSS, and JavaScript**, this layer ensures a user-friendly and interactive experience. Dashboards are provided according to the user role, allowing admins to manage users and generate reports, teachers to mark attendance and upload assignments, and students to submit assignments and provide feedback. The separation of the presentation layer ensures that any changes to the interface do not affect the underlying business logic or database structure.

Application Layer:

The application layer contains the **core business logic** of Sikshantra. Built using **PHP**, it processes user requests, performs validation, executes system operations, and interacts with the database. This layer handles critical functionalities such as authentication, role-based access control, attendance management, assignment tracking, and feedback processing. It ensures that data from the presentation layer is correctly interpreted, validated, and transferred to the database layer. The application layer also facilitates modularity, allowing new features to be added without disrupting other components.

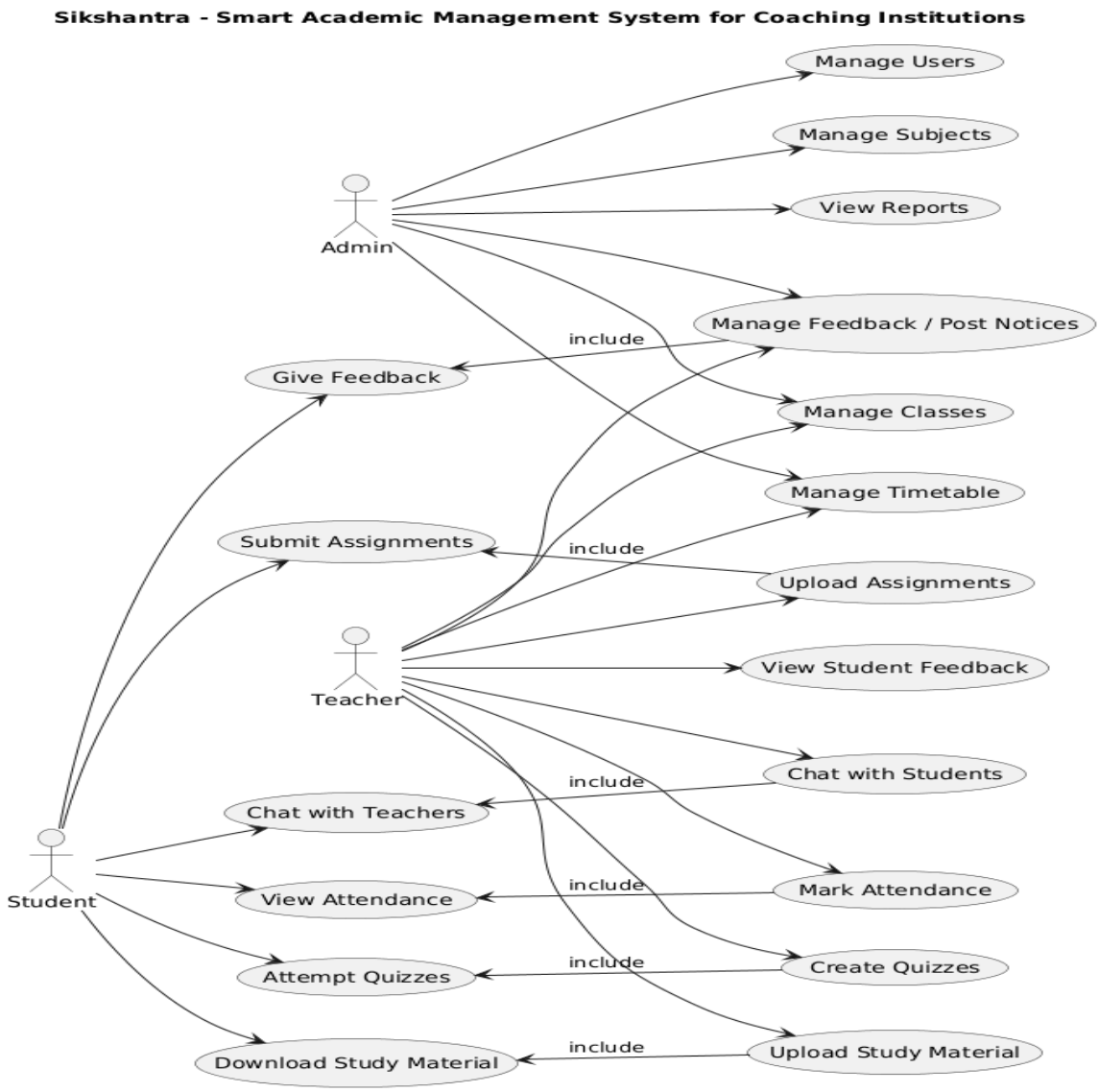
Database Layer:

The database layer is responsible for **storing, retrieving, and managing all data** in the system. Using **MySQL**, it maintains entities like Users, Teachers, Students, Assignments, Attendance, and Feedback. Proper database design, normalization, and indexing ensure fast data retrieval, integrity, and support for concurrent access. The database layer also interacts with the application layer to handle queries, updates, and transactions, making it a crucial part of the overall system architecture.

4.2 UML Diagrams

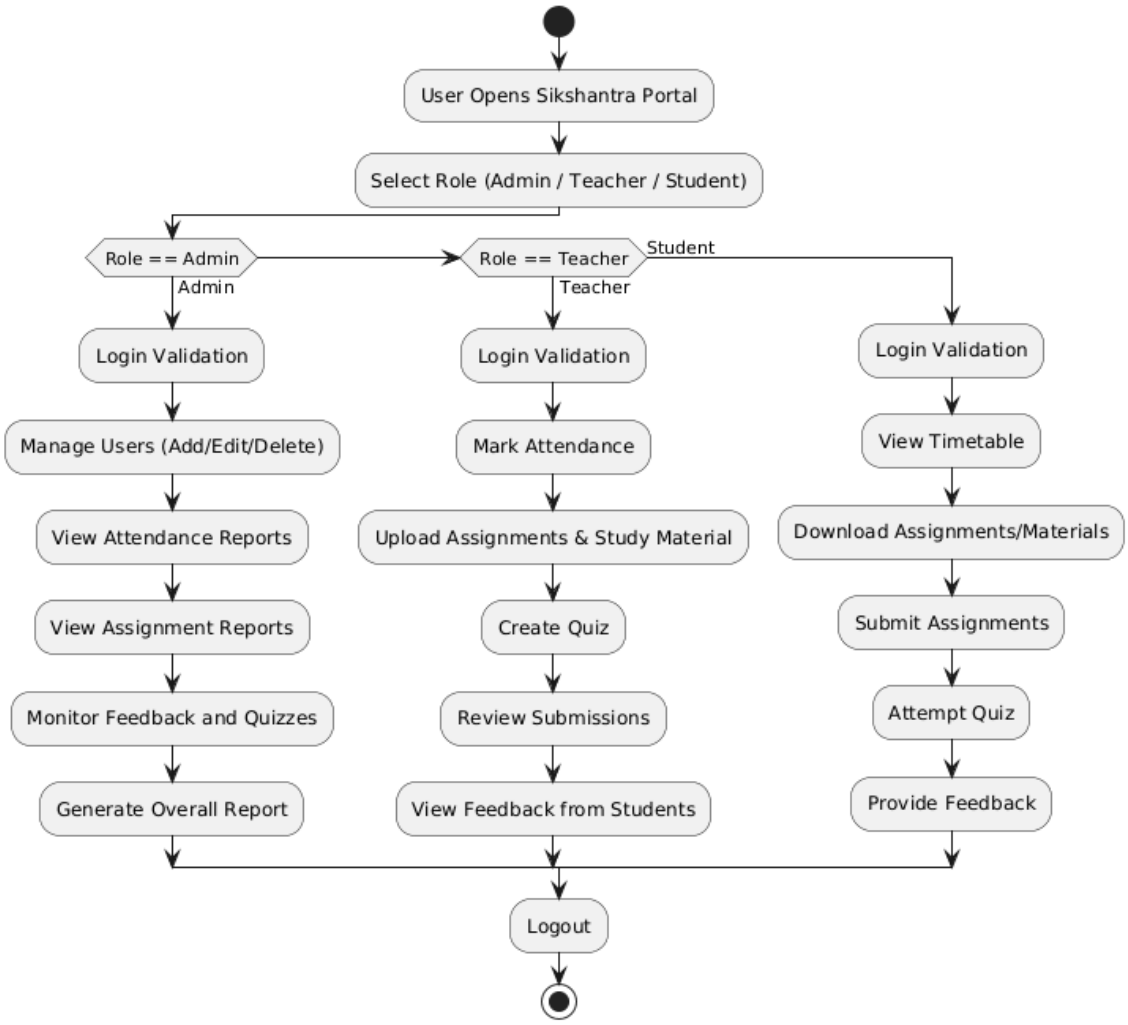
4.2.1Use Case Diagram:

A Use Case Diagram provides a **high-level visual representation of system functionality** and how different users (actors) interact with it. In Sikshantra, the primary actors are Admin, Teacher, and Student. This diagram shows the key modules like Attendance, Assignments, Feedback, and Study Material, and illustrates the interactions between actors and these modules. It helps in understanding the functional requirements of the system and ensures that all user actions are captured. Use Case Diagrams are essential during requirement analysis and system design to clarify the scope of the system.



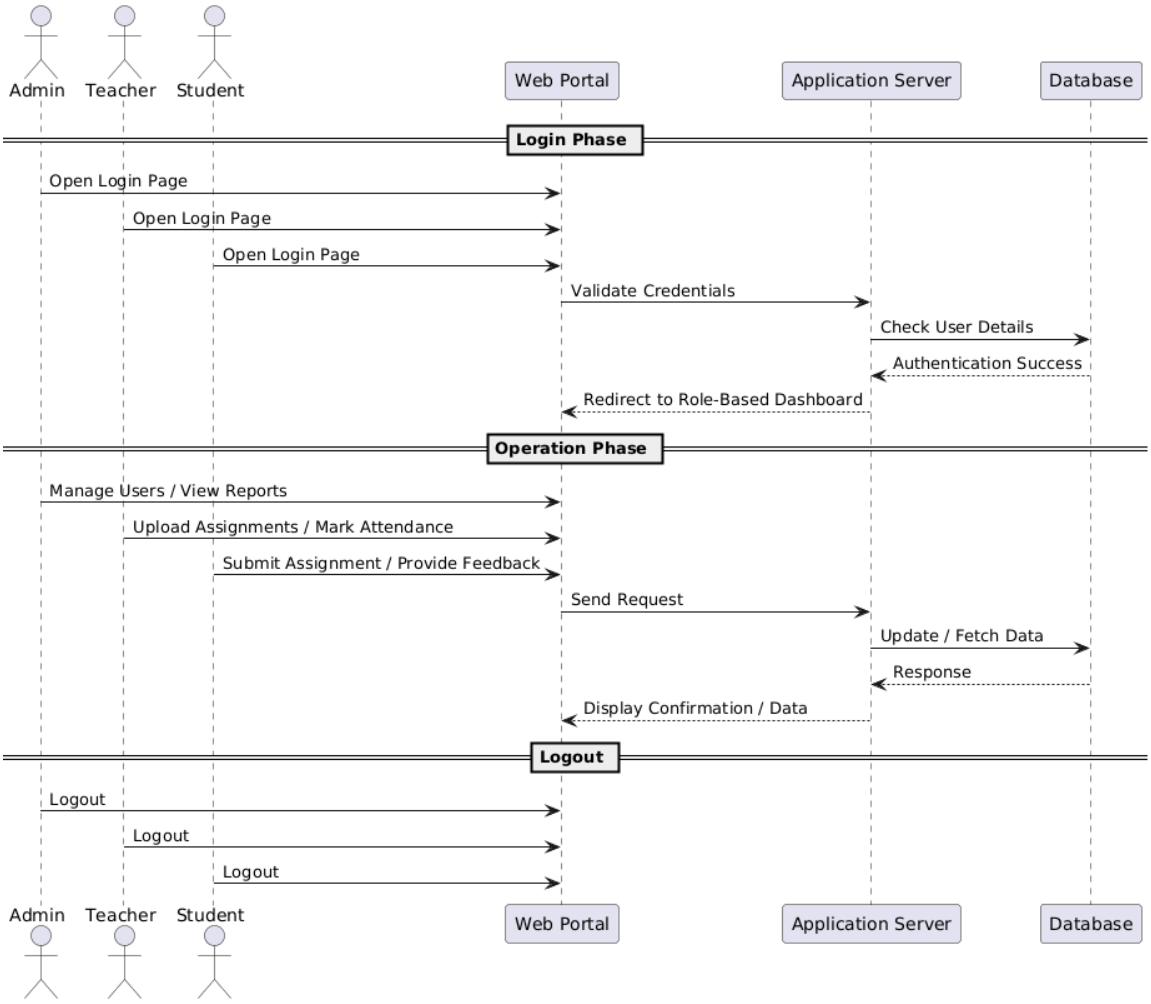
4.2.2Activity Diagram:

An Activity Diagram represents the **workflow or sequence of activities** in a process. For example, it can show the steps a student follows to submit an assignment, from login to submission confirmation. This diagram helps in visualizing complex processes, identifying potential bottlenecks, and improving system efficiency. Activity Diagrams are useful for both developers and stakeholders to understand the dynamic behavior of the system. They are particularly effective for modeling business processes and user interactions.



4.2.3Sequence Diagram:

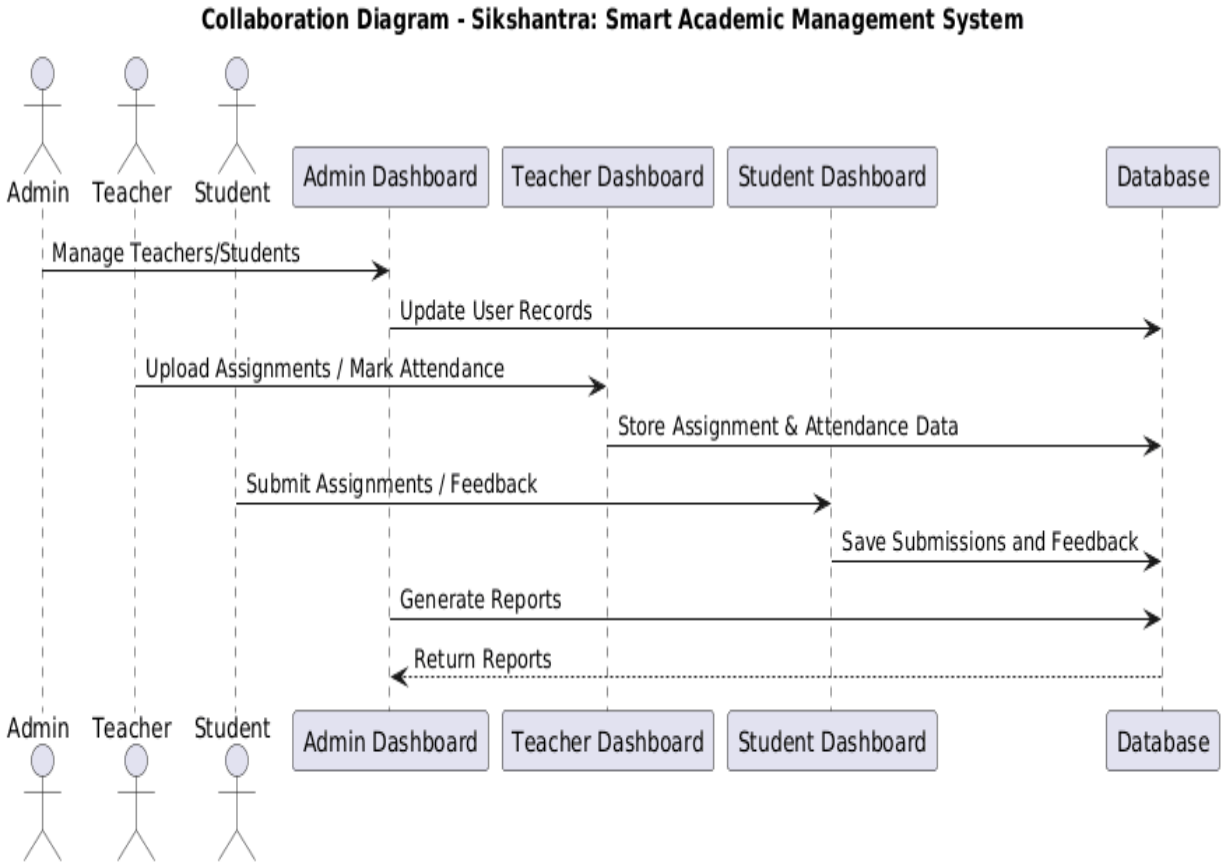
A Sequence Diagram illustrates the **order of interactions between objects** over time for a specific operation, such as login or feedback submission. It shows how messages are exchanged between the user interface, application layer, and database to complete a process. Sequence Diagrams help in identifying the flow of data, the responsibilities of different components, and the timing of events. They are crucial for designing and validating the logic of system operations before implementation.



