# *Elmanassa Software Documentation*

## Content :

## 1. Introduction

### I. Purpose

The purpose of Elmanassa is to create a comprehensive software solution for teachers to streamline the management of educational activities. The system aims to enhance teacher-student interaction, simplify administrative tasks, and support efficient learning processes. By providing both a web platform and a mobile application The system reduces manual workload, automates repetitive tasks, and offers a modern, easy-to-use digital experience for teachers and students alike. It aligns with the growing need for digital transformation in education and contributes to Sustainable Development Goal (SDG) 9 — Industry, Innovation, and Infrastructure

## II. Scope

### In Scope :

1. User Management: Teachers, students, and admins will have different accounts with specific permissions.
2. Content Sharing: Teachers can upload educational materials (like PDFs, videos, and notes) for students.
3. Quizzes & Assignments: Teachers can create quizzes, assignments, and grade student submissions.
4. Attendance Tracking: Students can scan QR codes to mark their attendance automatically.
5. Lesson Scheduling: Teachers can create lesson schedules for students to follow.
6. Notifications & Announcements: Teachers can send announcements and notifications directly to students.
7. Progress Tracking: Students can track their own progress, while teachers can track student attendance, quiz scores, and grades.
8. Reports & Analytics: Teachers can view reports on student performance, attendance, and class activity.
9. Cross-Platform Access: Teachers use the web app, while students access the system using a mobile app.

### Out of Scope :

1. Live Video Classes: Elmanassa will not support live video streaming for lessons.
2. Automatic Essay Grading: It will not automatically grade long-form written answers.
3. Offline Access: Students and teachers must have an internet connection to use Elmanassa.
4. Payment Handling: The system will not handle financial transactions or payments.
5. Language Support: The system will not support multiple languages in its first version.

## III. Acronyms & Definitions

### I. Acronyms

| Acronym | Full Form |
|---|---|
| **SDG** | Sustainable Development Goal Elmanassa supports SDG 9 Industry, Innovation, and Infrastructure |
| **API** | Application Programming Interface (enables communication between different software components). |
| **RBAC** | Role-Based Access Control (restricts access based on user roles like student, teacher, or admin). |
| **UI** | User Interface (the visual design of the app or website). |
| **UX** | User Experience (how users feel when using the system). |
| **SQL** | Structured Query Language (used to manage the database). |
| **QR Code** | Quick Response Code (used for attendance scanning). |
| **CRUD** | Create, Read, Update, Delete (basic actions on data). Create, Read, Update, Delete (basic actions on data). |
| **cPanel** | Control Panel (used for teacher authentication). |

### II. Definitions

| Term | Definition |
|---|---|
| **Elmanassa** | A software system for teachers to manage education, track student progress, and automate tasks. It includes both a web platform and a mobile app |
| **Authentication** | Verifying user identity to allow access to the system. |
| **SQL Server** | The database where all data is stored. |
| **Context Diagram** | A diagram that shows the system and its interactions with users |
| **Activity Diagram** | A flowchart of activities within the system. |
| **Use Case Diagram** | A diagram showing the interactions between users and the system. |
| **Flutter** | is a popular open-source framework used to build natively compiled applications for mobile, web, and desktop from a single codebase. |
| **Frontend** | The part of the system that users interact with directly. It includes the web platform and mobile app interfaces, built using HTML, Bootstrap, Java Script ,and Flutter, respectively |

# 2. Requirements

## I. System Requirements

### i. Software Stack

**Backend:**

- **Language**: C# (.NET Core 8.0).
- **Framework**: .NET Core for API development.
- **Database**: SQL Server 2022.

**Frontend (Web):**

- **HTML, Bootstrap**: For responsive and styled web pages.
- **jQuery**: To add interactivity and handle AJAX requests.

**Mobile App:**

- **Flutter**: To create a cross-platform mobile app compatible with both Android and iOS.

### ii. System Requirements for Server

**Server Hardware Requirements:**

- **CPU**: 8-core processor (e.g., Intel Xeon or AMD EPYC) to manage API requests and database operations efficiently.
- **RAM**: 16 GB or more, ensuring fast data processing, especially for handling real-time analytics, notifications, and large material uploads.
- **Storage**:
  - **Primary Storage**: SSD (1 TB or more) for storing application files and data.
  - **Backup Storage**: An additional storage option (e.g., cloud or external hard drives) for data backups and failover scenarios.

- **Network Bandwidth**: At least 1 Gbps for consistent data transfer, especially with multimedia content (videos, PDFs).

**Server Software Requirements:**

- **Operating System**: Windows Server 2022 for compatibility with SQL Server and .NET Core.
- **Web Server**: IIS (Internet Information Services) for hosting the API and web content.
- **Database**: SQL Server 2022 for optimal performance, security, and scalability.
- **.NET Core Runtime**: .NET Core 8.0 (latest LTS version) for API development and backend processes.
- **Additional Software**:
    - **SSL Certificates** for secure data transmission.
    - **Backup Solutions** such as Azure Backup or SQL Server Maintenance Plans for data safety.
    - **Security Software**: Firewalls and endpoint protection to secure against unauthorized access.

## iii. User Device Requirements

Each type of user device needs enough power to handle multimedia and interactive features without lags.

**For Teachers and Administrative Users:**

These users require devices capable of managing the web interface (HTML, Bootstrap, jQuery) and uploading/organizing content.

