

RFID-Driven Attendance and In-Campus Payment

System for Enhanced Student Experience

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Computer Science

by

Hossam Mohamed

Faculty of Computer Science

October University for Modern Sciences and Arts (MSA)

Egypt

July 2025

Advisor: Dr. Moataz Samy

# Abstract

The project proposes an RFID system to enhance the students' experience, as well as operational efficiency, in the learning centers. Traditional in-campus payment and attendance practices are usually cumbersome, prone to errors, and do not permit monitoring in real-time. The proposed solution is founded on the utilization of RFID technology to enable automatic attendance monitoring as well as secure non-cash payment in the in-campus centers.

The MERN stack (MongoDB, Express.js, React, Node.js) system features role-based access among students, teachers, parents, and admins. The system features RFID integration modules, wallet management, in-built data logs, attendance and financial transaction dashboards, even a machine learning module that translates attendance data into predicting attendance trends in advance to facilitate proactive decision-making.

The system promotes accountability and transparency because it provides the parent with direct, in-time access to the financial and learning record of the child. The administrators benefit from centralized control of their data, while the students can check their spending and attendance.

The entire project is compliant with smart campus initiatives and is a scalable, secure, and effective solution to institution workflow modernization leveraging automation, data analysis, and customer-centric design.

# Acknowledgments

My sincere gratitude to Dr. Moataz Samy for the ongoing guidance and encouragement throughout the project.

His leadership and inspiration helped in the project's completion.

I would also extend gratitude to the Faculty of Computer Science, MSA

University, for the necessary infrastructure and facilities to carry out the work. Finally, sincere gratitude to family and friends for their inspiration and support along the way.

Table of Contents

[Abstract 2](#_Toc201470862)

[Acknowledgments 3](#_Toc201470863)

[List of Tables 6](#_Toc201470864)

[List of Figures 7](#_Toc201470865)

[Chapter 1: Introduction 8](#_Toc201470866)

[1.1 Project Objective 8](#_Toc201470867)

[1.2 Project Scope 9](#_Toc201470868)

[1.3 System Approach Overview 10](#_Toc201470869)

[1.4 Business Context 11](#_Toc201470870)

[1.5 Significance of the Study 12](#_Toc201470871)

[Chapter 2: Background 13](#_Toc201470872)

[2.1 Significance of Attendance and Financial Systems in Schools 13](#_Toc201470873)

[2.2 Traditional Learning Environment Barriers 13](#_Toc201470874)

[2.3 Recent Work and Contemporary Solutions 14](#_Toc201470875)

[2.4.Role of Simulation and Automation 15](#_Toc201470876)

[Chapter 3: Software Requirement Specification (SRS) 16](#_Toc201470877)

[3.1 Overall Summary 16](#_Toc201470878)

[3.2 Functional Requirements 17](#_Toc201470879)

[3.3 Use Case Diagram 21](#_Toc201470880)

[3.4 Initial Object-Oriented Domain Analysis 22](#_Toc201470881)

[3.5 Interface Requirements 23](#_Toc201470882)

[3.6 Performance Requirements 24](#_Toc201470883)

[3.7. Constraints 25](#_Toc201470884)

[3.8 Other Non-Functional Requirements 25](#_Toc201470885)

[3.9 Early Routine 25](#_Toc201470886)

[Chapter 4: System Implementation 26](#_Toc201470887)

[4.1 System Overview 26](#_Toc201470888)

[4.2 Frontend Architecture 27](#_Toc201470889)

[4.3 Backend Architecture 28](#_Toc201470890)

[4.4 Real-Time Communication (Socket.IO) 29](#_Toc201470891)

[4.5 RFID Simulation 30](#_Toc201470892)

[4.6 Scheduled Task and CRON Jobs 31](#_Toc201470893)

[4.9 Implementation Highlights 31](#_Toc201470894)

[Chapter 5: Testing and Evaluation 33](#_Toc201470895)

[5.1 Introduction 33](#_Toc201470896)

[5.2 Functional Testing 33](#_Toc201470897)

[5.3 Performance Testing 34](#_Toc201470898)

[5.4 Evaluation Summary 34](#_Toc201470899)

[Chapter 6: Conclusion, Lessons Learned & Problems Faced 35](#_Toc201470900)

[6.1 Conclusion 35](#_Toc201470901)

[6.2 Lessons Learned 35](#_Toc201470902)

[6.3 Problems Faced 36](#_Toc201470903)

[6.4 Summary 36](#_Toc201470904)

[Appendix A 37](#_Toc201470905)

[Bibliography 42](#_Toc201470906)

List of Tables  
[Table 1: Project Development Timeline (Overview of project phases from requirements gathering to deployment) 26](#_Toc201470720)

[Table 3: Database Collections and Fields (Summary of MongoDB collections and key attributes) 28](#_Toc201470721)

[Table 2: User Roles and Access Permissions (Mapping of actions per role: Admin, Student, Teacher, etc.) 28](#_Toc201470722)

[Table 4: ML Prediction Accuracy 31](#_Toc201470723)

[Table 5:API Endpoints and Methods 32](#_Toc201470724)

[Table 6: Sample Attendance Record 33](#_Toc201470725)

[Table 7: Sample Attendance Record 34](#_Toc201470726)

List of Figures  
[Figure 1: Full System Architecture Overview 11](#_Toc201450424)

[Figure 2: Login and Role-Based Access 17](#_Toc201450425)

[Figure 3: Real-Time RFID Flow (Socket.IO) 19](#_Toc201450426)

[Figure 4: Class Diagram 23](#_Toc201450427)

[Figure 5: Full System Architecture 26](#_Toc201450428)

[Figure 6: Login and Role-Based Access 27](#_Toc201450429)

[Figure 7: Real-Time RFID Flow Using Socket.IO 29](#_Toc201450430)

[Figure 8: Screenshot for trained model for the Attendance prediction 35](#_Toc201450431)

[Figure 9: Screenshot from training model for the Spending prediction 36](#_Toc201450432)

[Figure 10: Login Page - Role Based Access 37](#_Toc201450433)

[Figure 11: Admin Dashboard 37](#_Toc201450434)

[Figure 12: Admin Users CRUD 38](#_Toc201450435)

[Figure 13: Admin Courses CRUD 38](#_Toc201450436)

[Figure 14: Admin Lectures 39](#_Toc201450437)

[Figure 15: Admin Transactions 39](#_Toc201450438)

# Chapter 1: Introduction

The trend of the smart campus is revolutionizing the functioning of the campuses. Manual attendance monitoring and secure payment methods in most cases are ineffective, time-consuming, and do not provide real-time insight. The project suggests a web-based, RFID-supported system that replicates attendance monitoring and secure, cashless payment facilitation in an educational environment. Automating the process via role-based log-in and replicating RFID scans, the system sets the bar for operational efficiency, transparency, and ease of use a step higher.

## 1.1 Project Objective

The project's objective is to create an RFID system based on simulation that integrates automated attendance and payment using a digital wallet. Unlike using actual RFID sensors, the system mimics RFID capability through interactive elements that simulate the process of tag scans. This enables the possibility of accurate recording of the records, live monitoring, as well as access control without the need to use specialized hardware.

The system is designed to serve a set of users within the institution who play different roles and have different levels of access:

* Level 1 Administrator – Full control of all users, settings, analytics, and back-end functionality

Level 2 Administrator (Faculty Dean) - Manages faculty-level operations, assignment of users, and reporting

* Level 3 Admin (Vice Dean) - Manages chosen academic information and supports reporting by faculty
* Teacher – Responsible for attendance sheets and timetables of students
* Student – Interacts with the RFID payment and attendance simulation; views personal statistics
* Parent - Monitors child's attendance, spending, and wallet activity in real-time
* Merchant – RFID-based payment acceptance and transaction log viewing

## 1.2 Project Scope

The objective of the project is to develop a fully software-simulated RFID system.

The project scope is as under:

Attendance Tracking: The students perform mock RFID card swipes for attendance tracking.

* Digital Payment: Students make payments for school services using digital wallets.
* Role-Based Dashboards: Different users from various categories view a role-based tailored interface.

•Admin Management: Multi-level admin controls to handle users, transactions, and analytics

* Parental control: Real-time web portal for viewing attendance and spending.
* Predictive Features: Basic analytics that identify attendance or wallet patterns.

Beyond the scope

* Physical RFID readers, sensors, or hardware integration
* External payment gateways or bank APIs

## 1.3 System Approach Overview

The platform is an online simulation system developed using the MERN stack:

• MongoDB stores attendance histories, wallet balances, and the users' details

Express.js and Node.js manage back-end logic and simulate RFID responses

React.js makes interactive interfaces accessible to everyone

Socket.IO/Web APIs facilitate real-time communication between components

Simulation buttons or events mimic the process of RFID scans without using physical devices.

#### 1.3.1 System Architecture

The system is client-server modular in structure:

* Frontend: Provides customized dashboards for students, parents, teachers, admins, and merchants
* Back-end: Responsible for handling RFID logic, data, authentication, and wallet features

•Database: Stores users' accounts, attendance reports, and transactional history

* Simulation Layer: Simulates the RFID scans with virtual elements (i.e., “Scan” buttons)

Its design makes the project testable, extendable, and hardware-independent, thereby apt to be employed in simulation environments as well as in-class demonstrations.