4.2.4 Collaboration Diagram:

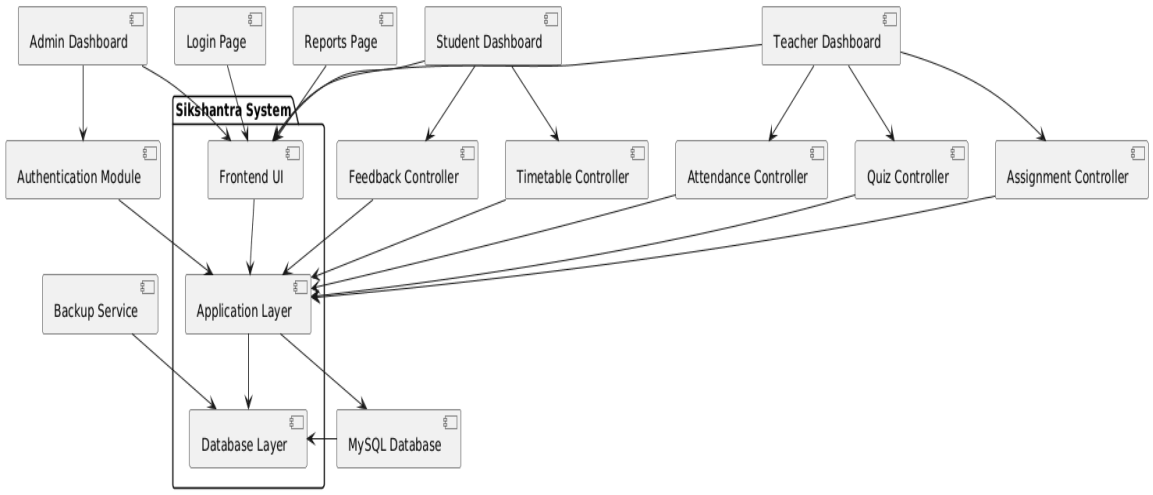
A Collaboration Diagram (or Communication Diagram) depicts **object interactions and relationships** during a process. It emphasizes the structural organization of objects and how they collaborate to perform system functions. In Sikshantra, it can show interactions between the Student object, Assignment object, and Database object during assignment submission.

Collaboration Diagrams are useful to understand dependencies between system components and improve modular design.



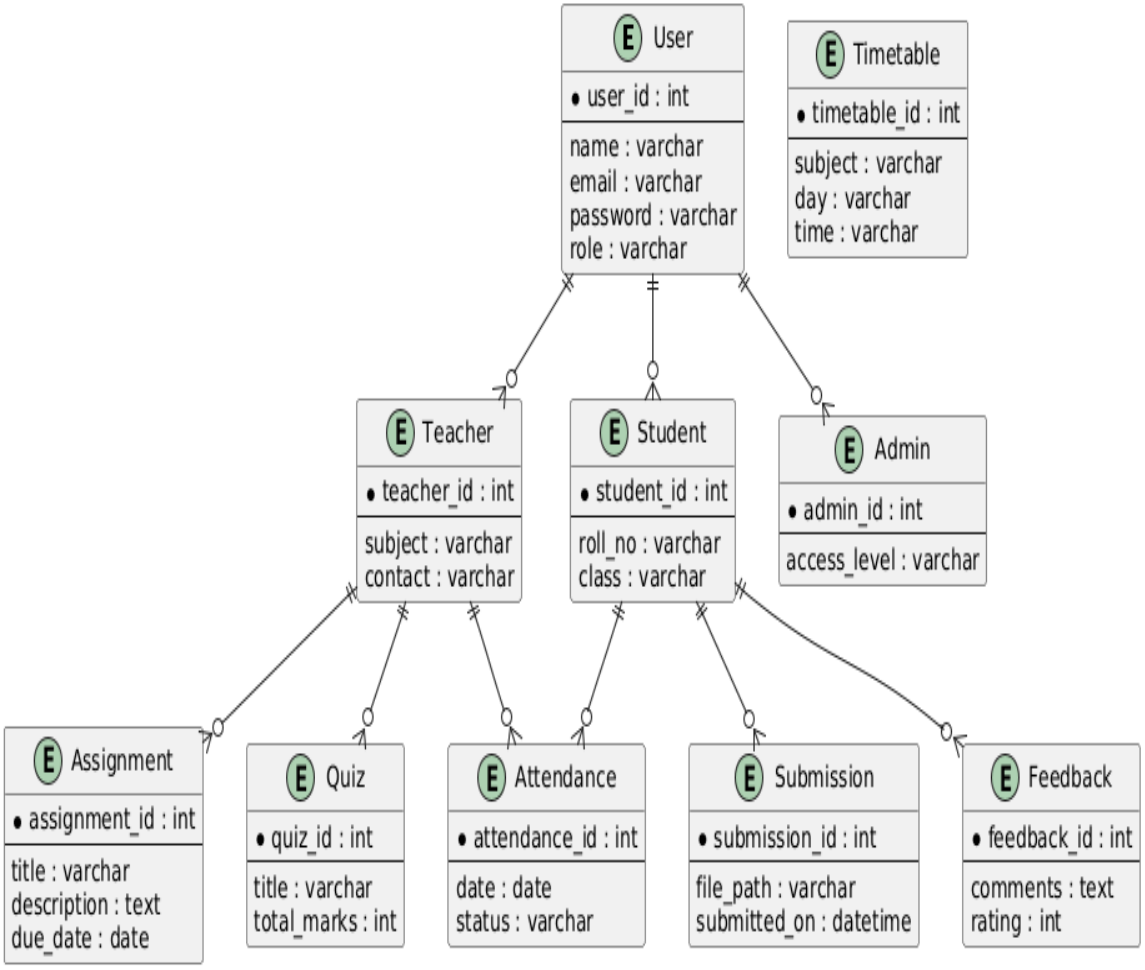
4.2.5 Component Diagram:

A Component Diagram shows the **modular components of the system** and their dependencies. It represents how the system is broken into independent, replaceable modules such as User Management, Attendance Module, Assignment Module, and Feedback Module. Component Diagrams help in planning system architecture, facilitating maintainability, and enabling future enhancements or replacements without affecting other components.



4.2.6 ER Diagram (Entity-Relationship Diagram):

An ER Diagram illustrates the **data model and relationships** between entities in the system. Entities like User, Teacher, Student, Assignment, Attendance, and Feedback are represented along with their relationships. ER Diagrams are critical for database design, ensuring proper normalization, avoiding redundancy, and maintaining data integrity. They also help developers understand how data flows and is structured within the system.



4.2.7 DFD (Data Flow Diagram) – Levels 0:

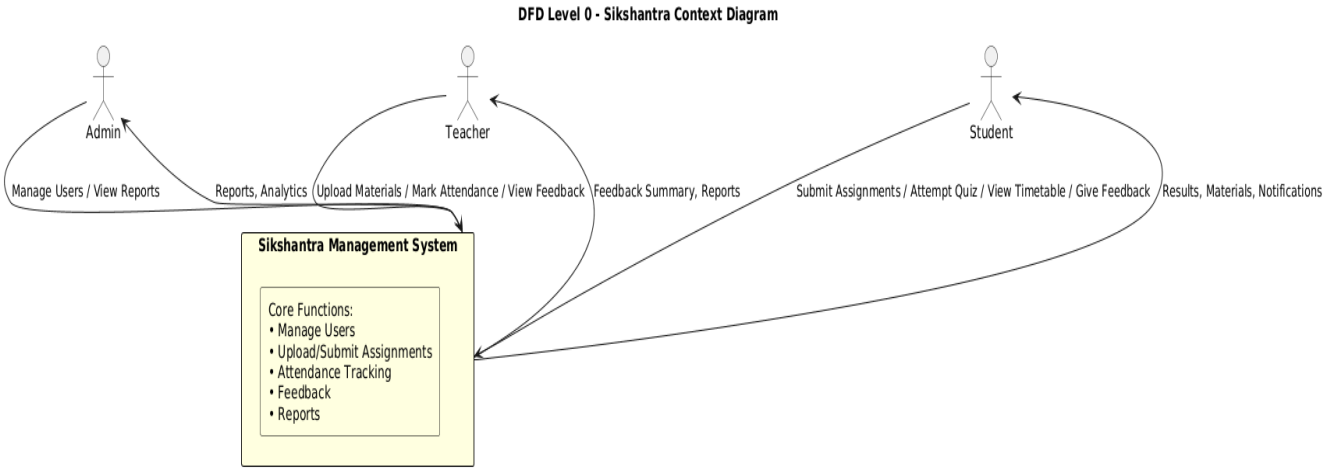
The **Level 0 Data Flow Diagram** represents the entire *Sikshantra* system as a single, high-level process and illustrates how it interacts with external entities such as the **Admin**, **Teacher**, and **Student**.

At this level, the system is treated as one unified process — *"Sikshantra Management System"* — which receives data from and sends information to these users.

- **Admin** manages users, generates reports, and monitors all system modules.

- **Teacher** uploads assignments, marks attendance, views feedback, and interacts with students' data.
- **Student** logs in to submit assignments, access materials, attempt quizzes, and provide feedback.

The flow of information between users and the system is clearly defined — for example, *Students* submit data (assignments, feedback) and receive outputs (grades, updates), while *Teachers* upload academic content and attendance records. This level provides a clear understanding of how the system interacts with its environment without diving into internal logic.



4.2.8 DFD (Data Flow Diagram) – Levels 1:

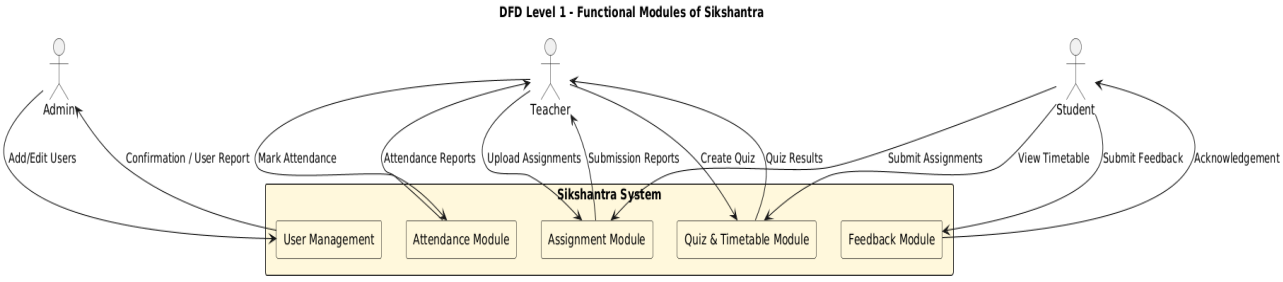
The Level 1 DFD decomposes the *Sikshantra* system into its main functional modules. Each process represents a specific subsystem or operation handled by the system.
At this stage, the high-level "Sikshantra Management System" process from Level 0 is divided into the following key processes:

1. **User Management** – Handles registration, authentication, and role-based access (Admin, Teacher, Student).
2. **Attendance Module** – Manages daily student attendance records entered by teachers.
3. **Assignment Module** – Allows teachers to upload assignments and students to submit them digitally.
4. **Feedback Module** – Collects student feedback and stores it for analysis by teachers and admins.
5. **Quiz & Timetable Module** – Allows teachers to create quizzes and upload class timetables, which students can view.

Each process interacts with the Database to either store or retrieve information.

- Admin interacts mainly with *User Management* and *Reports*.
- Teacher interacts with *Assignment*, *Attendance*, and *Feedback* modules.
- Student interacts with *Assignment*, *Quiz*, and *Timetable* modules.

This level provides a more detailed understanding of how data moves internally within the system and how each module performs its specific task.



4.2.9 DFD (Data Flow Diagram) – Levels 2:

The **Level 2 DFD** offers the most detailed representation of data flow within individual modules of *Sikshantra*. Each process from Level 1 is further broken down into smaller, specific operations, showing how inputs are transformed into outputs step-by-step.

For example:

a) Assignment Module (Level 2 Breakdown)

- **Teacher uploads assignment** → System validates and stores data in the Assignment Database.
- **Student downloads assignment** → System retrieves data from the same database.
- **Student submits assignment** → File and metadata are stored as a new record in the *Submission Database*.
- **Teacher reviews submission** → System fetches stored data and displays submission status.

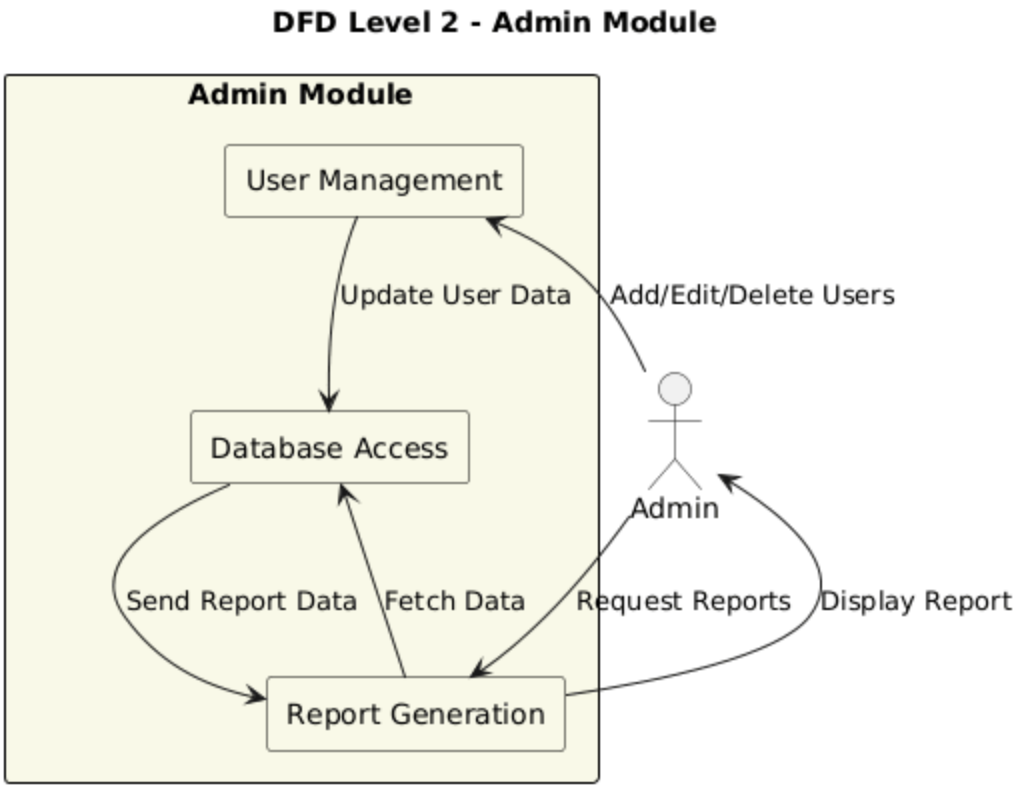
b) Attendance Module (Level 2 Breakdown)

- **Teacher marks attendance** → Data stored in *Attendance Database*.
- **Student views attendance** → System fetches data for the logged-in student.
- **Admin generates attendance report** → Aggregated data is processed and displayed in tabular or graphical form.

c) Feedback Module (Level 2 Breakdown)

- **Student submits feedback** → Input stored in *Feedback Database*.
- **Teacher/Admin reviews feedback** → Data fetched, summarized, and visualized.
- **Reports generated** → Summarized data stored in *Report Database* for decision-making.

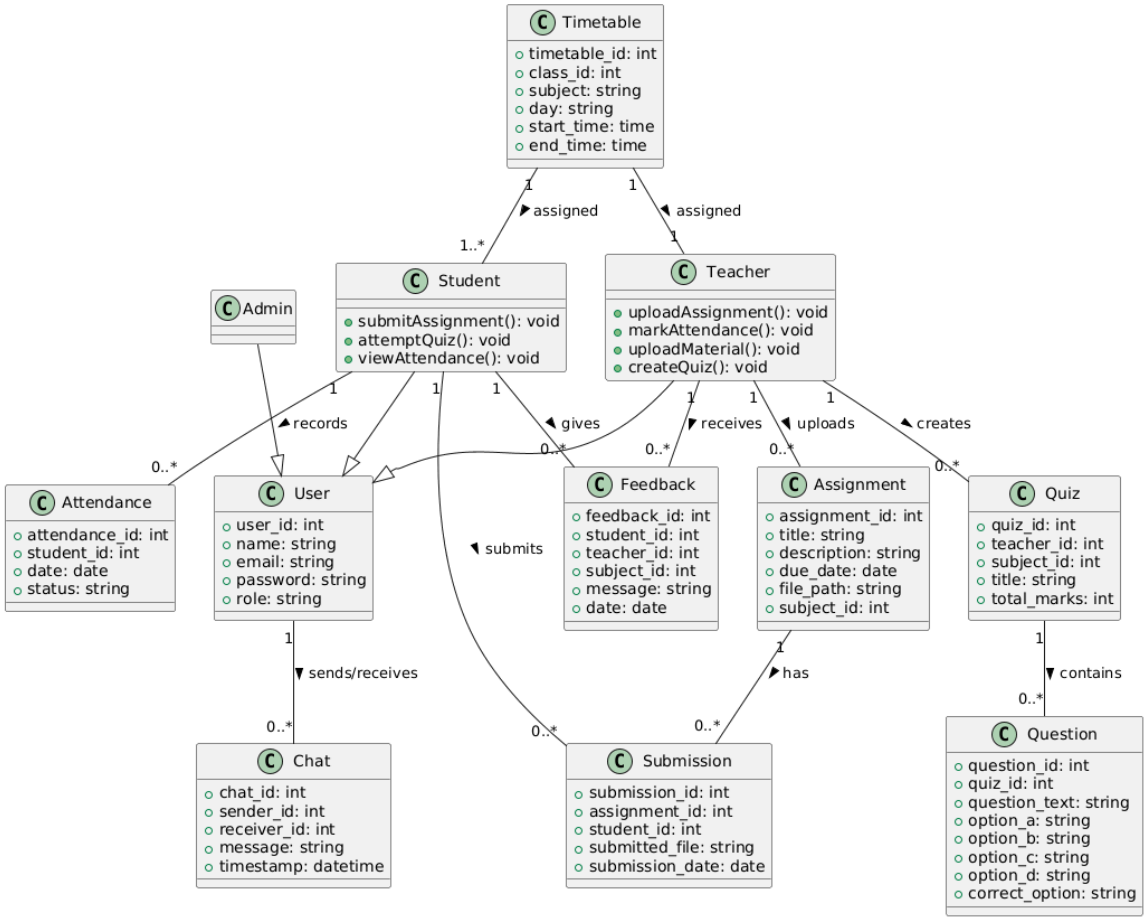
This detailed view highlights how different modules communicate with the **central database**, how each actor interacts with the system, and how data transformations occur internally.
By mapping these interactions, the Level 2 DFD ensures that all data pathways are clearly understood, leading to a more accurate and efficient system design.



4.2.10 Class Diagram:

A **Class Diagram** is one of the most important UML (Unified Modeling Language) diagrams used in system design. It represents the **static structure** of the system by showing its classes, attributes, operations (methods), and the relationships among different entities. In object-oriented modeling, a class diagram serves as a blueprint for constructing the software system, providing a clear and detailed view of how various objects interact and collaborate.

In the context of **Sikshantra: Smarter Management for Smarter Coaching**, the class diagram illustrates the main classes such as **User**, **Admin**, **Teacher**, **Student**, **Assignment**, **Attendance**, and **Feedback**, along with their key attributes and methods. For example, the User class may include attributes like *user_id*, *name*, *email*, and *password*, and serve as a parent class for *Admin*, *Teacher*, and *Student* through inheritance relationships.



CHAPTER 5: IMPLEMENTATION

The implementation of the **Sikshantra** system is carried out using a combination of modern web technologies to ensure **reliability, scalability, and ease of use**. The system has been carefully designed to meet the practical needs of small coaching centers, integrating multiple academic management functions into a single, user-friendly platform.

5.1 Technology Stack Used

- **Backend:** PHP – Handles server-side processing, including data validation, session management, user authentication, and database interactions. PHP is chosen for its compatibility with web servers and ability to create dynamic web applications.
- **Database:** MySQL – Stores and manages all academic records, such as user information, attendance, assignments, feedback, quizzes, and study materials. The database design uses proper normalization and indexing to ensure fast retrieval and maintain data integrity.
- **Frontend:** HTML, CSS, JavaScript – Provides an interactive, responsive, and visually appealing interface. HTML structures the content, CSS manages the design and layout, and JavaScript adds interactivity like real-time form validation and dynamic content updates.

5.2 Algorithm / Pseudocode

User Login Algorithm (Generalized)

START

1. User enters username and password.

2. System checks the credentials against the database.

3. IF credentials are valid THEN

    a. Identify user role (Admin / Teacher / Student)

    b. Redirect to respective dashboard

  ELSE

    Display error message "Invalid username or password"

END

Attendance Management Algorithm (Teacher)

START

1. Teacher selects class and date.

2. Retrieve student list for the class from database.

3. Mark attendance as Present / Absent for each student.

4. Save attendance records in the database.

5. Generate summary report (optional).

END

Assignment Submission Algorithm (Student)

START

1. Student selects assignment to submit.

2. Upload assignment file(s) and add comments (if any).

3. System validates file type and size.

4. Save submission details in the database.

5. Confirm successful submission to the student.

END

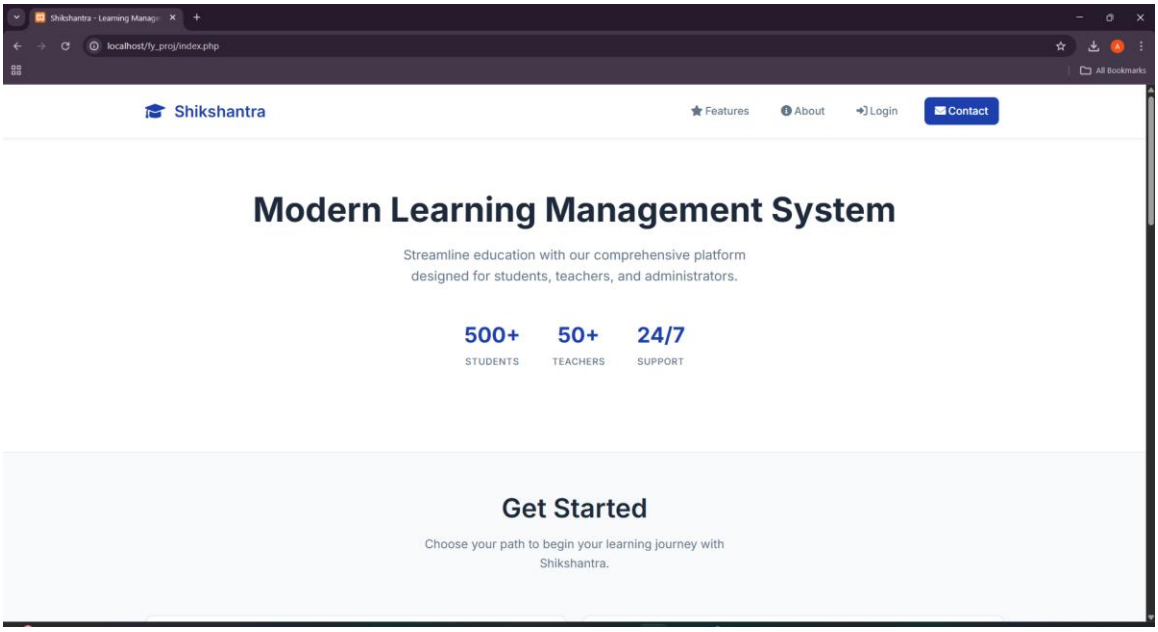
5.3 Step-by-Step Implementation

1. **Database Design and Setup**
    - Created MySQL tables for Users, Students, Teachers, Assignments, Attendance, Feedback, and Quizzes.
    - Applied normalization and indexing for faster queries and data integrity.
2. **Backend Development (PHP)**
    - Implemented user authentication and role-based access control.
    - Developed CRUD operations for managing students, teachers, assignments, attendance, and feedback.
    - Added server-side validation and security measures such as input sanitization and password hashing.

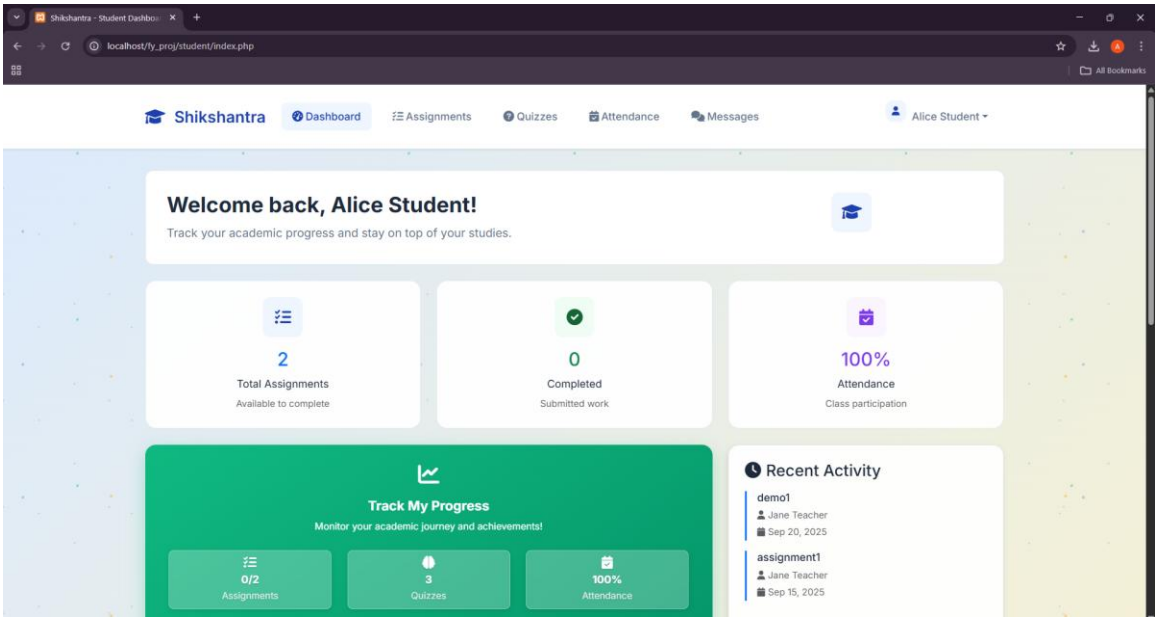
3. **Frontend Development (HTML, CSS, JavaScript)**
    - Designed separate dashboards for Admin, Teacher, and Student.
    - Implemented forms for attendance, assignments, and feedback submission with real-time validation.
    - Added responsive design to ensure usability across devices.
4. **Role-Based Dashboard Implementation**
    - **Admin Dashboard:** User management, report generation, overall system monitoring.
    - **Teacher Dashboard:** Class management, assignment uploads, attendance marking, quiz creation, and performance tracking.
    - **Student Dashboard:** Assignment submissions, quiz attempts, study material access, and feedback submission.
5. **Security and Error Handling**
    - Implemented session management to prevent unauthorized access.
    - Passwords stored using hashing algorithms.
    - Input sanitization to prevent SQL injection and XSS attacks.
    - Error handling for failed operations and invalid inputs.

5.4 SCREENSHOTS:

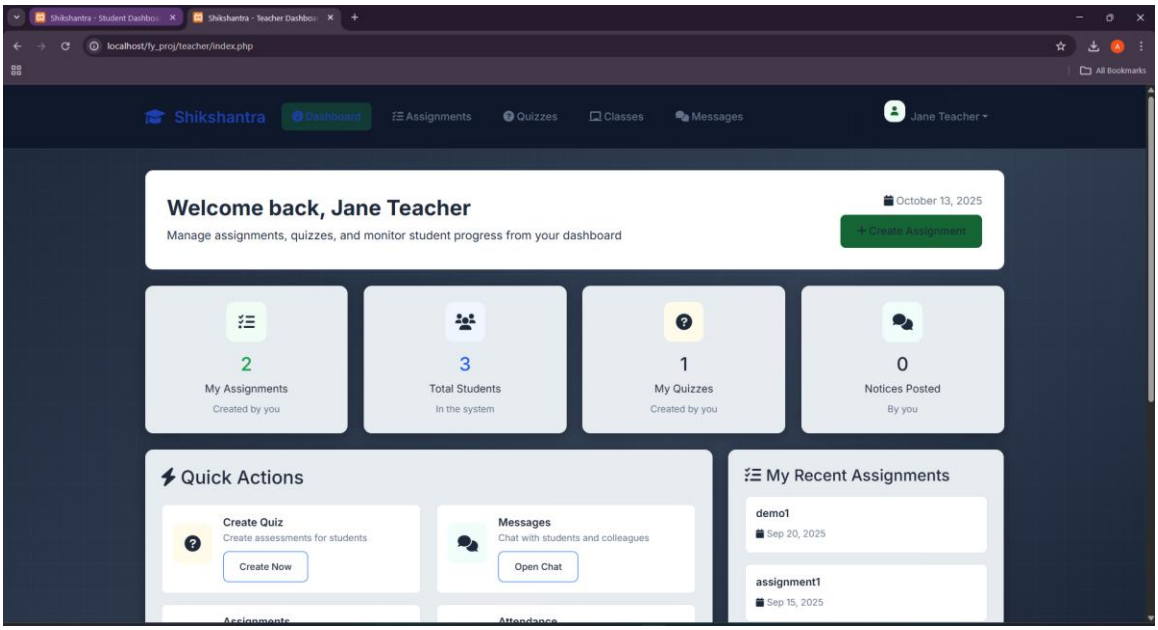
5.4.1 Main HomePage:



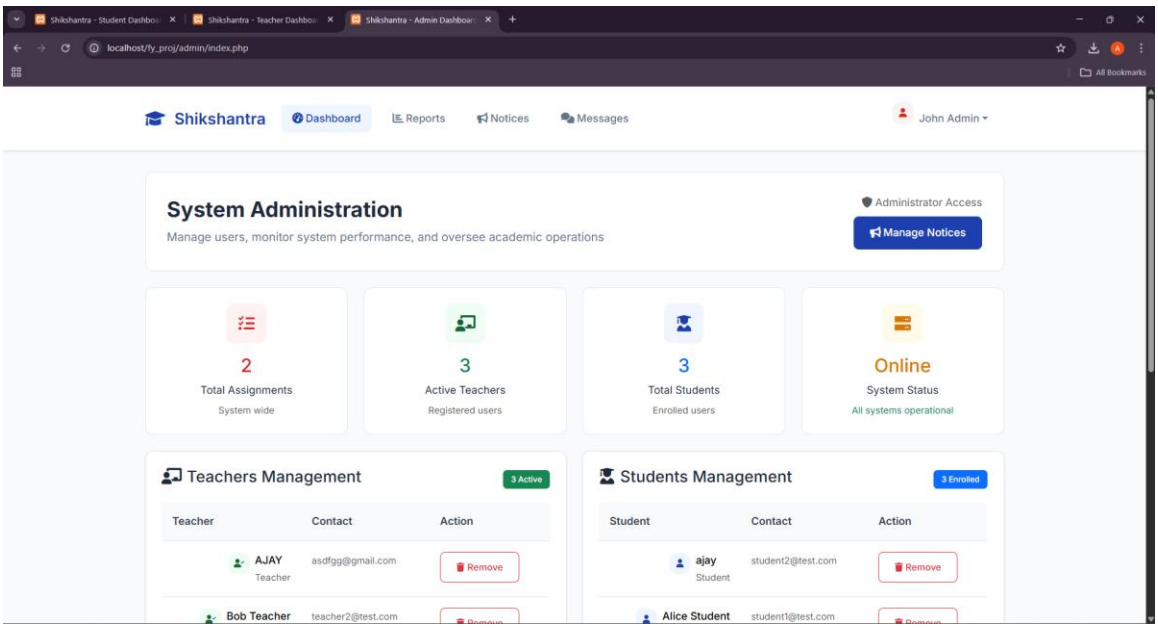
5.4.2 Dashboard(STUDENT):