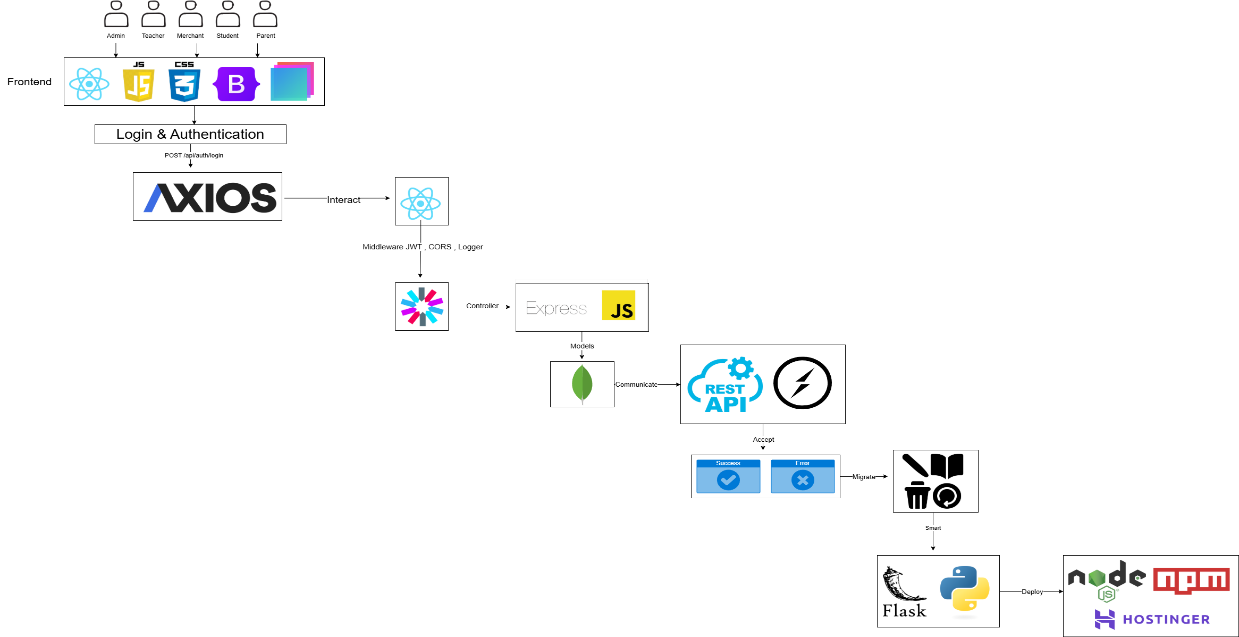


Figure 1: Full System Architecture Overview

## 1.4 Business Context

Schools and universities need to make their process simpler, lower their spending, and boost engagement. Manual attendance books and the payment of money are a thing of the past. On top of that, demand for data openness and parental engagement keeps growing.

It serves these needs directly by:

* Examining RFID-based services in an education setting
* Offering digital wallets and dashboards
* Providing actionable data to administrators and guardians

The modular strategy also leaves options open for any future integration, such as biometric scanning, mobile coverage, or even physical RFID hardware if required.

## 1.5 Significance of the Study

This study determines that schools can benefit from digitalization through simulation. It proves that problems in the real world such as inefficient attendance monitoring and insecure campus transactions can be solved by utilizing well-designed software systems without the installation of physical sensors or expensive hardware.

Seven different kinds of users bring in practicality as well as the hierarchy of actual scholastic entities. The project promotes operational realism and system scalability in design because of the workflow and permission emulation.

Ultimately, the system offers an economy- and educationally effective template for replicating smart campus operations.

# Chapter 2: Background

In the chapter, the fundamental principles, reasons, and problems that justified the installation of an RFID-based attendance monitoring and cashless payment system in the institutes of learning are discussed. These are founded on the limitations of the conventional approach, the growing need for intelligent infrastructure in the campuses, and the success of such a system in an educational environment.

## 2.1 Significance of Attendance and Financial Systems in Schools

Attendance and in-college transactions are fundamental functions of any school operation. Accurately monitoring attendance is essential to academic integrity, grading, and compliance with policies. Similarly, processing transactions -- from cafeteria food purchase to library charges to student parking -- is a function of a student's life.

These functions were once dependent on paper-based or manual mechanisms, which are prone to human error and are tedious. As digital infrastructure increases, institutions now turn to the use of integrated mechanisms that automate processes, offer instantaneous monitoring, and offer data to aid in decision-making.

## 2.2 Traditional Learning Environment Barriers

Based on case studies and observations, a number of problems plague standard attendance and payment systems:

* Manual Attendance Records: Have no real-time updates and are susceptible to erroneous entries or tampering.
* Cash Transactions: Mismanagement of risk, theft, and insufficiency in recording finances.
* Excessive administrative burdens: Manual reporting, reconciliations, and entries add to employees' workload and expense.
* Limited Parental Guidance: Parents likely do not know students' attendance patterns or spending patterns.
* No Personal Budgeting Tools: Students do not receive organized financial instructions.

•Data Fragmentation: Schools do not have centralized systems to maximize services based on students' preferences.

These challenges necessitated the need for an automated, role-based system capable of delivering consistent, real-time intelligence.

## 2.3 Recent Work and Contemporary Solutions

Several institutions have explored RFID-based systems to bring automation to campus services. They are:

• An IT centre of a university implemented attendance monitoring using RFID and attained 128.57% accuracy, time savings, and student engagement (Okoli, 2022).

•A case study also found that the implementation of RFID-based cashless technology sped up the speed of transactions by 40% and reduced the frequency of financial errors by 25% (Shinde, 2024).

While they function, they are physical RFID technology, a costly and difficult-toscale technology to use in research projects. Commercial ones also do not accommodate a range of users such as vendors, faculty deans, or parents to name a few.

Your system is founded on these findings but performs the same function of RFID in software, the same learning outcomes, and the same operational benefits - without cards or actual RFID readers.

## 2.4.Role of Simulation and Automation

The standout innovation of the project lies in the use of a simulated RFID capability instead of actual sensors. Using a simulated RFID interface in a MERN web application, the system provides a hardware-independent and riskless space to:

Simulate scans of RFID cards during transactions and attendance.

• Provide instant responses to all users.

Represent and capture institution data with utmost flexibility Enable testing, demonstration, and scholarly scalability.

This solution supports the underlying aims of developing a smart campus without the costs or logistics of hardware. It also offers a platform for gaining experience in learning automation ideas and full-stack integration in a platform that is software alone.

# Chapter 3: Software Requirement Specification (SRS)

The chapter defines the overall software requirements of the suggested RFID-based attendance and cashless payment system in a school. System architecture, system users, the system's most important functionalities, the system's interface components, and system constraints are discussed in the chapter.

## 3.1 Overall Summary

#### 3.1.1 Product Perspective

It's an RFID web application that functions on the principle of simulation using the MERN stack technology (Node.js, Express.js, React.js, and MongoDB). It mimics the process of RFID card scans by simulating UI interactions. Software-only based, as opposed to the typical case of having physical sensors. It's for institutes that want to monitor attendance and transaction in a multi-role setting through software alone.

#### 3.1.2 Product Functions

* Simulate RFID-based attendance scans.
* Simulate RFID wallet payment.

•Implement CRUD operations among users, courses, schedules, wallets, and transactions

* Create visual dashboards and exportable reports

•Enable live parental monitoring

* Provide role-based tailored dashboards to users

## 3.2 Functional Requirements

#### 3.2.1 User Authentication

* They can be registered or be managed by admins.
* Users log in with email and password
* Passwords are securely stored using hashing (bcrypt)
* Sessions are authenticated using JSON Web Tokens (JWT)
* Role-based redirecting after login
* Logout security and password reset