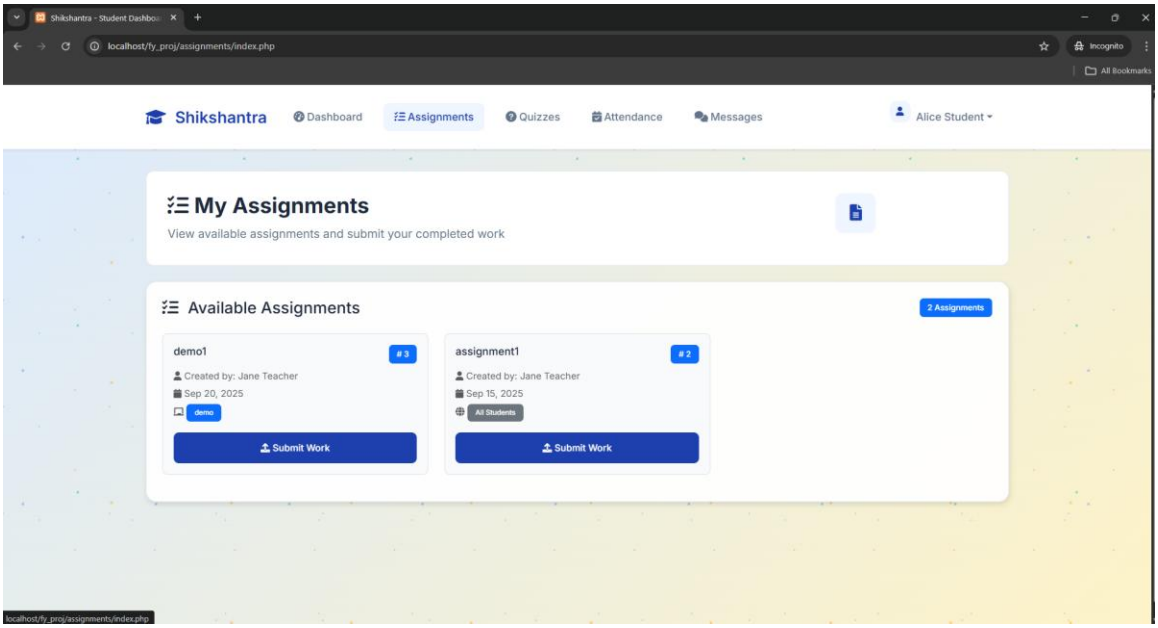
5.4.3 Dashboard(Teacher):



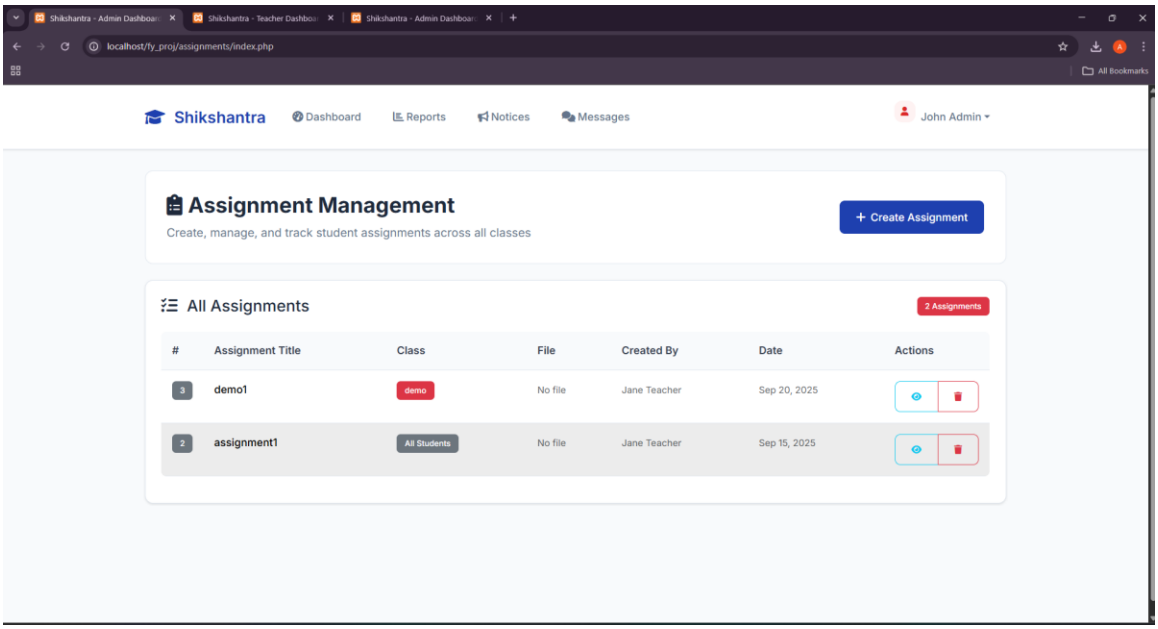
5.4.4 Dashboard(Admin):



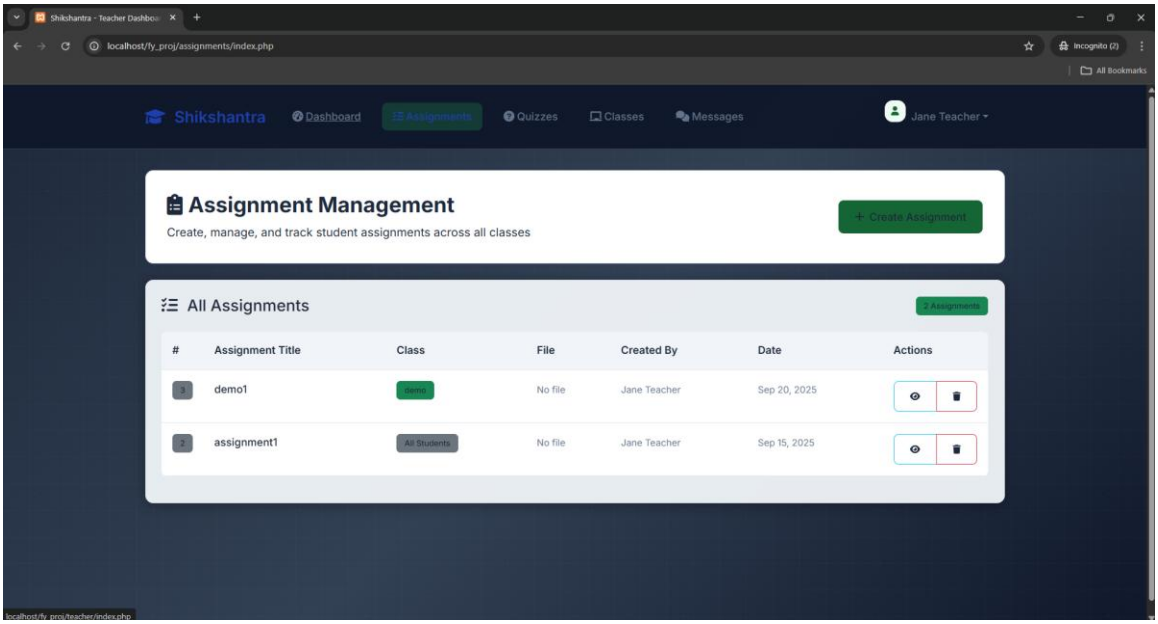
5.4.5 Assignment(Student):



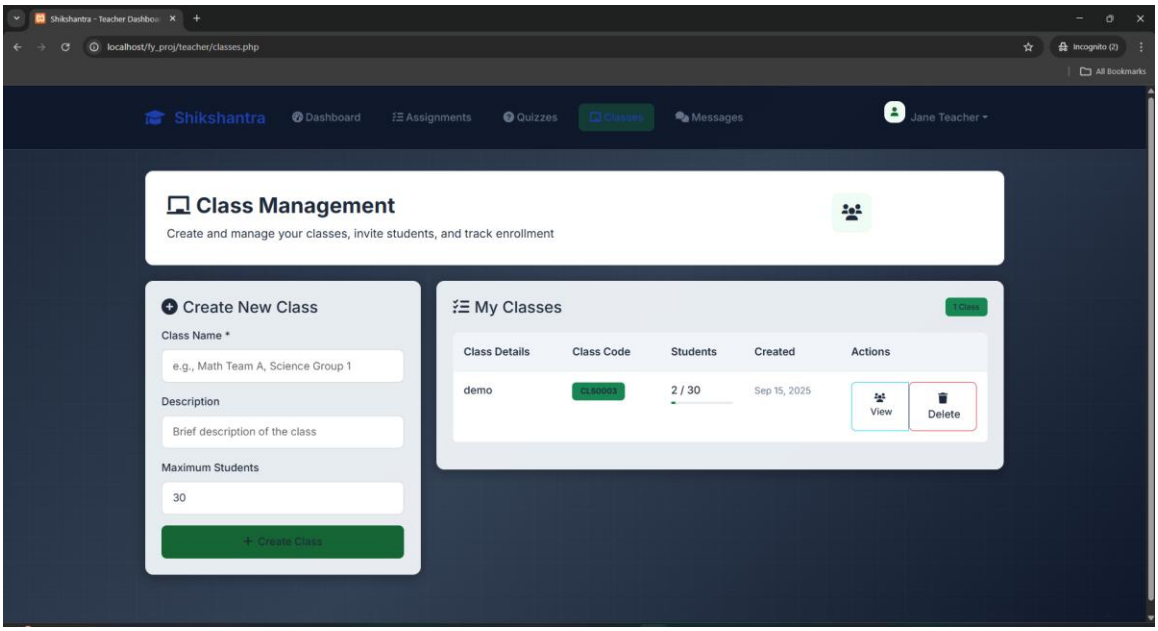
5.4.6 Assignment(Admin):



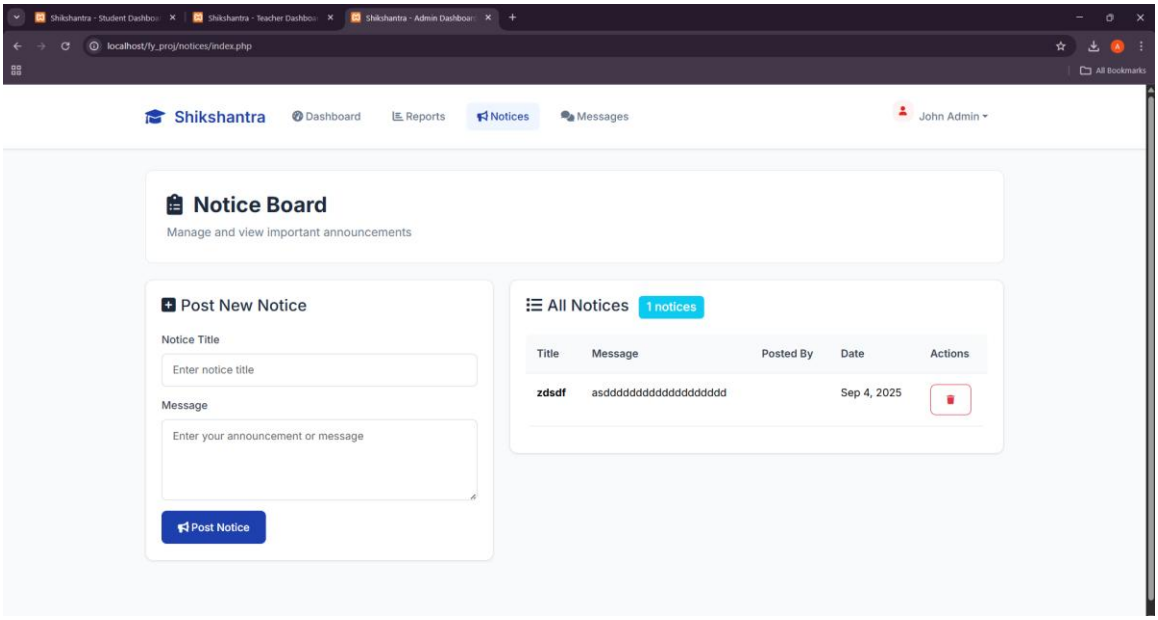
5.4.7 Assignment(Teacher):



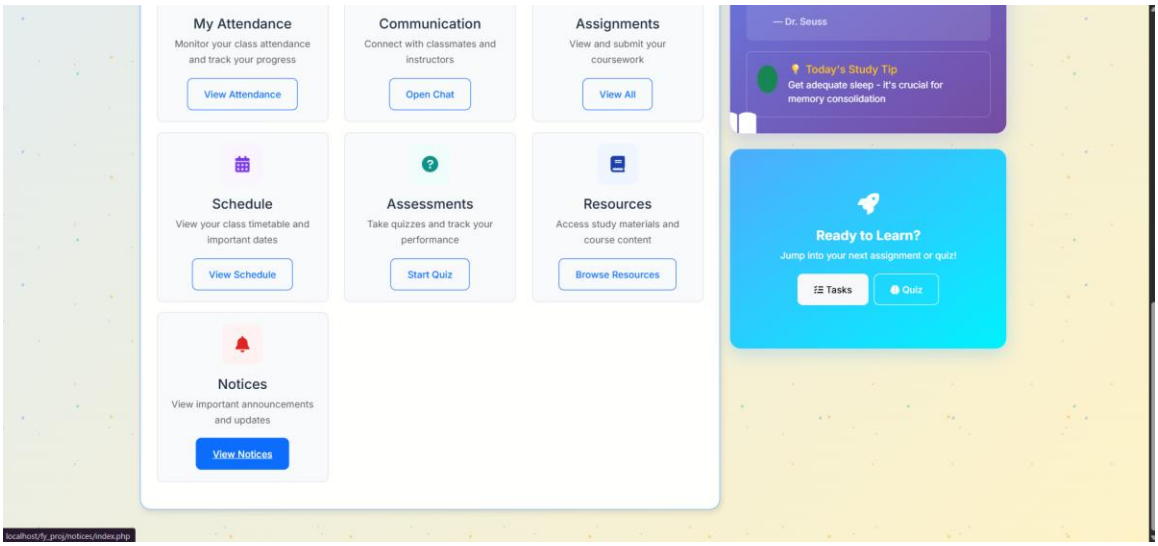
5.4.8 Class(Teacher):



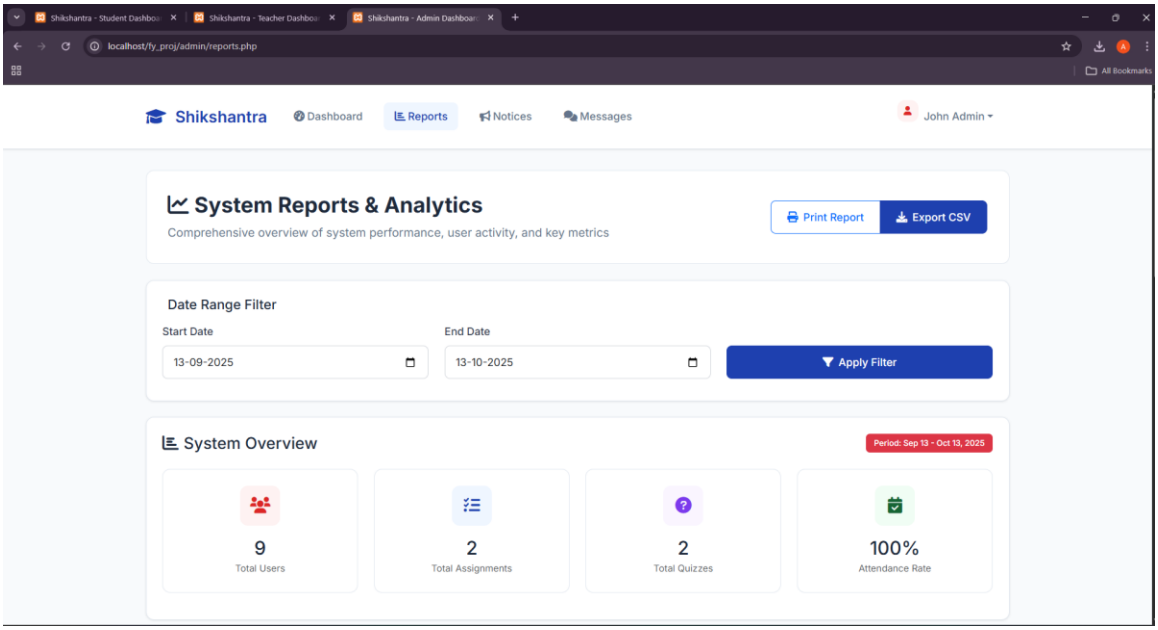
5.4.9 Notice(Admin):



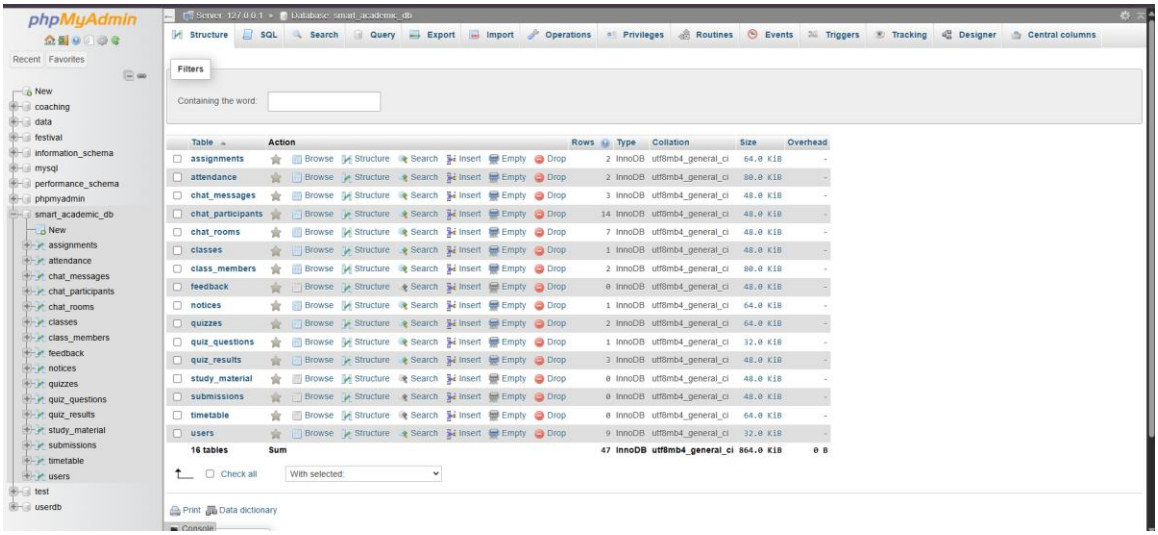
5.4.10 Notice(Student):



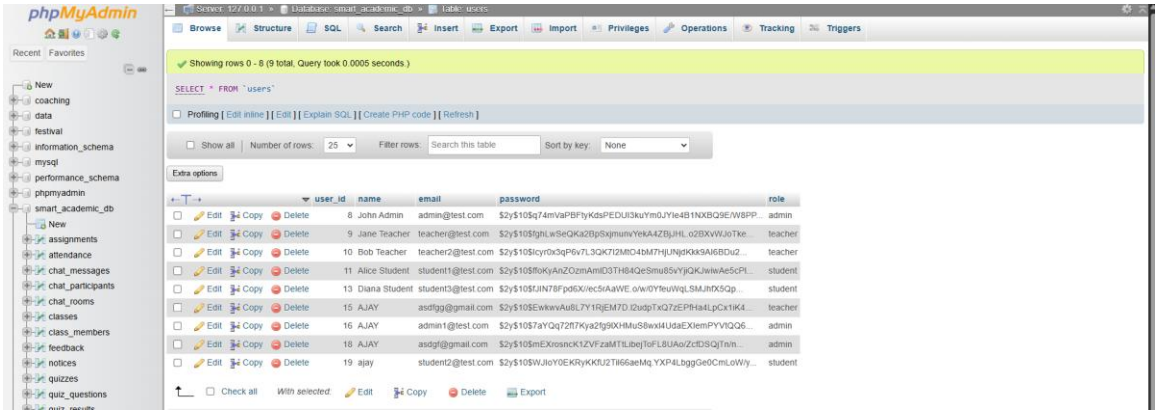
5.4.11 ReportsPage(Only for Admin):



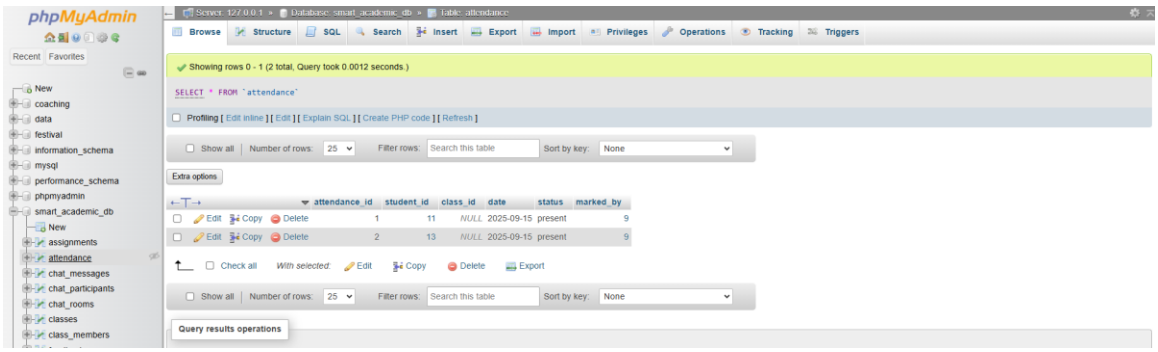
5.4.12 DATABASE(PHP):



5.4.13 User(Table):



5.4.13 Attendance(Table):



CHAPTER 6: RESULTS AND DISCUSSION

The testing phase of the Sikshantra system was conducted using **simulated data sets** to evaluate the performance, accuracy, and reliability of various modules. Different scenarios were created to mimic real-world operations in a coaching center, including multiple users with different roles, various classes, assignment submissions, attendance records, and feedback entries. This approach ensured that the system was tested under conditions similar to actual usage.

During testing, the system successfully performed all **key operations**. The **login module** correctly authenticated users based on their roles, preventing unauthorized access and ensuring secure access to dashboards. The **attendance management module** accurately recorded and retrieved attendance for students across different classes, reducing the likelihood of errors that are common in manual attendance tracking. The **assignment management module** enabled teachers to upload assignments and allowed students to submit their work efficiently, while maintaining proper timestamps and submission records. The **feedback collection module** allowed students to submit feedback on courses and teaching methods, and ensured that the data was stored accurately and could be easily retrieved for analysis.

6.1 Testing Methodology

The **testing phase** of Sikshantra was conducted using **simulated datasets** to evaluate the **performance, accuracy, and reliability** of all system modules. Various scenarios were created to mimic real-world operations in a coaching center, including:

- Multiple users with **different roles** (Admin, Teacher, Student)
- Various classes and subjects
- Assignment uploads and submissions
- Attendance marking and retrieval
- Feedback collection

This **scenario-based approach** ensured that the system was tested under conditions similar to actual usage, identifying potential issues before deployment.

6.2 Testing of Individual Modules

1. **Login and Authentication Module:**
    - Verified that users are **authenticated based on their roles**.
    - Prevented unauthorized access to sensitive modules.

- Result: Secure and role-based dashboard access successfully implemented.

2. **Attendance Management Module:**
    - Teachers marked attendance for students across multiple classes.
    - Attendance retrieval and reports were tested for accuracy.
    - Result: Reduced errors compared to manual attendance tracking.
3. **Assignment Management Module:**
    - Teachers uploaded assignments with deadlines.
    - Students submitted assignments efficiently, maintaining **timestamps and submission records**.
    - Result: Automated workflow reduced manual errors and saved time.
4. **Feedback Collection Module:**
    - Students submitted feedback on courses and teaching methods.
    - Data was stored accurately and could be **retrieved for analysis**.
    - Result: Enabled administrators and teachers to evaluate teaching effectiveness.
5. **Reports and Analytics:**
    - Graphical reports tested for **attendance patterns, feedback trends, and class performance**.
    - Charts provided visual insights for faster decision-making.

6.3 Sample Input and Output

6.3.1 Sample Input – Attendance:

Student ID	Name	Class	Date	Status
101	Ajay	CS-TY	12/10/2025	Present
102	Priya	CS-TY	12/10/2025	Absent

6.3.2 Sample Output – Attendance Report:

Student Name	Total Classes	Attended	Attendance %
Ajay	20	18	90%
Priya	20	15	75%

6.3.3 Sample Input – Assignment Submission:

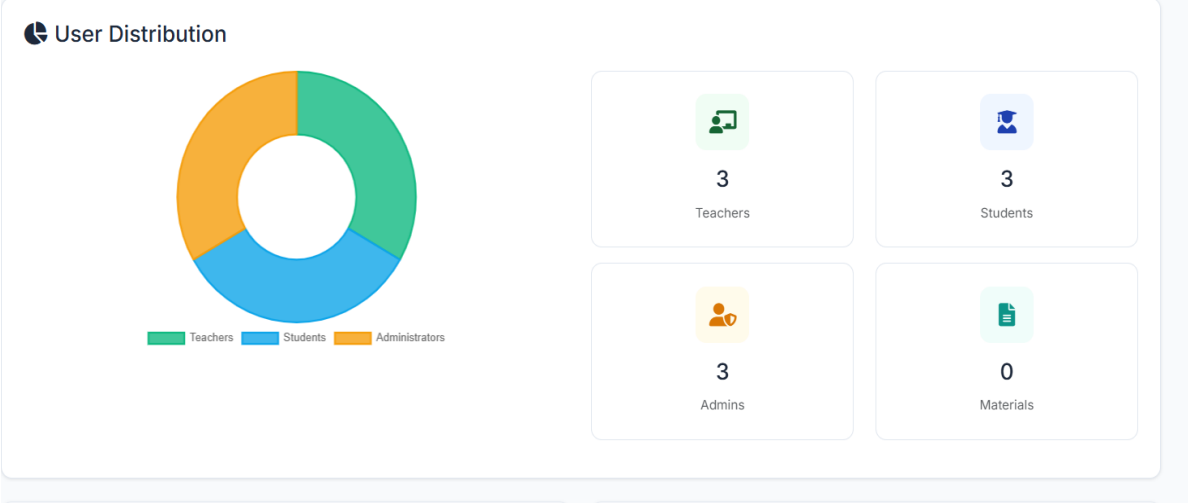
Assignment ID	Title	Student Name	Submission Date	Status
A101	Python Project	Ajay	11/10/2025	Submitted
A101	Python Project	Priya	-	Pending

6.3.4 Sample Output – Assignment Status:

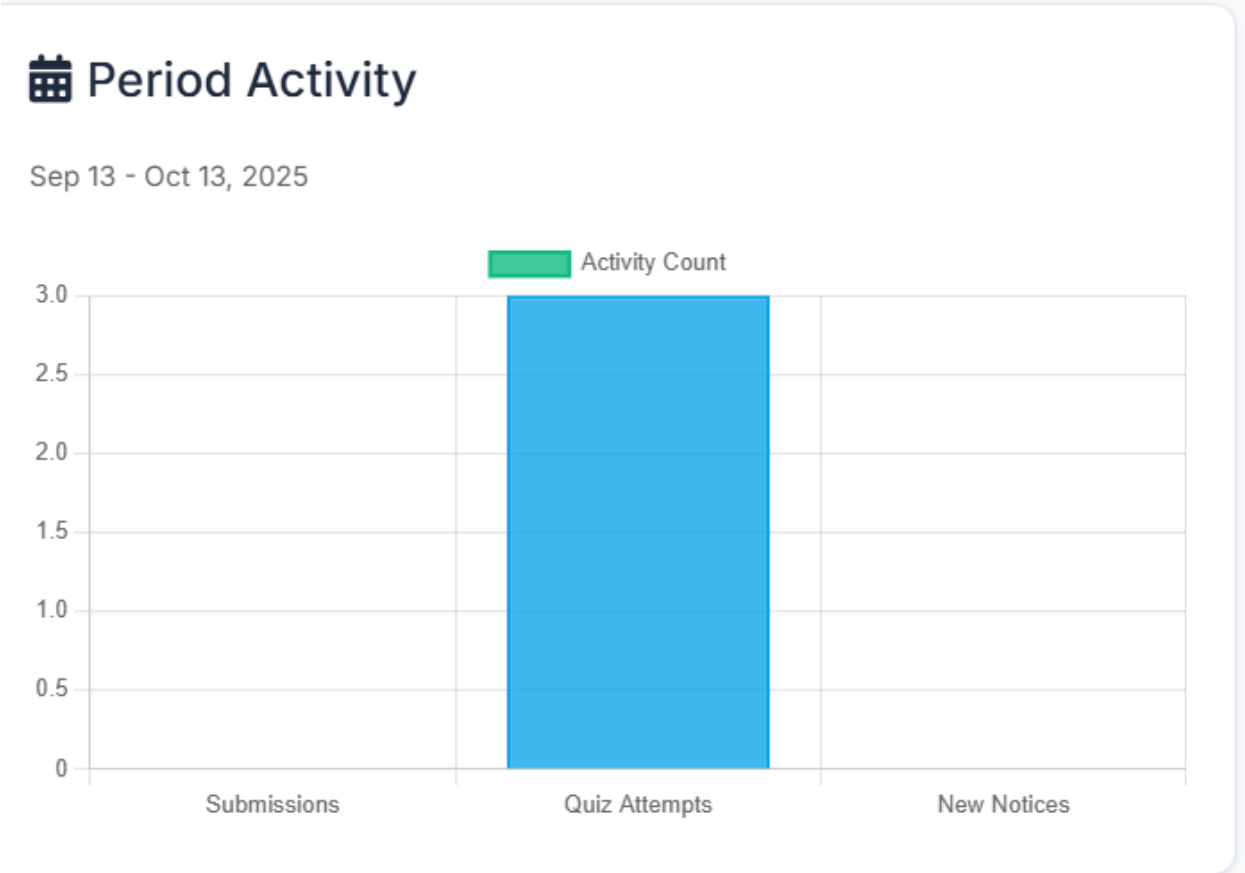
Assignment Title	Total Students	Submitted	Pending
Python Project	2	1	1

6.4 Graphs and Charts

- **User-Chart :**



- **Activity Chart:**



6.5 Analysis of Results

- **Performance:** All modules responded efficiently with minimal latency.
- **Accuracy:** Attendance, assignments, and feedback data were **recorded and retrieved accurately**.
- **Efficiency:** Automated workflows **reduced manual effort** for teachers and administrators.
- **User-Friendliness:** Intuitive dashboards allowed users to complete tasks **without prior technical knowledge**.
- **Reliability:** System performed well under **simulated multiple-user conditions**, confirming scalability potential.
- **Decision Support:** Graphical reports enabled **data-driven decisions** for attendance monitoring, performance evaluation, and feedback analysis.

CHAPTER 7: CONCLUSION AND FUTURE SCOPE

7.1 Summary of Work

The Sikshantra system successfully achieves its primary objective of **streamlining academic operations** for small coaching centers and educational institutions. By integrating key modules such as **attendance management, assignment handling, feedback collection, and study material distribution** into a single platform, the system reduces the manual workload of administrators and teachers while improving data accuracy and accessibility. Role-based dashboards ensure that each user—whether an administrator, teacher, or student—can efficiently perform their respective tasks, making the overall academic process more organized and transparent.

Through its **modular and adaptable design**, Sikshantra provides a **cost-effective and practical solution** for institutions that cannot afford large-scale, internet-dependent platforms. Even smaller institutions can now benefit from digital management tools without requiring extensive technical infrastructure or ongoing maintenance. Testing with simulated data sets confirmed the **reliability, usability, and efficiency** of the system in performing key operations such as login authentication, attendance tracking, assignment submission, and feedback collection.

7.2 Achievements

- Developed a **fully functional role-based academic management system** for Admins, Teachers, and Students.
- Integrated multiple academic modules under a **single user-friendly platform**, reducing manual effort.
- Ensured **data accuracy, security, and easy accessibility** through proper backend and database design.
- Tested the system thoroughly to validate its **efficiency and reliability**.
- Provided scope for **modular expansion**, making it adaptable for future enhancements.

7.3 Limitations

- The current system does not yet fully support **multi-branch institutions**.
- **Mobile application** integration is not implemented, limiting accessibility on smartphones and tablets.
- **AI-based analytics** and automated predictive insights are not part of the current implementation.
- Notifications via SMS or email are not fully automated at present.