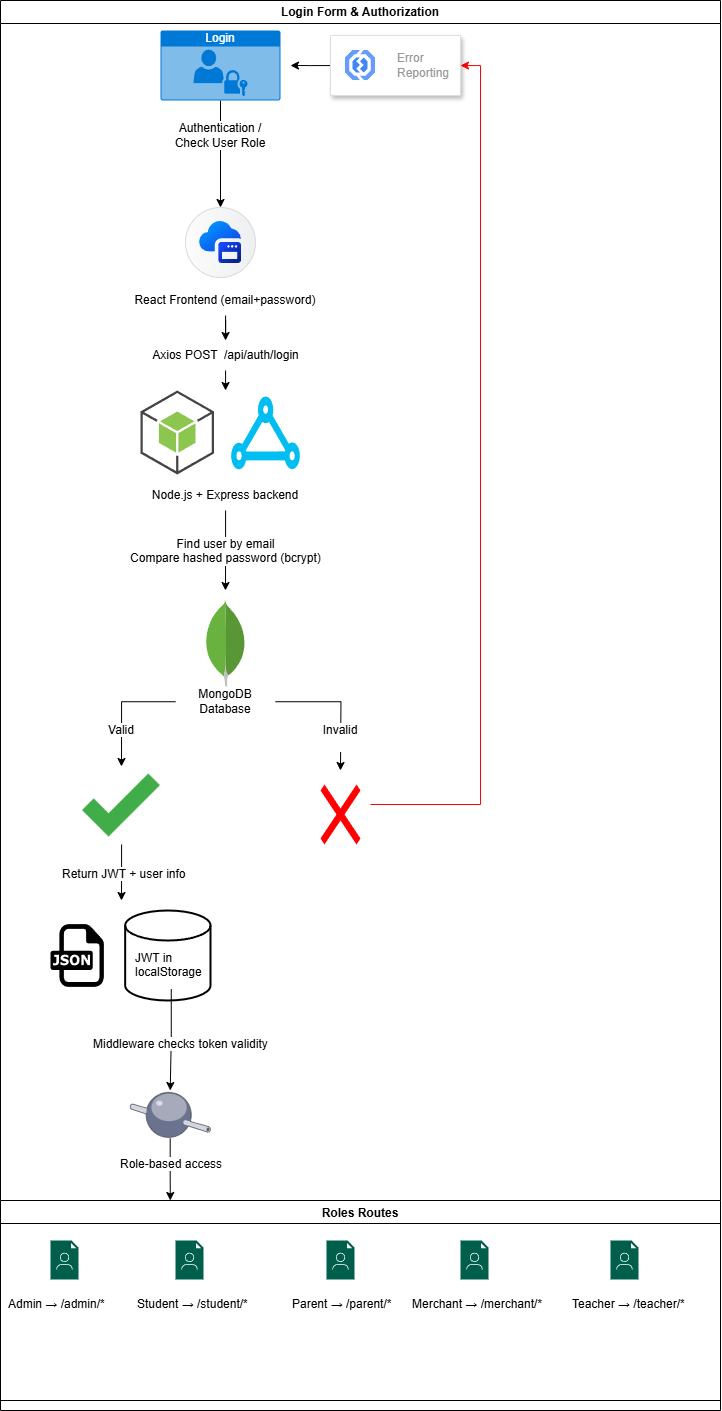


Figure 2: Login and Role-Based Access

#### 3.2.2 RFID Attendance Simulator

•They can simulate RFID scans using a button in their dashboard

* Logs course, timestamp, and student ID
* Attendance is automatically marked as (Present, Late, Absent)
* Subject teachers can also see attendance by date/course.
* Admins can edit or override attendance logs
* Export option in PDF and CSV modes

#### 3.2.3 RFID-based in-campus payment

* Students test payment by "swiping" their student ID into a merchant terminal
* Wallet has been confirmed and deducted accordingly

•Student, merchant, product, quantity, timestamp are held in the transaction histories.

•Merchants view live receipts

* Parents can view categories of transactions
* Admins can process a refund or report anomalies

#### 3.2.4 User Management

* Administrators can add users.
* Admin can read/view users
* Admin can change the users' details.

•Administrators can remove users

#### 3.2.5 Course Management

* Admin can add new courses.
* Administrators view course details
* Administrators can edit course details
* Administrators can cancel courses

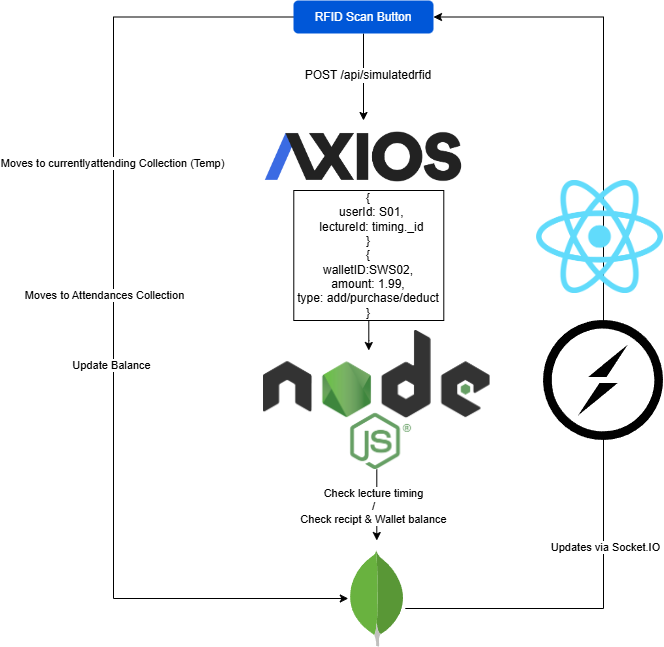


Figure 3: Real-Time RFID Flow (Socket.IO)

#### 3.2.6 Schedule Management

* Admin authorities can upload class schedules
* Administrators can see assigned schedules
* Administrative personnel can alter schedule details
* Administrators can remove schedules

#### 3.2.7 Wallet Management

* Admins or parents can initialize/create wallets
* Administrators can see wallet balances
* Admins may add/remove funds
* Parents may add funds
* Administrators can reset or remove a wallet

#### 3.2.8 Transaction Management

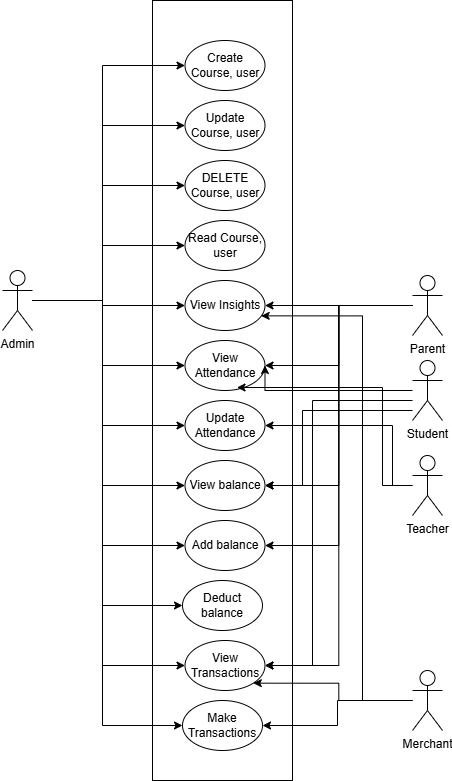
•Admin can see all the transactions

* Admins can filter transactions by user, role, or by date
* Admins may change transaction status, for example, refund issued.
* Admin can remove invalid or redundant records

#### 3.2.9 Reporting and Dashboard Access

* Students see attendance and wallet history
* Teachers view attendance reports for classes
* Parents view live monitoring of their child's activity
* Admins see platform-wide analytics based on graph filters
* PDF/Excel modes for exporting

## 3.3 Use Case Diagram



## 3.4 Initial Object-Oriented Domain Analysis

#### 3.4.1 Inheritance Relationships

* User (base) → Administrator, Educator, Learner, Parent, Merchant

•Transaction → AttendanceRecord, Payment

* Admin → Has full control over system objects

#### 3.4.2 Class Diagram

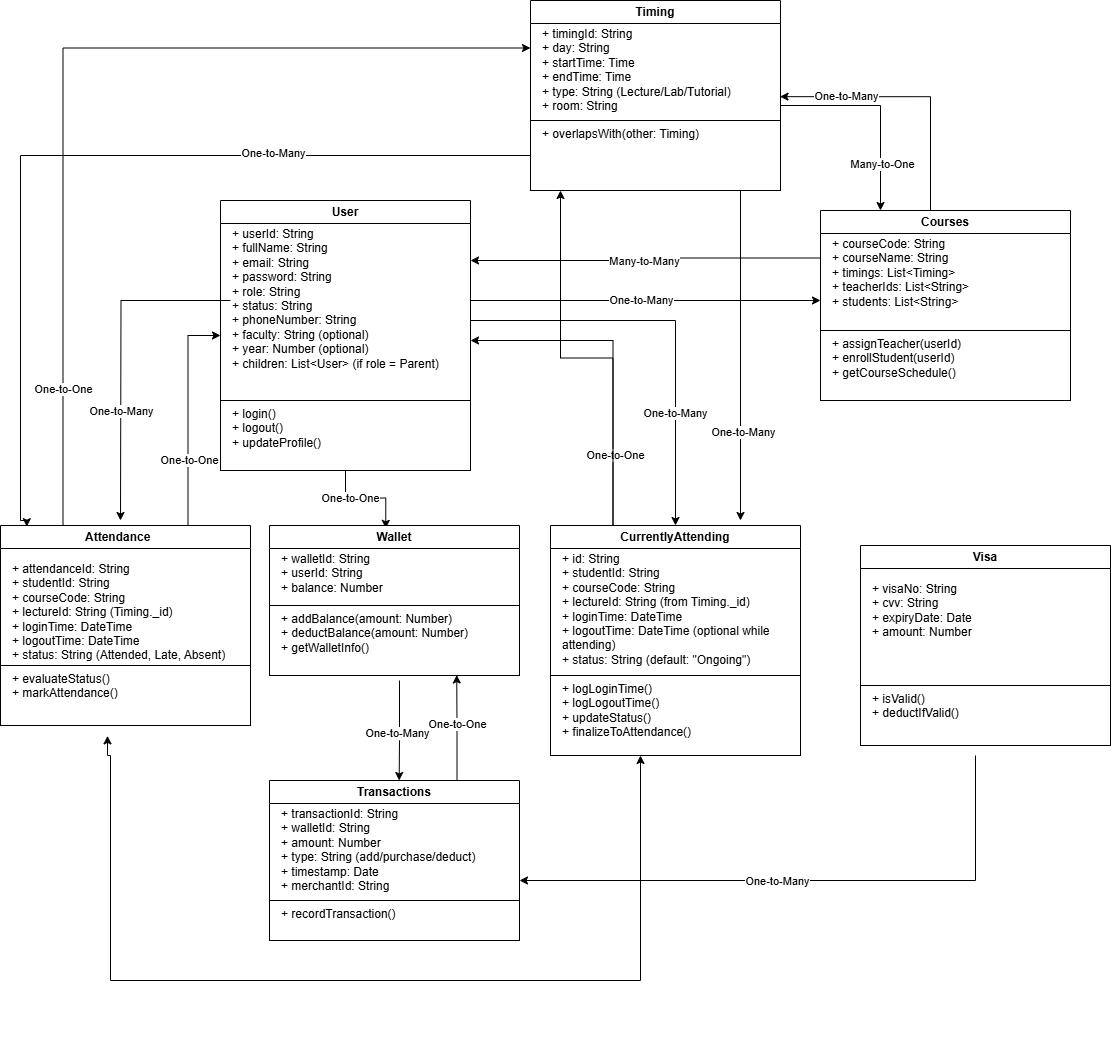


Figure 4: Class Diagram

## 3.5 Interface Requirements

#### 3.5.1 Graphical User Interface

React-based role-specific dashboards

* Scan the simulation buttons for RFID commands

•Tables and graphical representations for log display

* Real-time updates through WebSocket or polling

#### 3.5.2 Application Programming Interface (API)

All APIs are RESTful and use JWT authentication.