7.4 Future Enhancements

- **Mobile Application Integration:** Teachers and students can access dashboards and submit assignments on smartphones or tablets.
- **Automated Notifications:** SMS or email alerts for assignments, attendance updates, and quizzes to improve engagement.
- **AI-Based Analytics:** Generate insights on student performance trends, predictive analysis, and personalized feedback.
- **Cloud Synchronization:** Enable multi-branch support for institutions with multiple centers, keeping all data synchronized in real-time.
- **Enhanced Reporting:** Visual dashboards and analytics to track overall academic performance and feedback trends.

REFERENCES

1. PHP Official Documentation – https://www.php.net/

2. MySQL Reference Manual – https://dev.mysql.com/doc/

3. MDN Web Docs – https://developer.mozilla.org/

4. W3Schools – https://www.w3schools.com/

APPENDIX

Note:

The source code included in this document represents a **working skeleton or simplified reference implementation** of the Sikshantra system. It is **not the complete original code** used in the full project but demonstrates the **core functionality and module structure**.

This reference code covers the essential modules of the system, including:

- **User Authentication:** Login and role-based access for Admin, Teacher, and Student.
- **Attendance Management:** Marking, storing, and retrieving student attendance.
- **Assignment Management:** Uploading assignments by teachers and submission tracking for students.
- **Feedback Collection:** Recording and retrieving feedback from students.
- **Timetable Management:** Adding and displaying class schedules.

The code is designed as a **starting point** to help understand the architecture and functionality of Sikshantra. It can be **extended and customized** to include additional features such as:

1. **Role-Based Dashboards:** Separate interfaces for Admin, Teachers, and Students with appropriate permissions.
2. **Quiz Module:** Online quiz creation, submission, and automatic evaluation.
3. **Graphical Analytics:** Visual representation of attendance and feedback using libraries like Chart.js.
4. **Notifications:** Email alerts or system notifications for assignments, attendance updates, or feedback reminders.
5. **Enhanced Security:** Implementation of password hashing, prepared statements, and input validation to prevent SQL injection and unauthorized access.

This reference code is intended for **educational and demonstration purposes**, illustrating the system's modular design and workflow. The **full project code** may include additional modules, optimized queries, user interface enhancements, and security measures that are **not fully reflected in this simplified skeleton**.

In essence, this skeleton code provides a **conceptual blueprint** for Sikshantra, showing how the modules interact and how data flows between the **frontend, backend, and database layers**. Developers can **adapt and extend it** to create a fully functional, production-ready system.

1. Database Connection (db.php)

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$database = "sikshantra";

// Create connection
$conn = new mysqli($servername, $username, $password, $database);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>
```

2. User Authentication (login.php)

```
<?php
session_start();
include 'db.php';

if (isset($_POST['login'])) {
    $email = $_POST['email'];
    $password = $_POST['password'];

    $query = "SELECT * FROM users WHERE email='$email' AND password='$password'";
    $result = $conn->query($query);

    if ($result->num_rows == 1) {
        $user = $result->fetch_assoc();
        $_SESSION['user_id'] = $user['user_id'];
        $_SESSION['role'] = $user['role'];

        if($user['role'] == 'Admin') {
            header("Location: admin_dashboard.php");
        } elseif($user['role'] == 'Teacher') {
            header("Location: teacher_dashboard.php");
        } else {
            header("Location: student_dashboard.php");
        }
```

```
    } else {
        echo "<script>alert('Invalid Email or Password');</script>";
    }
}
?>
```

HTML Form (login.html)

```
<form method="POST" action="login.php">
    Email: <input type="email" name="email" required><br>
    Password: <input type="password" name="password" required><br>
    <button type="submit" name="login">Login</button>
</form>
```

3. Attendance Module (attendance.php)

```
<?php
session_start();
include 'db.php';

if(isset($_POST['mark'])) {
    $student_id = $_POST['student_id'];
    $status = $_POST['status'];
    $date = date('Y-m-d');

    $query = "INSERT INTO attendance(student_id, date, status) VALUES ('$student_id','$date','$status')";
    if($conn->query($query) === TRUE){
        echo "Attendance marked successfully.";
    } else {
        echo "Error: " . $conn->error;
    }
}

// Display attendance for a student
$student_id = 101; // example
$result = $conn->query("SELECT * FROM attendance WHERE student_id='$student_id'");
while($row = $result->fetch_assoc()){
    echo $row['date']." - ".$row['status']."<br>";
```

```
}
?>
```

HTML Form (attendance.html)

```
<form method="POST" action="attendance.php">
    Student ID: <input type="number" name="student_id" required><br>
    Status:
    <select name="status">
        <option value="Present">Present</option>
        <option value="Absent">Absent</option>
    </select><br>
    <button type="submit" name="mark">Mark Attendance</button>
</form>
```

4. Assignment Module (assignment.php)

```
<?php
include 'db.php';


if(isset($_POST['upload'])) {
   $title = $_POST['title'];
   $deadline = $_POST['deadline'];
   $file = $_FILES['file']['name'];
   $tmp = $_FILES['file']['tmp_name'];


   move_uploaded_file($tmp, "uploads/".$file);


   $query = "INSERT INTO assignments(title, file, deadline) VALUES ('$title','$file','$deadline')";
   if($conn->query($query)){
      echo "Assignment uploaded successfully.";
   } else {
      echo "Error: " . $conn->error;
   }
}


// Display assignments
$result = $conn->query("SELECT * FROM assignments");
while($row = $result->fetch_assoc()){
```

```
    echo $row['title']." - ".$row['deadline']."<br>";
}
?>
```

HTML Form (assignment.html)

```
<form method="POST" action="assignment.php" enctype="multipart/form-data">
    Title: <input type="text" name="title" required><br>
    Deadline: <input type="date" name="deadline" required><br>
    File: <input type="file" name="file" required><br>
    <button type="submit" name="upload">Upload</button>
</form>
```

5. Feedback Module (feedback.php)

```
<?php
include 'db.php';


if(isset($_POST['submit_feedback'])) {
   $student_id = $_POST['student_id'];
   $subject = $_POST['subject'];
   $feedback = $_POST['feedback'];


   $query = "INSERT INTO feedback(student_id, subject, feedback) VALUES ('$student_id','$subject','$feedback')";
   if($conn->query($query)){
      echo "Feedback submitted successfully.";
   } else {
      echo "Error: " . $conn->error;
   }
}


// Display feedback for a subject
$result = $conn->query("SELECT * FROM feedback WHERE subject='CN'");
while($row = $result->fetch_assoc()){
   echo "Student ".$row['student_id']." - ".$row['feedback']."<br>";
}
?>
```

HTML Form (feedback.html)

```
<form method="POST" action="feedback.php">
    Student ID: <input type="number" name="student_id" required><br>
    Subject: <input type="text" name="subject" required><br>
    Feedback: <textarea name="feedback" required></textarea><br>
    <button type="submit" name="submit_feedback">Submit Feedback</button>
</form>
```

6. Timetable Module (timetable.php)

```
<?php
include 'db.php';


if(isset($_POST['add'])) {
    $day = $_POST['day'];
    $time = $_POST['time'];
    $subject = $_POST['subject'];


    $query = "INSERT INTO timetable(day, time, subject) VALUES ('$day','$time','$subject')";
    if($conn->query($query)){
        echo "Timetable added successfully.";
    } else {
        echo "Error: ".$conn->error;
    }
}


// Display timetable
$result = $conn->query("SELECT * FROM timetable ORDER BY day, time");
while($row = $result->fetch_assoc()){
    echo $row['day']." ".$row['time']." - ".$row['subject']."<br>";
}
?>
```

HTML Form (timetable.html)

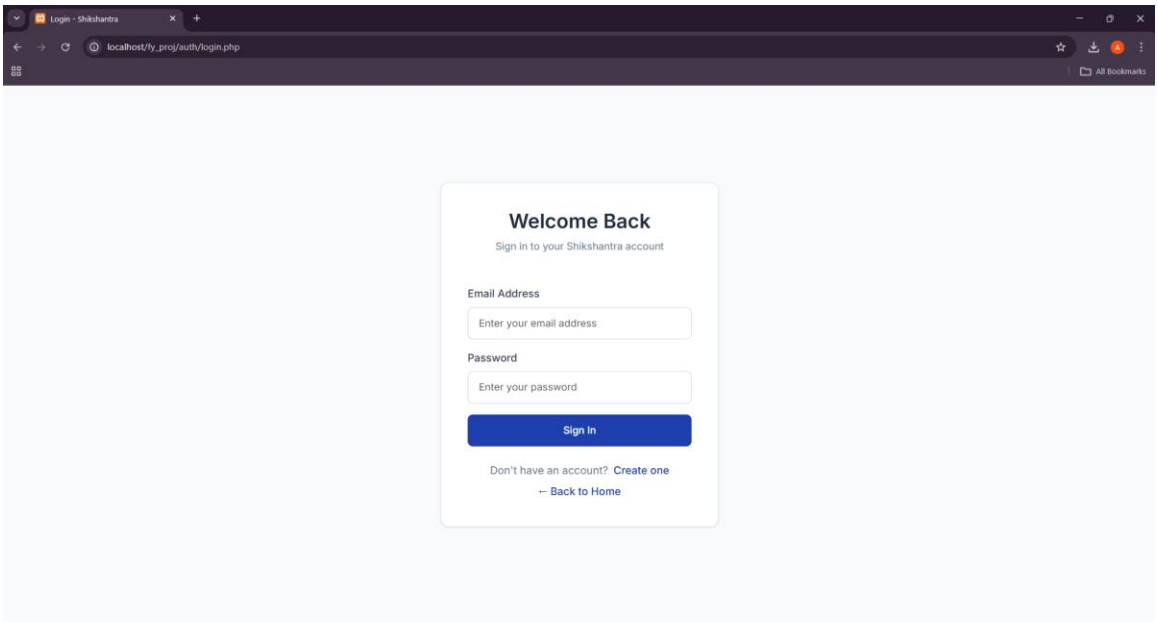
```
<form method="POST" action="timetable.php">
    Day: <input type="text" name="day" required><br>
    Time: <input type="time" name="time" required><br>
    Subject: <input type="text" name="subject" required><br>
    <button type="submit" name="add">Add</button>
</form>
```

7. Dashboard (dashboard.php example for Teacher)

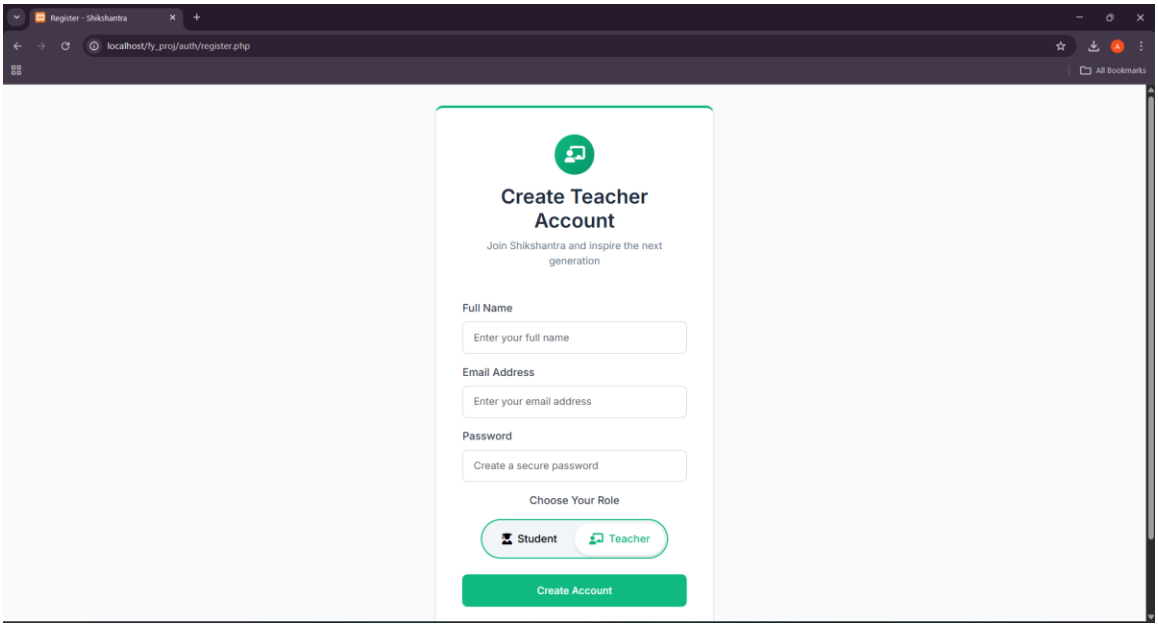
```
<?php
session_start();
if($_SESSION['role'] != 'Teacher'){
   header("Location: login.php");
   exit();
}
echo "<h1>Welcome Teacher</h1>";
echo "<ul>
       <li><a href='attendance.php'>Mark Attendance</a></li>
       <li><a href='assignment.php'>Upload Assignment</a></li>
       <li><a href='feedback.php'>View Feedback</a></li>
       <li><a href='timetable.php'>View Timetable</a></li>
     </ul>";
?>
```

SCREENSHOTS:

1)Login Page:

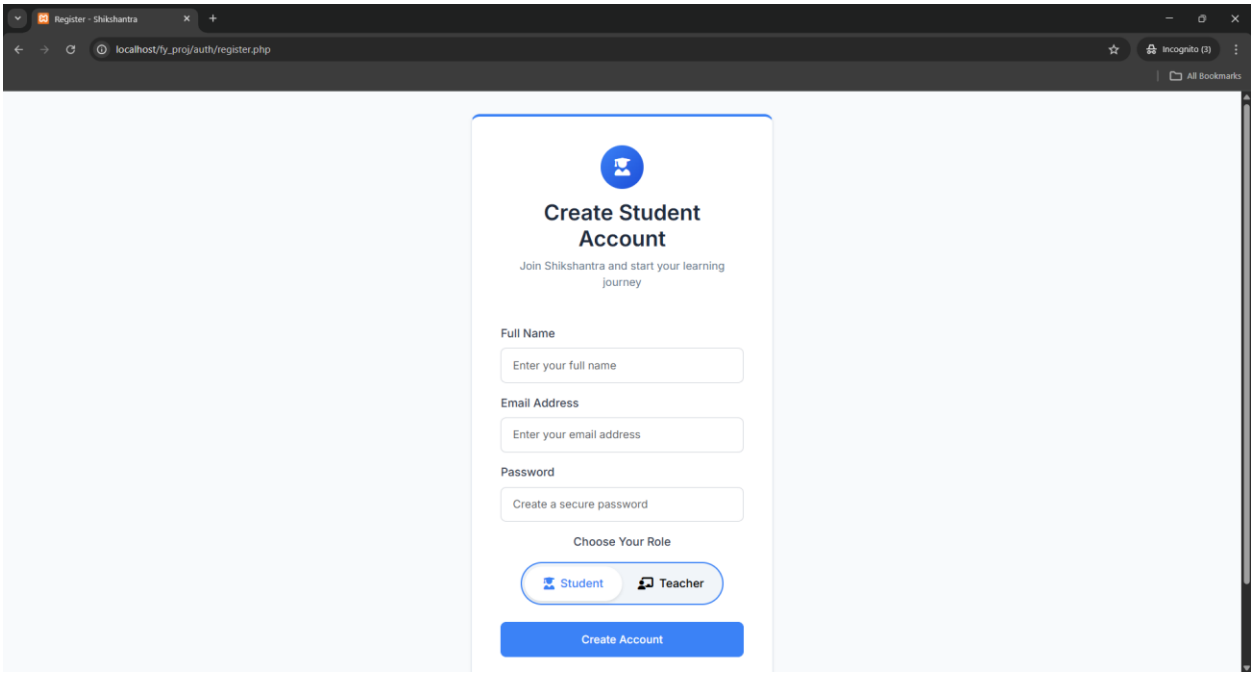


2)Register Page(Teacher):

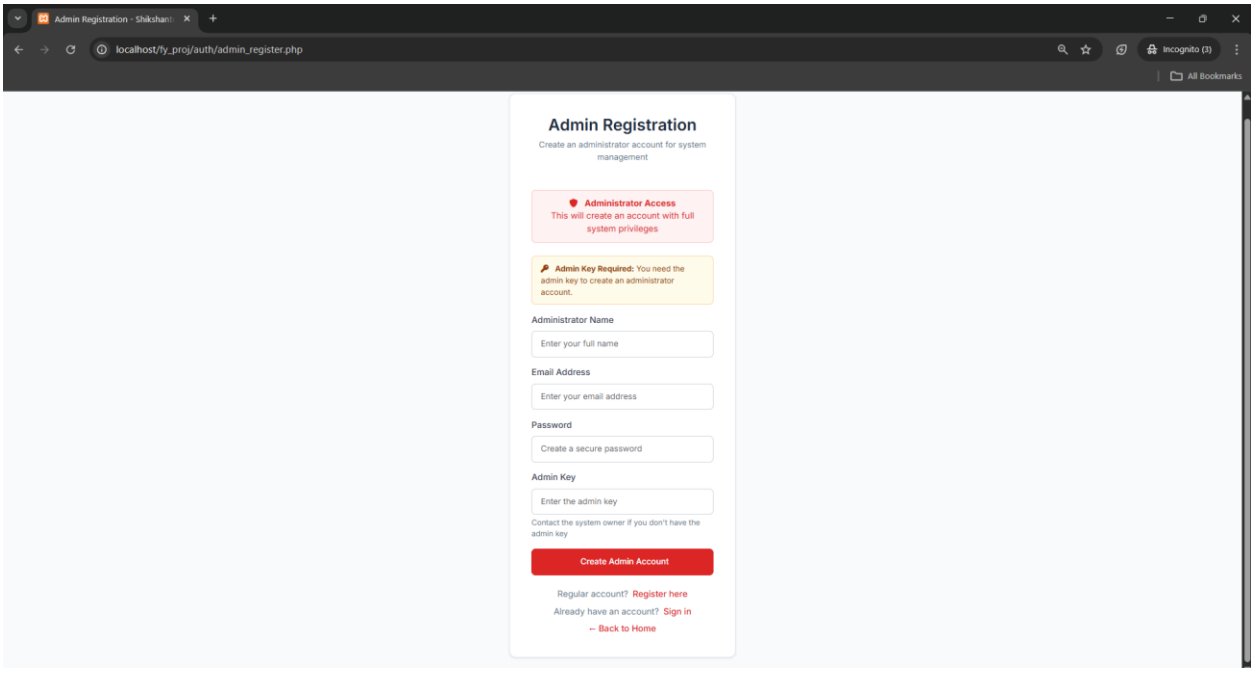


.

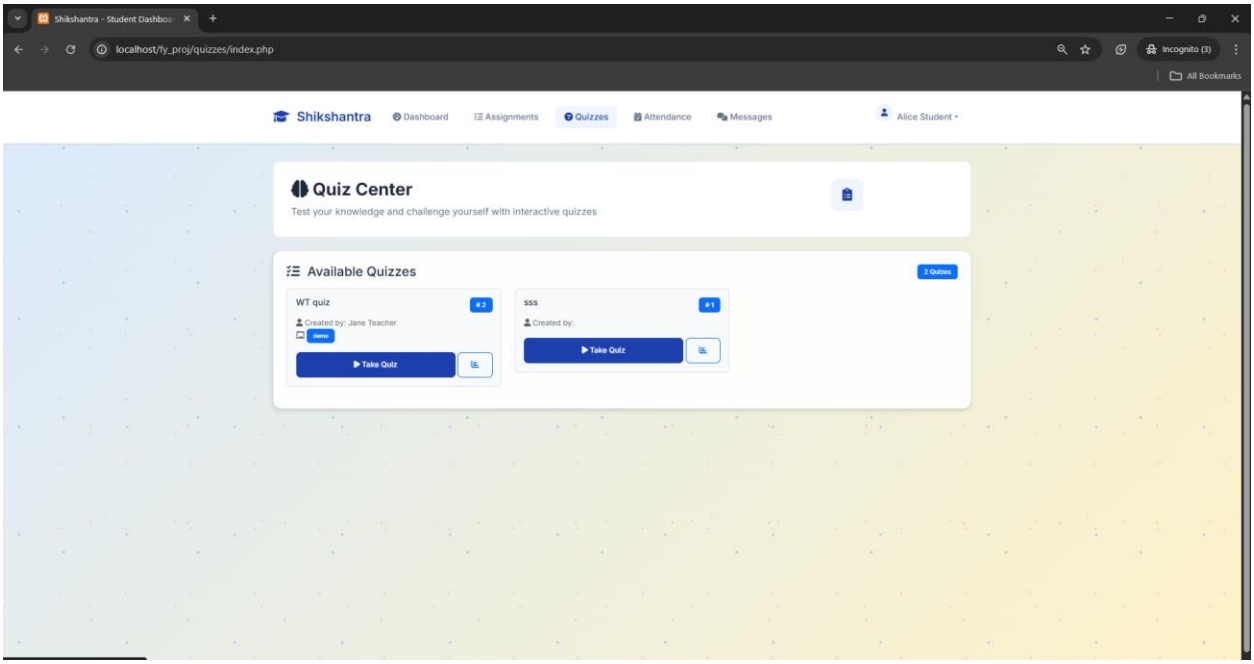
3)Register Page(Student):