## 3.6 Performance Requirements

* Less than 2 seconds of simulated RFID events
* Wallet transactions executed in < 2 seconds
* Supports more than 100 users in a high-performing system
* 1-second average query time using MongoDB

## 3.7. Constraints

•Emulation of RFID capability by UI without the need of physical hardware

• Supports modern browsers (Chrome, Firefox, Edge)

•System vocabulary: English

## 3.8 Other Non-Functional Requirements

* Security: jwt authentication, bcrypt hash, enforced HTTPS
* Scalable backend, modular to the multi-faculty level.
* Reliability: Stable data recording with roll-back functionality in important modules
* Usability: Low learning process along with easy-to-use interfaces

## 3.9 Early Routine

*Table 1*

|  |  |
| --- | --- |
| Phase | Timeline |
| Requirement Gathering | 4 Weeks |
| System Design & Architecture | 5 Weeks |
| Development Phase | 12 Weeks |
| Testing & Refinement | 3 Weeks |

Table 1: Project Development Timeline (Overview of project phases from requirements gathering to deployment)

# Chapter 4: System Implementation

## 4.1 System Overview

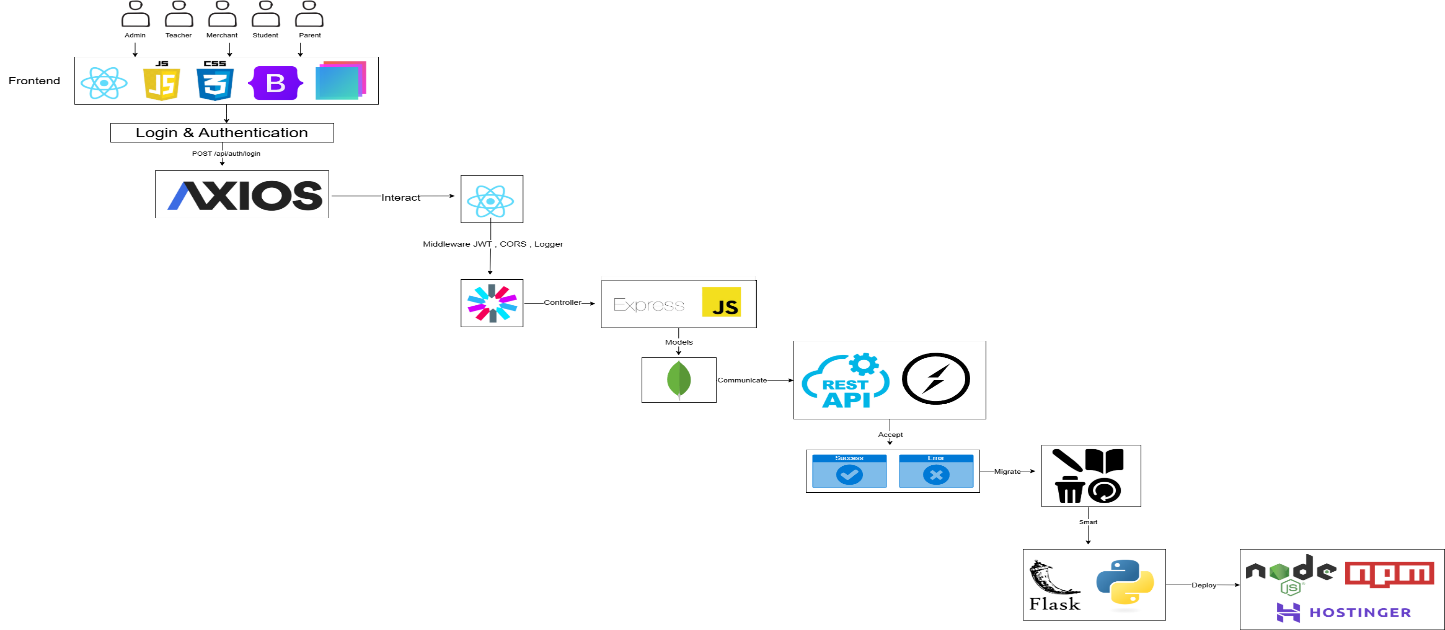
It's an MERN stack web application (MongoDB, Express.js, React.js, Node.js). The React frontend is bootstrapped with Create React App and has administrator, faculty dean, student, parent, and secretary role-based dashboards.

Figure 5: Full System Architecture

## 4.2 Frontend Architecture

Frontend is developed with React.js utilizing Create React App. with route-based layouts done with React Router, differentiating access and views by role. For instance:   
/admin uses AdminUsers to manage users, courses, lectures, and wallets.  
/student shows attendances, transactions, and RFID interactions.

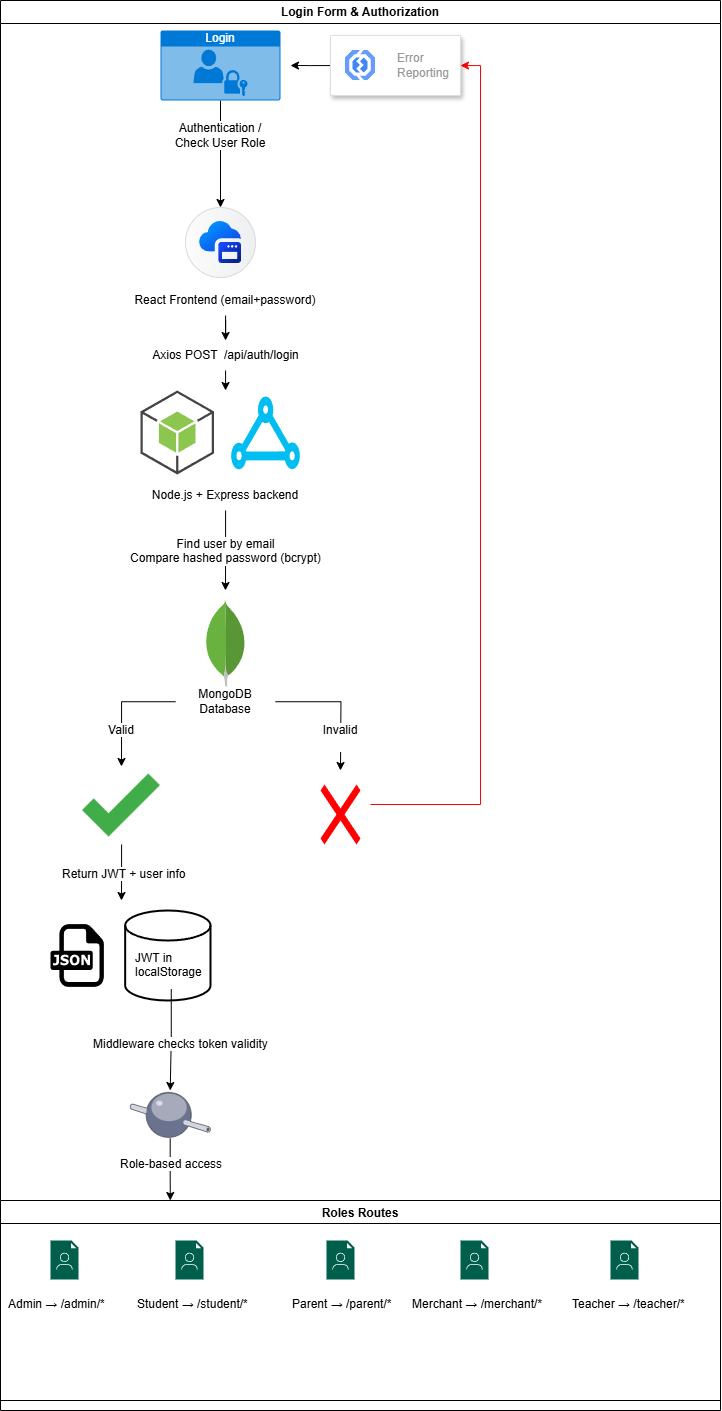


Figure 6: Login and Role-Based Access

There are cards, tables, and charts on every dashboard. Axios is used for REST API requests (obtaining lectures or sending RFID scan data, etc.).