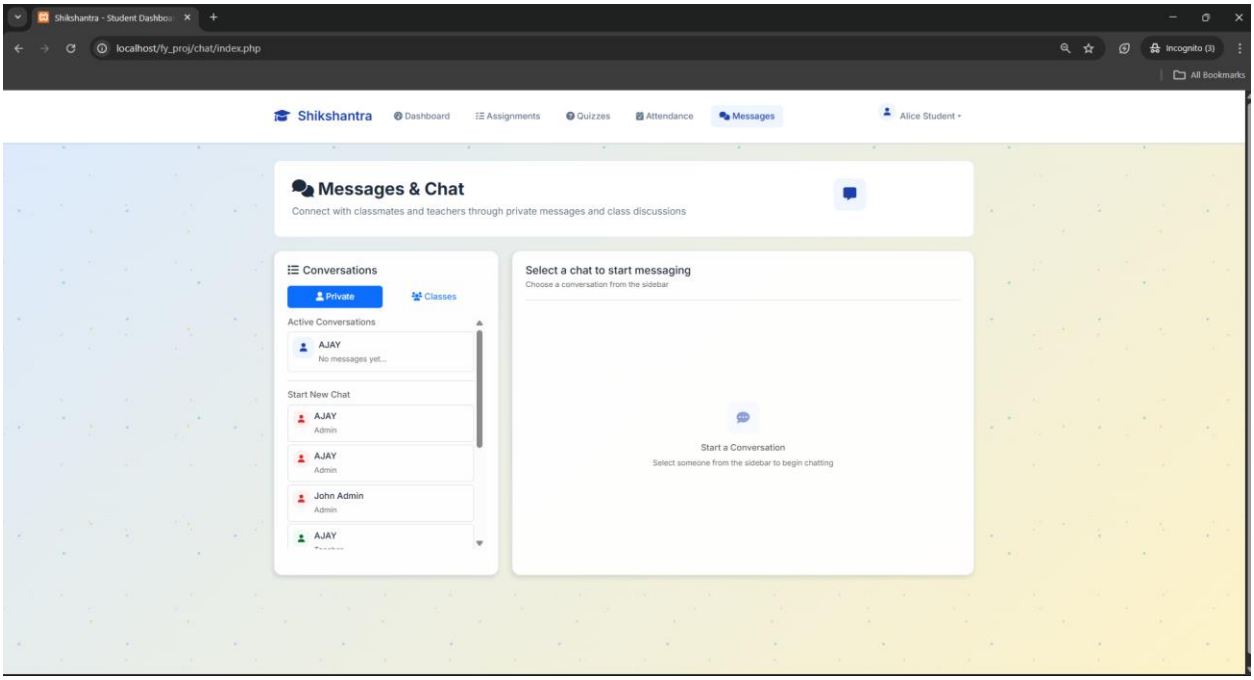
4)Register Page(Admin):



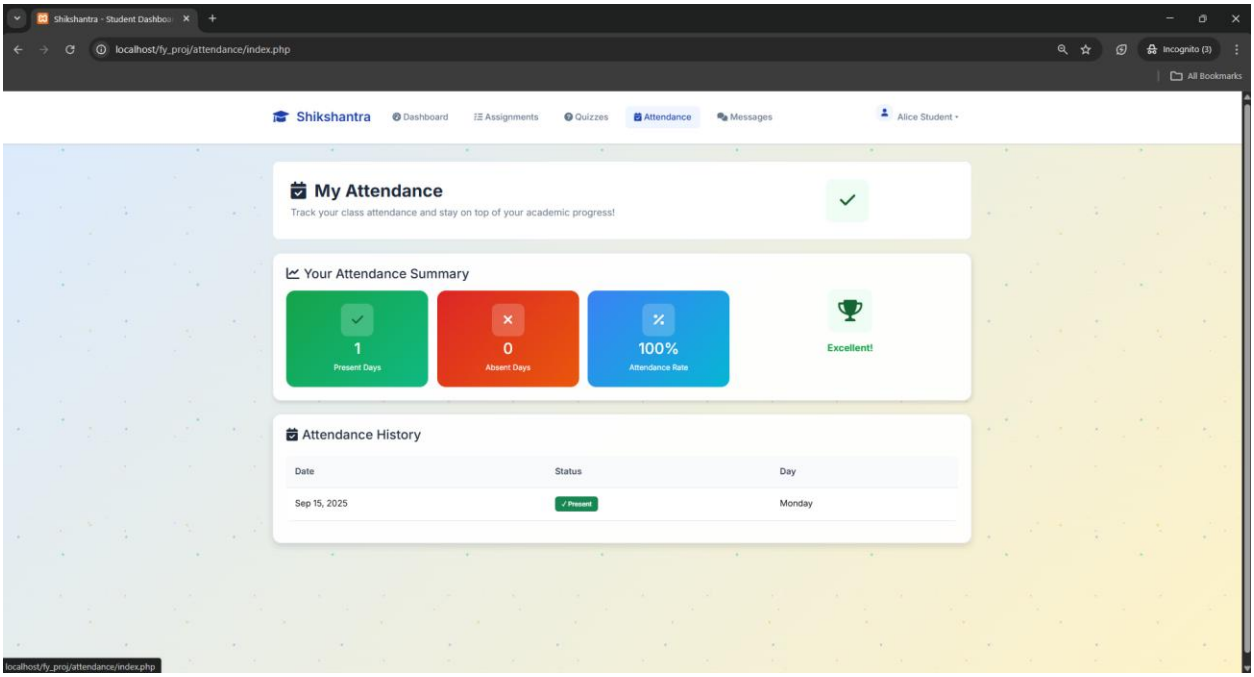
5)Quizzes Page(Student):



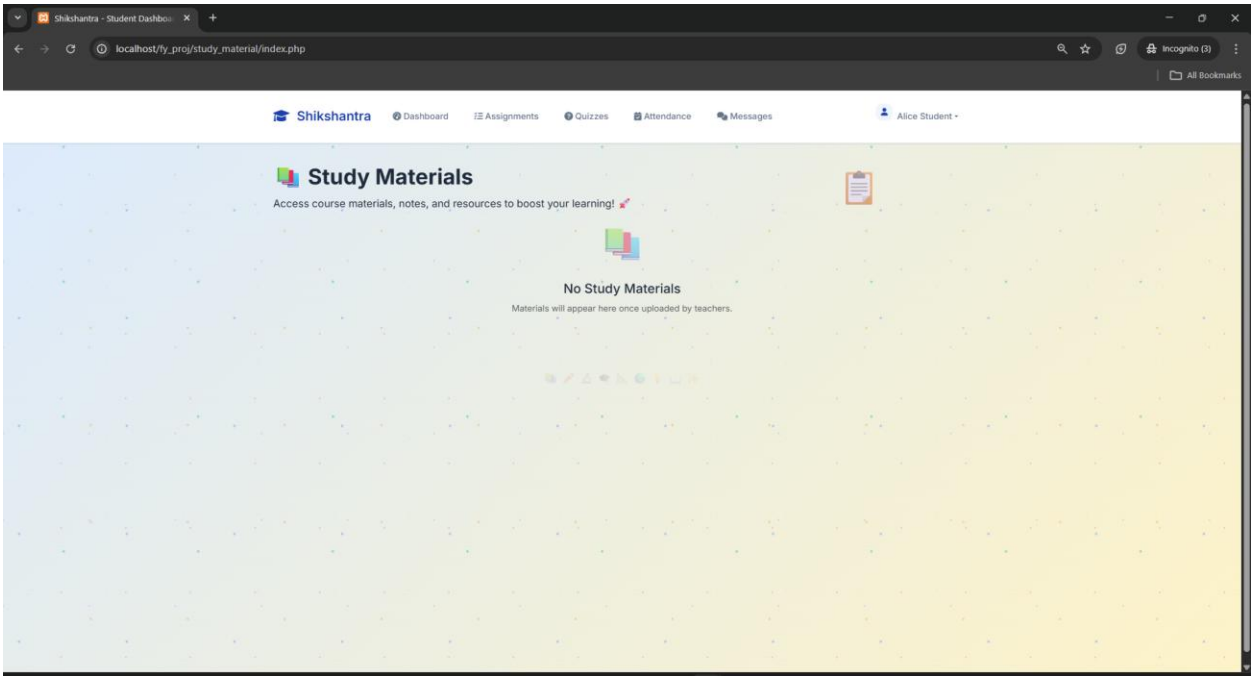
6)Messages(Student):



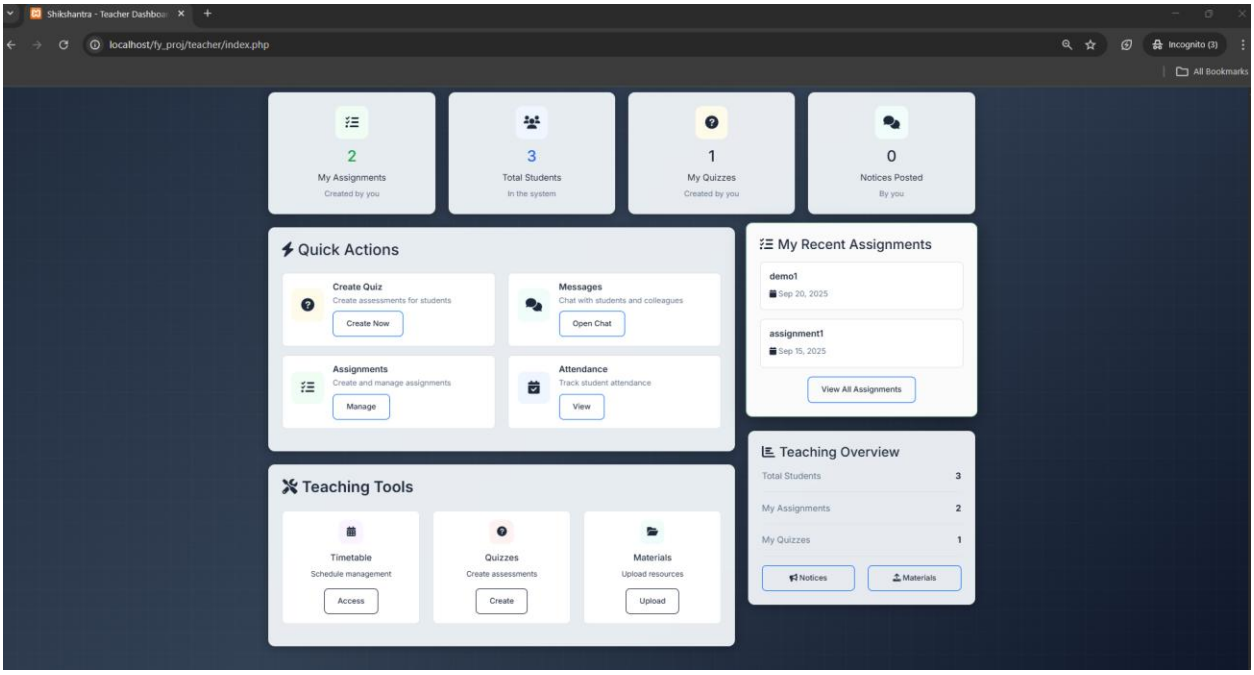
7)Attendance Status(Student):



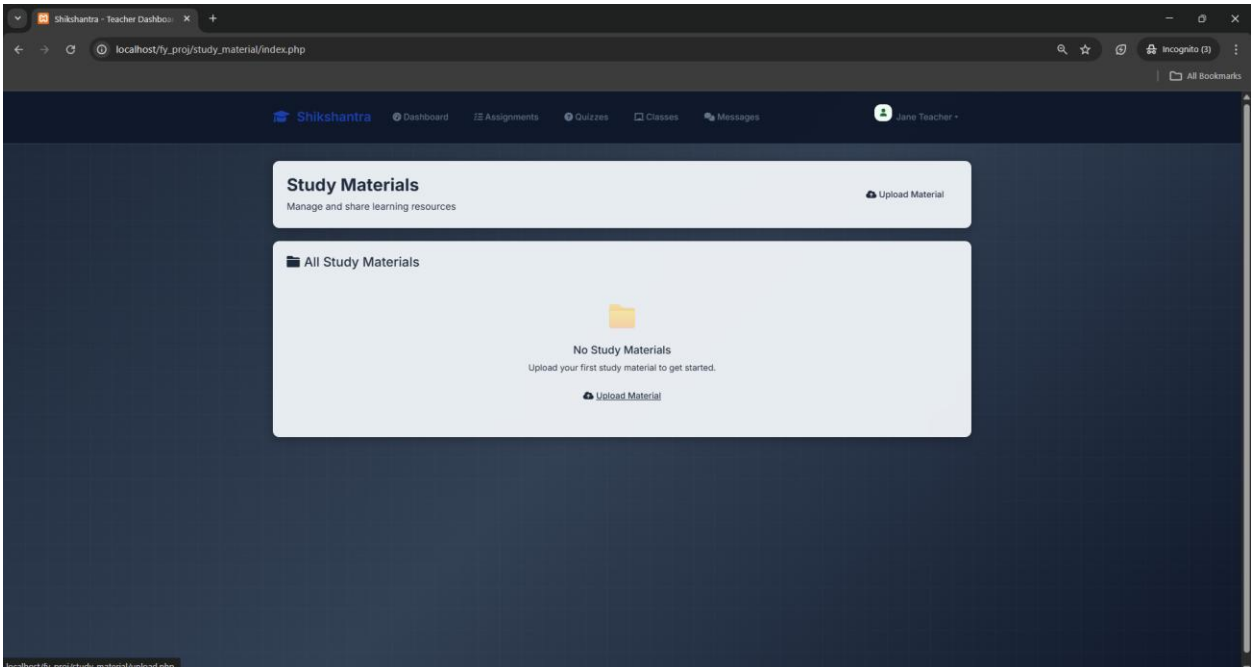
8)Study Material:



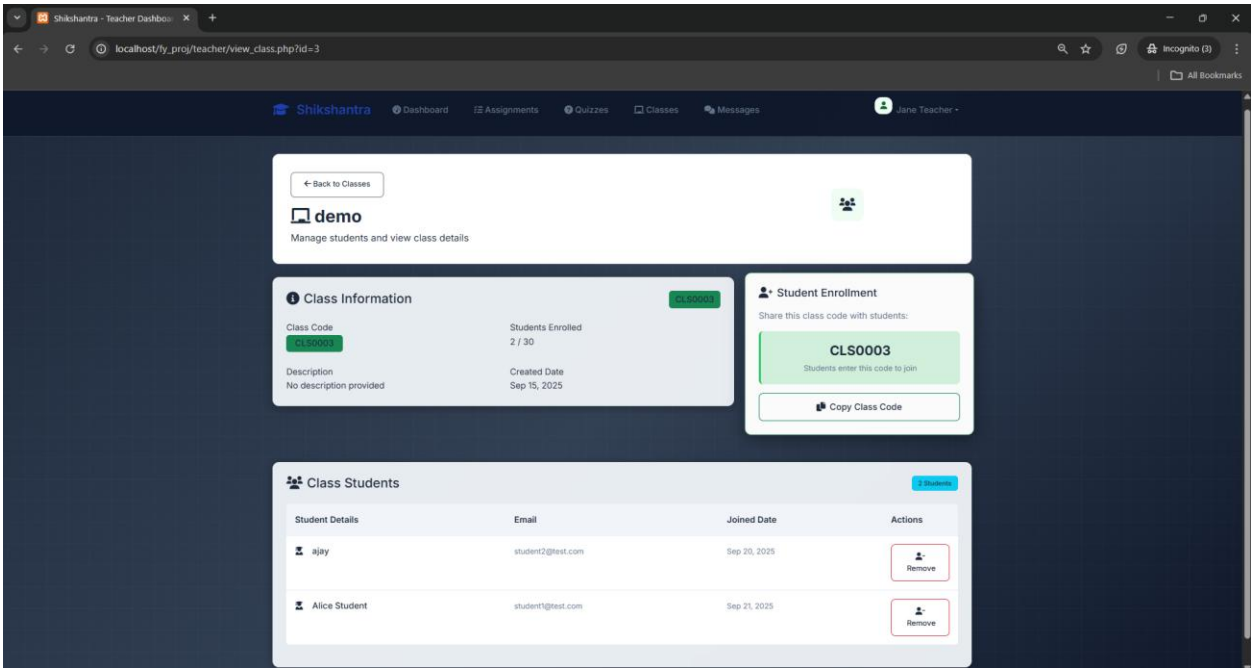
9)Teacher Dashboard:



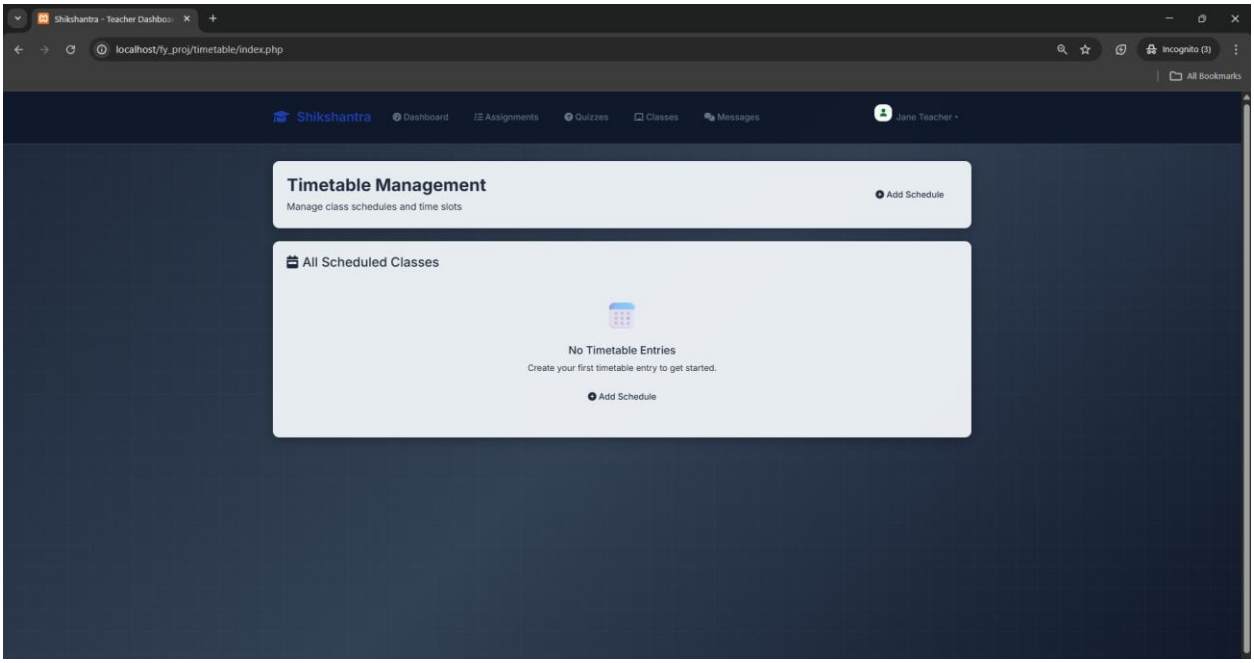
10)Teacher Study Material Upload:



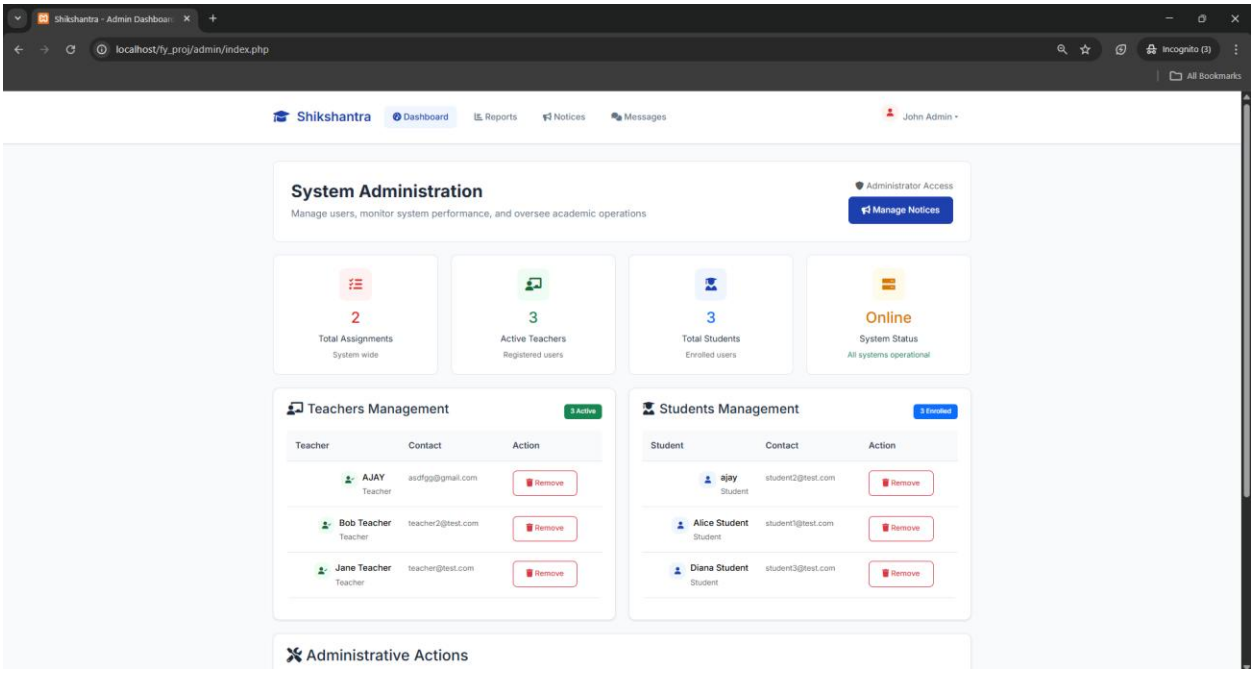
11)Teacher Single class:

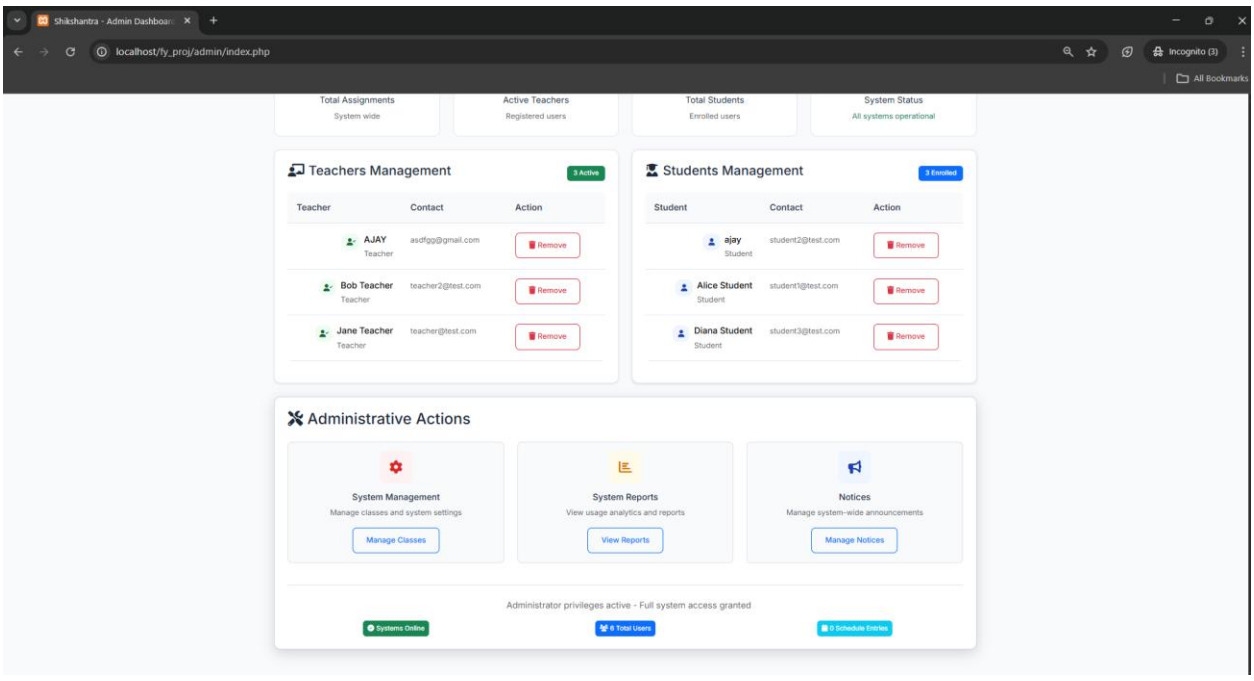


12)Timetanle Management(Teacher):

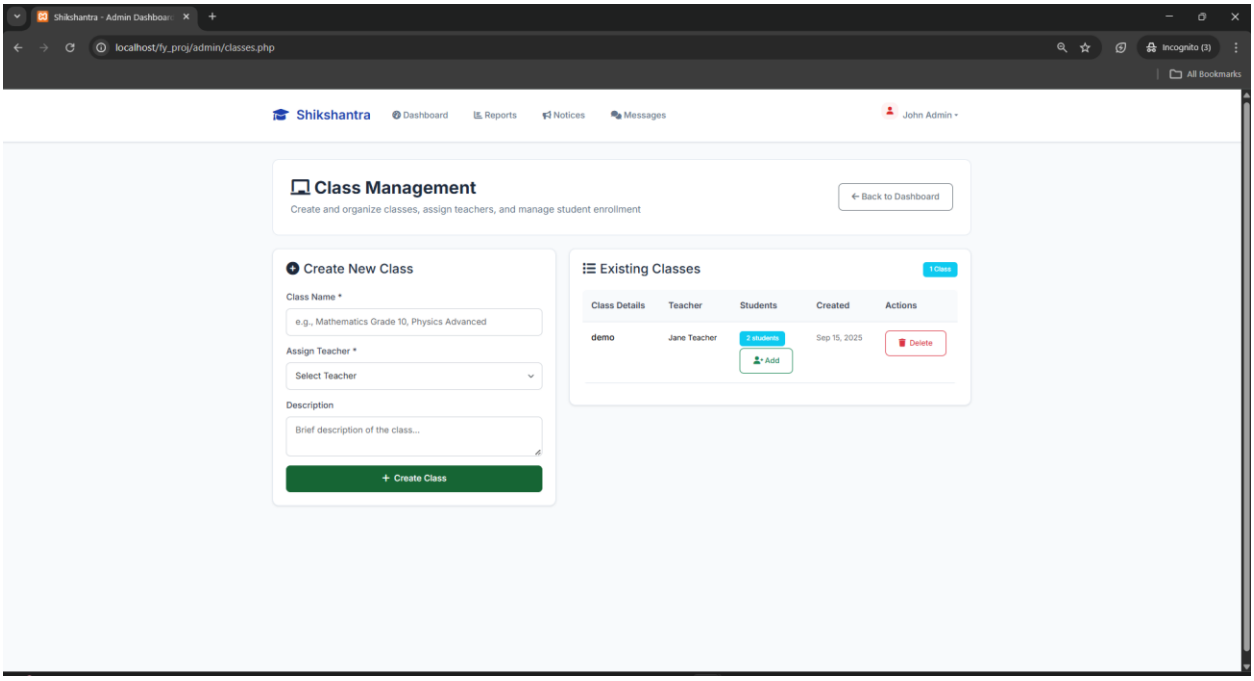


13)Admin Dashboard:





14)Class Management (Admin):


Shikshantra - Admin Dashboard
localhost/fy_proj/admin/classes.php
Incognito (3)
All Bookmarks
Shikshantra
Dashboard
Reports
Notices
Messages
John Admin
Class Management
Create and organize classes, assign teachers, and manage student enrollment
← Back to Dashboard
Create New Class
Class Name *
e.g., Mathematics Grade 10, Physics Advanced
Assign Teacher *
Select Teacher
Description
Brief description of the class...
+ Create Class
Existing Classes
1 Class
Class Details
Teacher
Students
Created
Actions
demo
Jane Teacher
2 students
Add
Sep 15, 2025
Delete