## 4.3 Backend Architecture

Backend is constructed using Node.js and Express.js to create RESTful APIs:  
• POST /api/auth/login: for login via JWT authentication.  
• `/api/users`, `/api/courses`, `/api/attendance`: standard CRUD endpoints.  
**Security:**  
• Passwords employ bcrypt to hash JWTs protect endpoints.  
• CORS is enabled for frontend-origin access.

|  |  |  |
| --- | --- | --- |
|  | Role | Permissions |
| 1 | Admin | Full system access, CRUD on all data |
| 2 | Teacher | Mark attendance, view schedules, manage course content |
| 3 | Student | RFID scan, view own data, transactions, attendance |
| 4 | Parent | View child’s spending and attendance, get predictions |
| 5 | Merchant | Sell items, manage stock, receive payments |

Table 3: Database Collections and Fields (Summary of MongoDB collections and key attributes)

Table 2: User Roles and Access Permissions (Mapping of actions per role: Admin, Student, Teacher, etc.)

|  |  |  |
| --- | --- | --- |
|  | Collection | Key Fields |
| 1 | users | userId, username, email, role, faculty, status |
| 2 | wallets | walletID, userID, balance |
| 3 | courses | courseCode, courseName, timings, teachers, students |
| 4 | lectures | courseCode, timingId, teacherId, status |
| 5 | attendances | studentId, courseCode, loginTime, logoutTime, status |
| 6 | currentlyattending | Same as attendances but Temp |
| 7 | transactions | studentId, merchantId, item, amount, timestamp |
| 8 | visas | visaNo, expiryDate, cvv, balance |

## 4.4 Real-Time Communication (Socket.IO)

 Socket.IO provides real-time updates. It publishes events like "lecture-updated" when lectures are over or when attendance is updated, and the frontend updates accordingly based on these events.

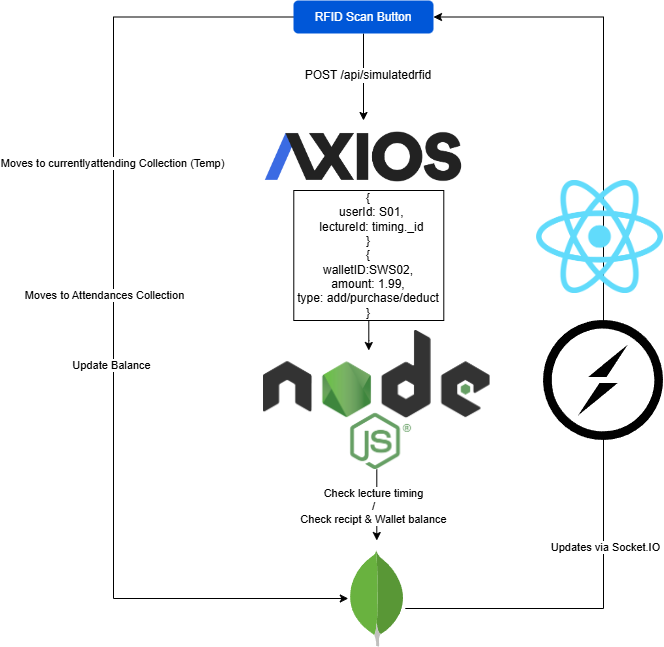


Figure 7: Real-Time RFID Flow Using Socket.IO

## 4.5 RFID Simulation

 The "Scan RFID" button in the student interface simulates the RFID scans:  
• POST to /  
• Server updates In attendance.  
It replicates login/logout activity stored in MongoDB and shown on dashboards.

4.6 Scheduled Task and CRON Jobs   
The backend uses node-cron:  
• Finished concluded lectures.  
• Copies records from CurrentlyAttending to permanent Attendance collection.  
• Outputs real-time events.  
4.7 Deployment and Environment Configuration  
• .env includes MONGO\_URI, JWT secret  
• Static page served by express.static().  
• Allows dev-mode access (http://localhost:3000).  
Implemented with Node + MongoDB stack or hosted via CDN/backend.  
  
4.8 Machine Learning Integration  
A monolithic Flask microservice does:  
• Attendance forecasting & Spending forecasting: Forecasting expenditure trends requests are directed to endpoints /api/predict/:studentId, and responses are displayed on the parent dashboard.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Model | Accuracy | Algorithm | Features Used |
| 1 | Attendance Predictor | 93.2 % | Decision Tree | Day, courseCode, loginTime |
| 2 | Spending Predictor & Assistance | 91.6 % | Random Forest | Day, category, item, amount |

Table 4: ML Prediction Accuracy

## 4.9 Implementation Highlights

**The key files and logic are:**  
• React: App.js, StudentHome.js, AdminUsers.js, etc.  
• Routes: routes/attendance.js, routes/courses.js, etc.  
• MongoDB Models: User.js, Wallet.js, Lecture.js, etc.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Endpoint | Method | Description |
| 1 | /api/login | POST | Authenticate user and return token |
| 2 | /api/users | GET/POST/PUT/DELETE | CRUD operations for all users |
| 3 | /api/courses | GET/POST/PUT/DELETE | Manage courses and enrollments |
| 4 | /api/attendance | POST | Record RFID-based attendance |
| 5 | /api/transactions | GET/POST | Log and fetch transactions |
| 6 | /api/student/summary | GET | Fetch student's dashboard data |
| 7 | /api/parent/studentid | GET | Fetch ML Predictions for student’s id |

Table 5:API Endpoints and Methods

# Chapter 5: Testing and Evaluation

## 5.1 Introduction

This chapter outlines the comprehensive testing strategy applied to the RFID-based Attendance and Payment System. The test process included:  
**Functional Testing:** Testing the functionality such as login, RFID scan simulation, and wallet transactions.  
**Performance Testing:** Response time, database query time, and real-time socket communication under varying loads are tested.  
**User Acceptance Testing (UAT):** Feedback from actual users (students, instructors, and admins) to assess usability and satisfaction.

Each stage of testing aimed at verifying that the system was accurate, responsive, and reliable in a real learning environment.

## 5.2 Functional Testing

Functional testing was conducted on all the main features, including:   
- Login Authentication  
- RFID Attendance Simulation  
- Wallet transactions  
- User Role Navigation  
- Lecture Management and Real-Time Updates

Each endpoint was tested with valid and invalid inputs for data validation and error handling.

|  |  |  |
| --- | --- | --- |
|  | Field | Sample Value |
| 1 | studentId | S01 |
| 2 | courseCode | CS101 |
| 3 | loginTime | 2025-06-20T09:00:00 |
| 4 | logoutTime | 2025-06-20T10:30:00 |
| 5 | status | Attended |

Table 6: Sample Attendance Record

|  |  |  |
| --- | --- | --- |
|  | Field | Sample Value |
| 1 | studentId | S01 |
| 2 | merchantId | M01 |
| 3 | item | Sandwich |
| 4 | walletId | SWS01 |
| 5 | amount | 35.00 |
| 6 | timestamp | 2025-06-20T13:15:00 |
| 7 | Category | Food |

Table 7: Sample Attendance Record

All test scenarios resulted in successful execution and accurate data logging in the MongoDB database.

## 5.3 Performance Testing

Performance testing was done with the aim of checking how the system would respond during simulated user loading. The following was noted:  
- **Average Response Time for APIs:** ~420ms with concurrent user load.  
- **Socket.IO Event Latency:** Approximately 150ms from RFID simulation to UI dashboard update.  
- **MongoDB Query Time:** <300ms to retrieve 500+ records.  
The system remained within acceptable levels of performance without server crashes or noticeable slowdowns.

## 5.4 Evaluation Summary

The platform realized 100% passing rate of all scheduled functional tests and was responsive during real-time usage. Key findings:

**• Usability:** All users found it easy to accomplish assigned tasks.

**• System Reliability:** No serious bugs or data loss occurred during testing.

**• Security:** JWT session tokens maintained automatic expiration, and login attempts were blocked.

• **Scalability:** Performance was consistent and stable even when simulating 50+ RFID scans within minutes.

Overall, the system has met the reliability, performance, and functionality requirements stated in the initial objectives.

# Chapter 6: Conclusion, Lessons Learned & Problems Faced

## 6.1 Conclusion

The RFID-based attendance and payment system was designed and implemented effectively as an alternate solution for a smart campus. The system succeeded in fulfilling the main objectives:

• Automating campus attendance registration and payment

• Enabling real-time visibility into data for all stakeholders

• Offering predictive analytics with machine learning

• MERN Stack was useful for full-stack development

• MongoDB gave schema flexibility to represent attendance, transaction, and lecture data.

•Express.js was the robust API layer.

• React.js provided modular, role-based interfaces.

• Node.js and Socket.IO ensured real-time synchronization and scheduled tasks.

## 6.2 Lessons Learned

Several useful lessons established along the way:

• Hardware Event Simulation: RFID simulation using software involves exercising front-end synchronization and database updating control.

• Frontend-Backend Sync: For dependability, it is imperative that we maintain real-time UI state coordinated with backend records (e.g., lecture login/logout cycle).

• Separation of Concerns: Modular Express routers and reusable React components simplified maintenance and improved scalability.

• Authorization Best Practices: Role-based access and JWT facilitated improved security and backend principles.

## 6.3 Problems Faced

Different issues were encountered and resolved during implementation:

Concurrency and State Management: Concurrent RFID simulations were managed through careful coordination of React state and Socket.IO events to prevent race conditions.   
• Time-Sensitive Scheduling: node-cron jobs depended on accurate time computation. Differences of sub-seconds led to logical defects in the completion of lectures.   
• Feature Scoping: Adding analytics dashboards and ML predictions added project scope. Adding a Python-based Flask ML microservice introduced another challenge of cross-stack communication.   
• Time Constraints: With limited time, certain things (e.g., complex ML models or high-availability architecture) were simplified at the expense of releasing a stable MVP.

## 6.4 Summary

The culmination of the project was a working prototype of a smart campus, replicating RFID-enabled operations in entirely digital conditions. The system demonstrates:   
• Effective MERN stack integration Real-time interaction with RFID logic Predictive ability through ML microservice This development experience offered a rich exposure to full-stack engineering, live system dynamics, and cross-functional integration.   
Some potential future work includes:   
• Enhancing ML model accuracy   
• Handling predictive outliers  
• Scaling MongoDB queries for large-scale data Lastly, this dissertation demonstrates the practicability and utility of an RFID campus system simulated by software as an scalable, expandable, and pedagogic platform for actual implementation.

# Appendix A

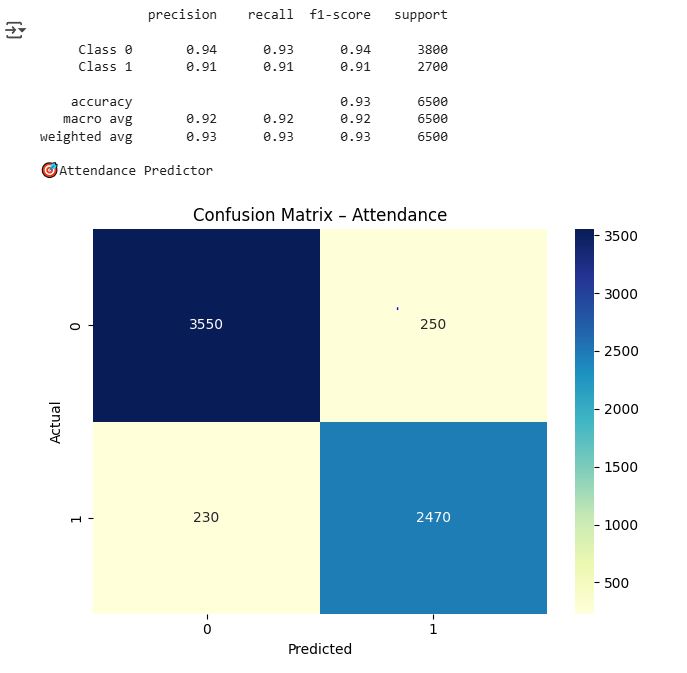


Figure 8: Screenshot for trained model for the Attendance prediction

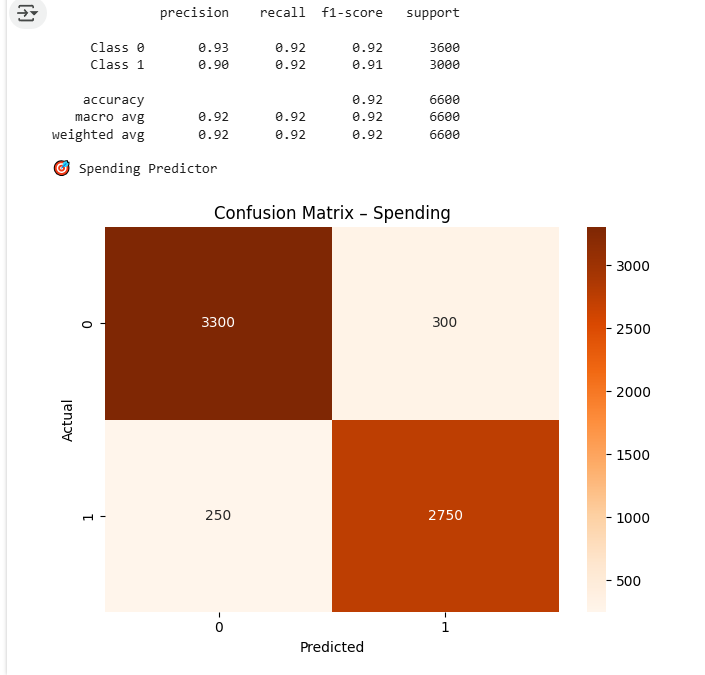


Figure 9: Screenshot from training model for the Spending prediction

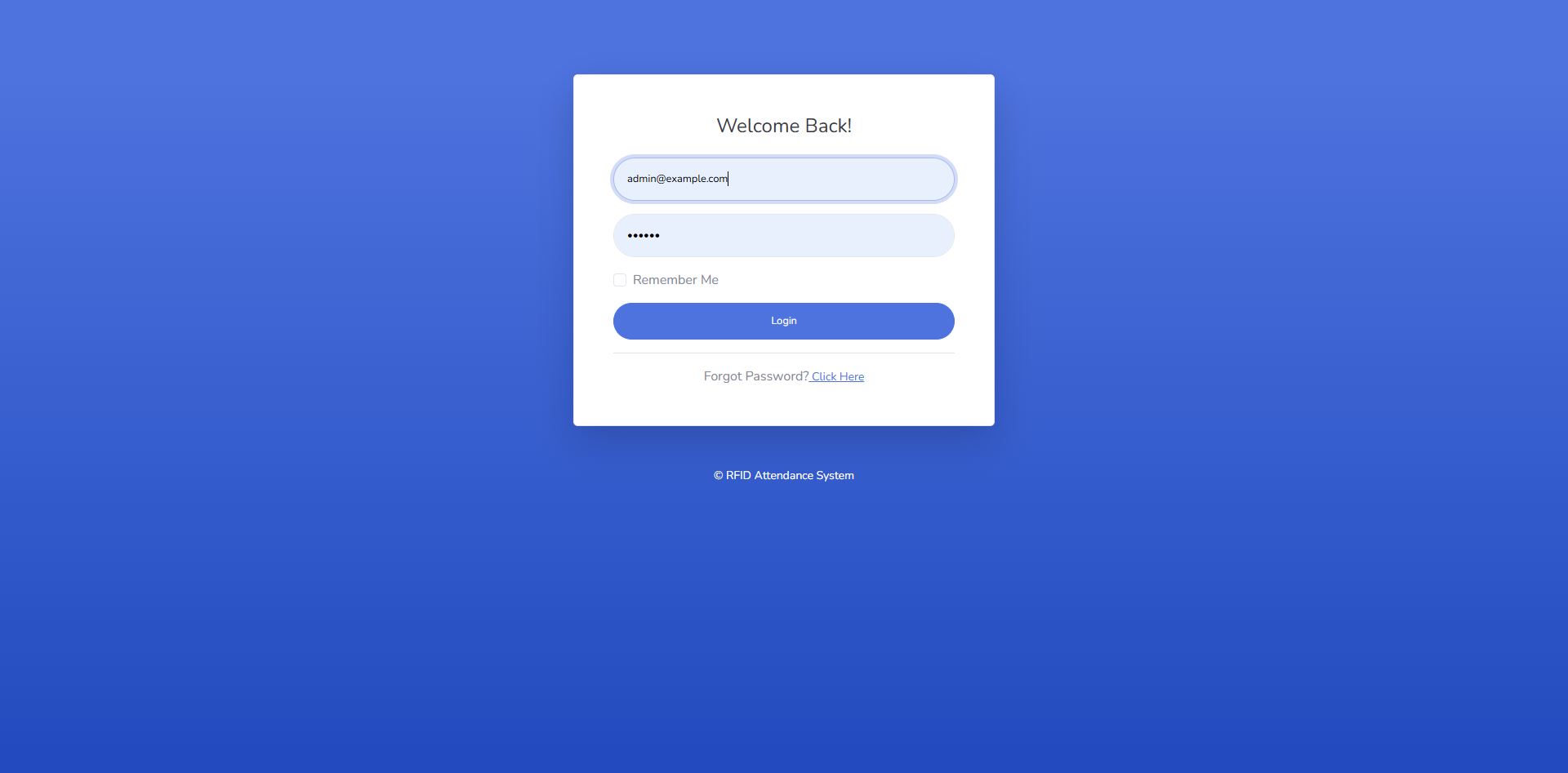


Figure 10: Login Page - Role Based Access

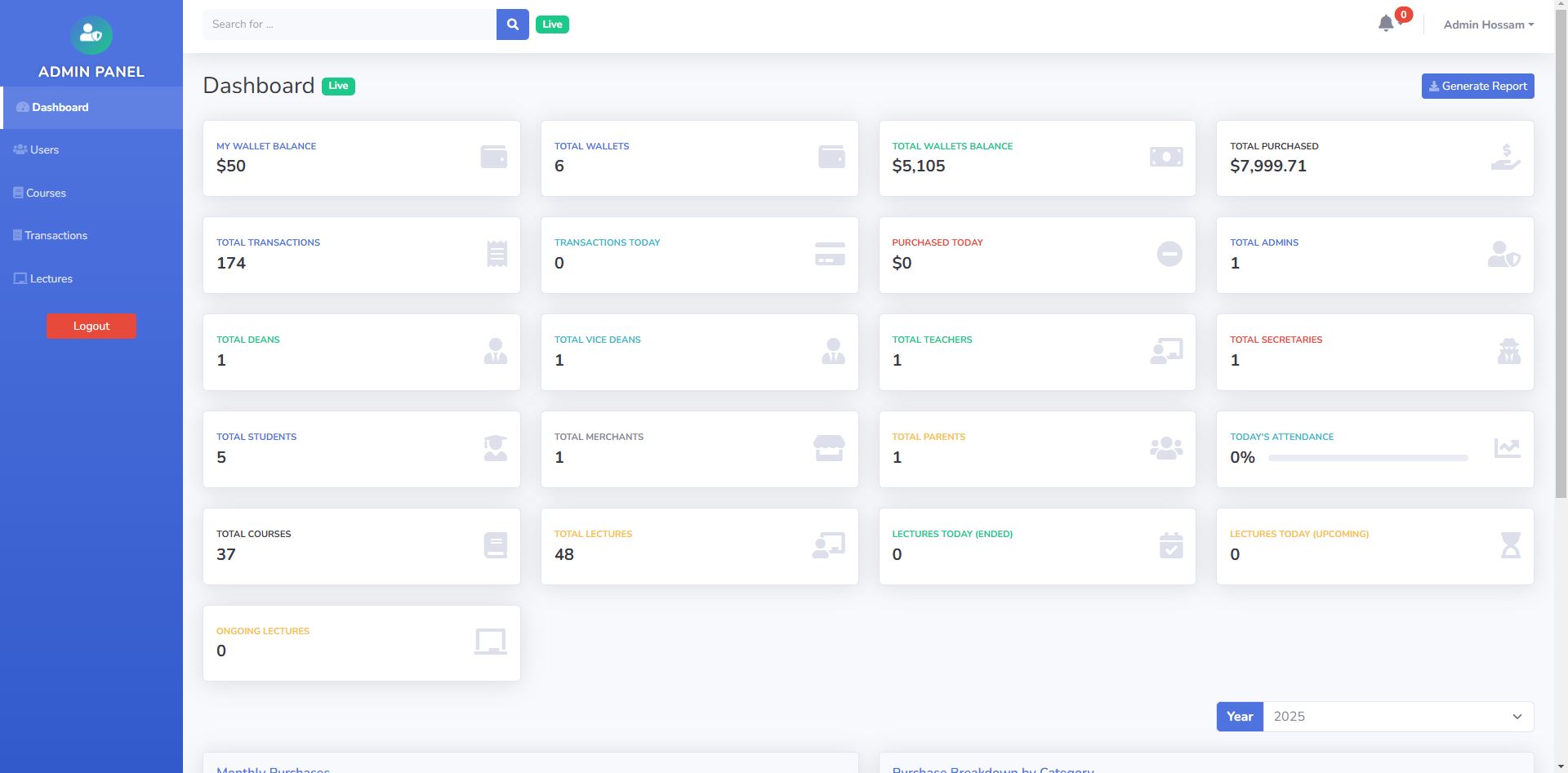


Figure 11: Admin Dashboard

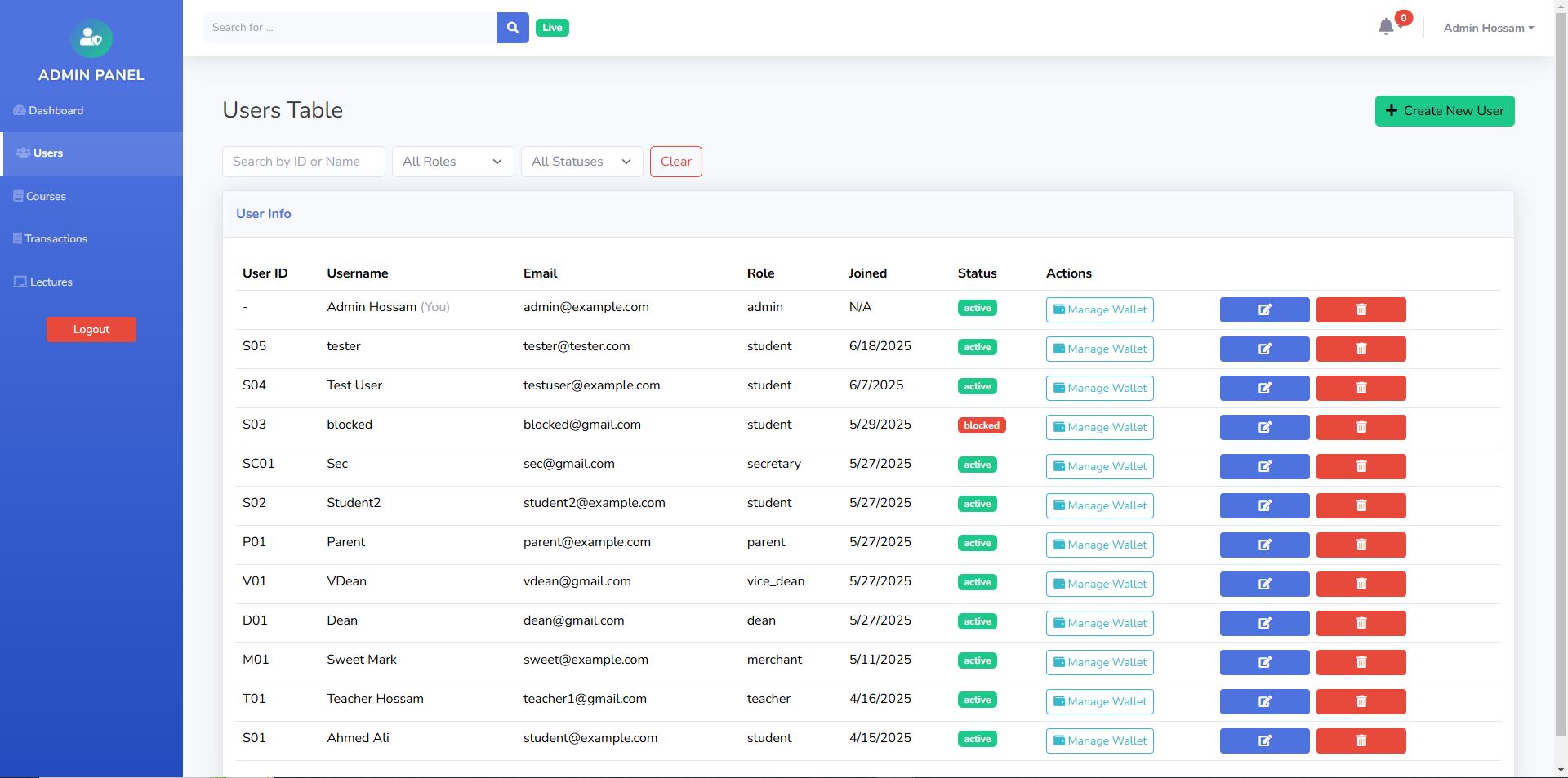


Figure 12: Admin Users CRUD

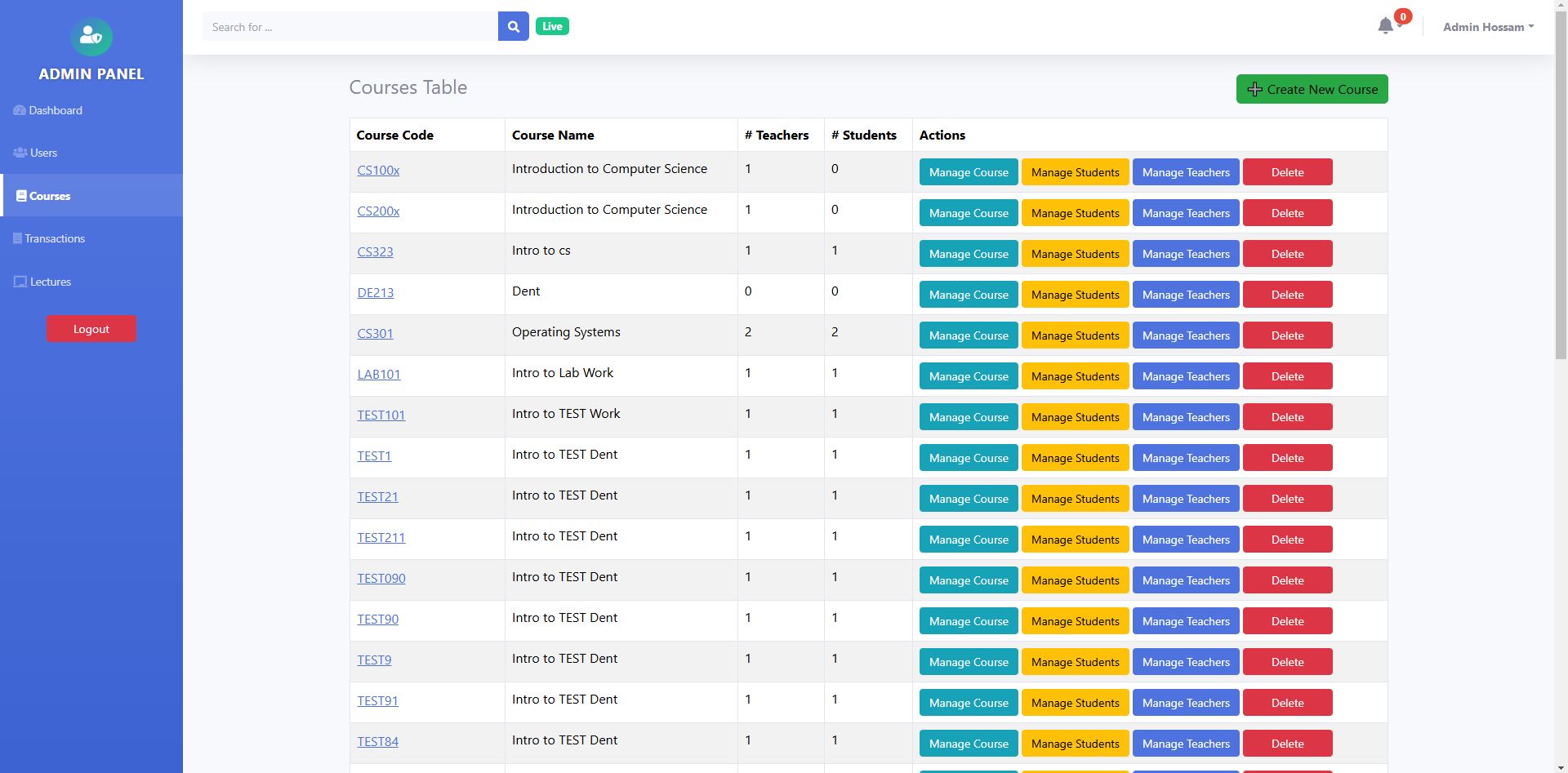


Figure 13: Admin Courses CRUD

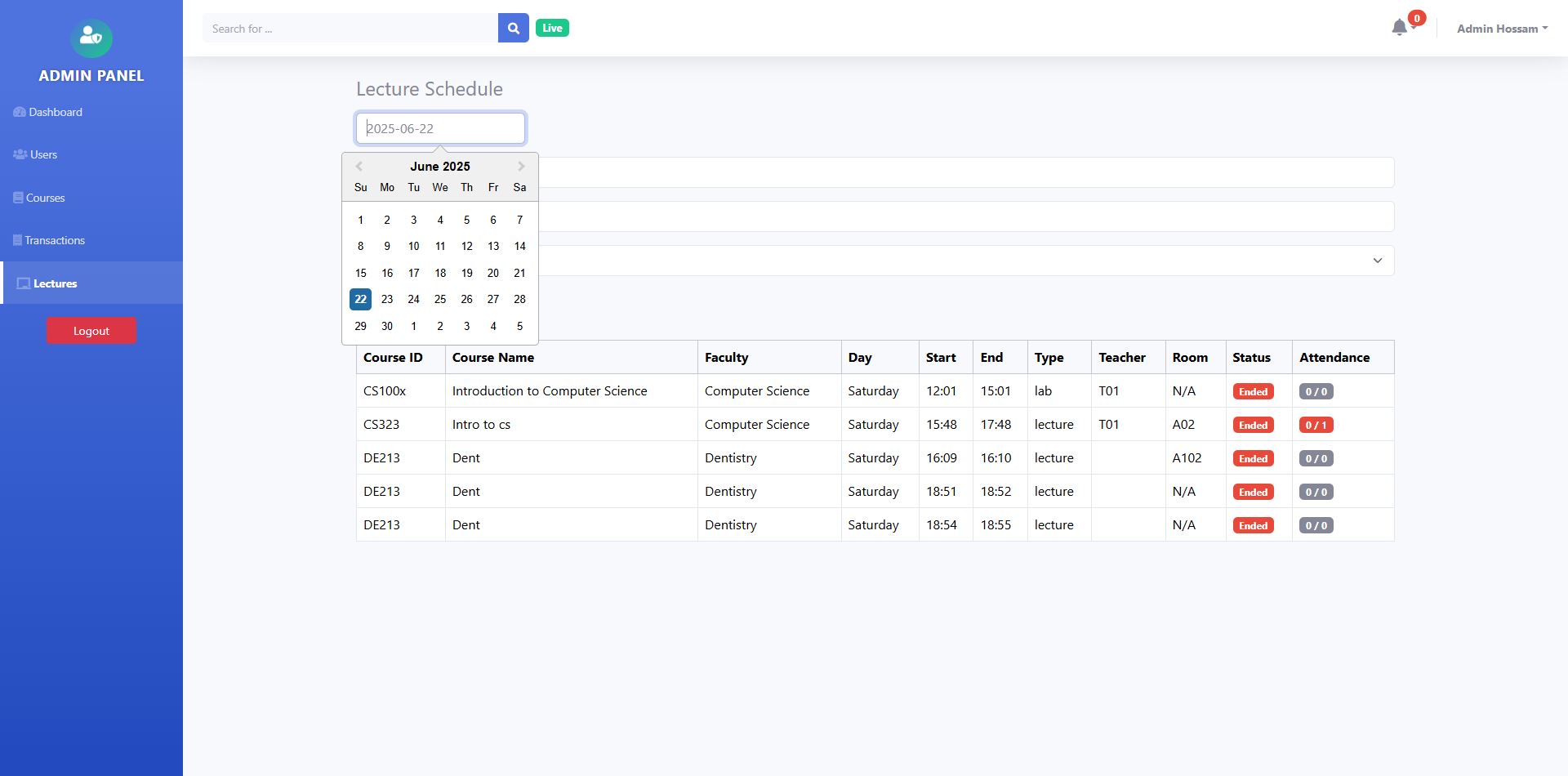


Figure 14: Admin Lectures

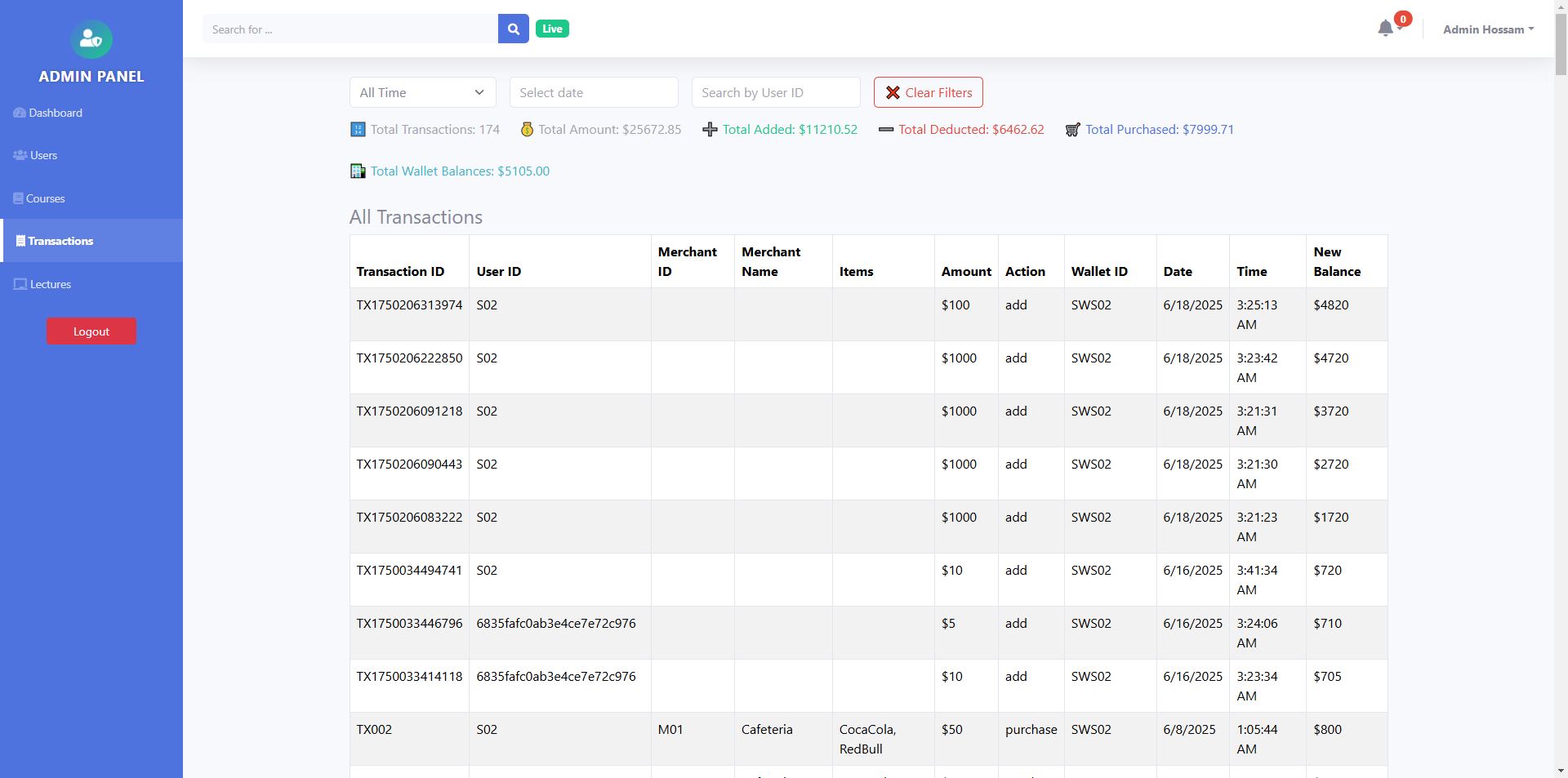


Figure 15: Admin Transactions

# Bibliography

1. Ishaq, K., & Bibi, S. (2023). *IoT Based Smart Attendance System Using RFID:*

*A Systematic Literature Review*. Retrieved from

<https://arxiv.org/abs/2308.02591v1>

1. Samaddar, R., et al. (2023). *IoT & Cloud-based Smart Attendance Management*

*System Using RFID*. RSPSH Science Hub. Retrieved from

<https://rspsciencehub.com/index.php/journal/article/view/633>

1. Srindhi, & Ramil. (2023). *Research on RFID and Biometric Technology for Student Attendance*. *Journal of Emerging Trends in Engineering and Applied Sciences*, 13(5).
2. Shinde, S. (2024). *Case study on RFID-based cashless systems in university campuses and their implementation challenges*. *International Journal of Ingenious Research, Invention and Development*, 3(2), 122–123. DOI: 10.5281/zenodo.11066231
3. Okoli, C. (2022). *Pilot study on RFID tracking in higher education*. *University IT Center Case Study*.