- **Device Type**: Laptop or desktop computer.
- **Operating System**: Windows 10 or later, macOS 11 (Big Sur) or later.
- **CPU**: Dual-core processor (e.g., Intel i5 or AMD Ryzen 5).
- **RAM**: 8 GB minimum.
- **Storage**: At least 256 GB (SSD recommended for faster loading times).

- **Browser**: Latest versions of Chrome, Firefox, or Edge for optimal compatibility with the frontend.

**For Students (Mobile App via Flutter):**

Since students will primarily use a mobile app, their devices should support Flutter-based applications for smooth functionality.

- **Operating System**:
    - **Android**: Version 9.0 (Pie) or higher.
    - **iOS**: iOS 13 or higher.
- **RAM**: 4 GB minimum for optimal performance with multimedia and interactive elements.
- **Storage**: At least 64 GB to handle the app and any downloaded materials.
- **Camera**: QR code scanning for attendance requires a camera with autofocus capability.
- **Internet Connection**: Wi-Fi or 4G/5G for accessing real-time notifications and materials.

**General Public Users:**

General users only need a browser to access the public home page.

- **Device Type**: Any device with a browser, including older computers and mobile devices.
- **Browser**: Latest versions of Chrome, Firefox, Safari, or Edge for compatibility with the HTML, Bootstrap, and jQuery frontend.

### iv. Network Requirements

- **For Server**: Minimum 1 Gbps bandwidth to handle high traffic and ensure fast data transfer.
- **For Users**: Minimum 4G connection recommended for a smooth experience, particularly for viewing multimedia content.

## II. Functional Requirements

### i. Registration & Login

Teacher Side: Secure login for teachers via cPanel. Student Side: Registration only for pre-approved emails by the teacher. Login with email and password to access the dashboard and materials. Basic account information like email, password should be collected. Provide an option to reset passwords via email

### ii. Profile Management

Teacher: Complete profile setup including contact details and photo. Student: Profile setup with details like name, age, level, department, photo, and contact number. Ability to edit and update profile information.

### iii. Lesson Management (for Teachers)

Quiz and Assignment Creation: Create, manage, and publish quizzes and assignments. Automated grading and feedback, with manual review options. Access grades Share Progress and Grades Reports with Parents Material Management: Upload and organize resources (PDFs, videos, etc.) in a searchable library Delete a lesson if it is no longer needed Post updates, reminders, and announcements for specific classes. Schedule Management: Create, manage, and sync offline lesson schedules with a calendar. Share lesson plans, resources, and teaching tips with students Event Announcements: Publish announcements for events like guest lectures or workshops.

### iv. Lesson Access & Viewing (for Students)

Profile and Materials Access: View profile, allowed materials, and quizzes. Download materials for offline use. Register into Courses ### Students should have an option to mark a lesson as completed. Students should be able to mark certain lessons as "favorites" to easily access them later. Attend quizzes and view grades Submitted assignments Submit questions as comments for lectures Basic information about the teacher such as feedback is accessible Progress Tracking: Personal dashboard with tracking of progress in quizzes, assignments, and material access. Calendar of Events and Lessons: Integrated view of assignments, quizzes, and lesson schedules.

### v. Search & Filtering

Teacher: Search and filter library resources by topic, date, or type. Student: Search for materials and by filters like keywords,level and subject.

### vi. Feedback and Ratings

Teacher: Student Progress tracking Provide feedback on student quizzes and assignments. Student: Receive feedback on assignments and quizzes. View feedback about the courses

### vii. Notifications

Teacher: Send push notifications for assignments, upcoming lessons, new materials, and updates. A notification when they receive feedback on a lesson. Student: Receive notifications about assignments, lessons, new materials, and reminders. Students should be notified by the new updates

### viii. Attendance Management

QR Code Attendance: Students scan QR codes at sessions to mark attendance. Attendance Tracking for Teachers: View attendance data in the analytics dashboard.

### ix. Communication

Messaging: Teacher can send messages to specific student groups.

## III. Non Functional Requirements

### i. Performance

Pages should load within a specified time for loading pages and resources, logging in, accessing courses, and viewing assignments to ensure smooth navigation. Must handle multiple simultaneous users, especially during peak hours ( beginning of classes or assignment deadlines, exam periods). High speed data handling, especially for file uploads, downloads and so on.

### ii. Usability

The platform should have a user friendly interface suitable for all users (students and teachers) should be able to navigate, find classes, and submit assignments with ease, load material and so on. The platform must support increasing numbers of users (students) without a drop in performance. Make sure there are consistent menus, icons, and buttons across all pages for easy navigation.

iii. **Reliability**

Ensures no data loss or corruption during storage, retrieval, or transmission. The platform should continue functioning in case of component failures, like database or server issues.

iv. **Security**

Encryption for data transmission, including sensitive information like personal. Secure login with multi-factor authentication and role-based access control authorization (teacher, students ).

v. **Maintainability**

Code should be modular to facilitate future updates and maintenance and complete technical documentation for developers. The platform should support easy updates with minimal downtime, especially during non-peak hours.

vi. **Portability**

Ensure compatibility across different devices, including desktops, tablets, and mobile phones. Supports major browsers (Chrome, Firefox, Safari, Edge) with consistent functionality.

vii. **Scalability**

Ability to add more servers to handle increased user load. The system should support growth in data volume (student records, content uploads, lecture videos).

viii. **Availability**

The platform should have a high level of availability, ensuring it's always accessible for students .

ix. **Backup and Recovery**

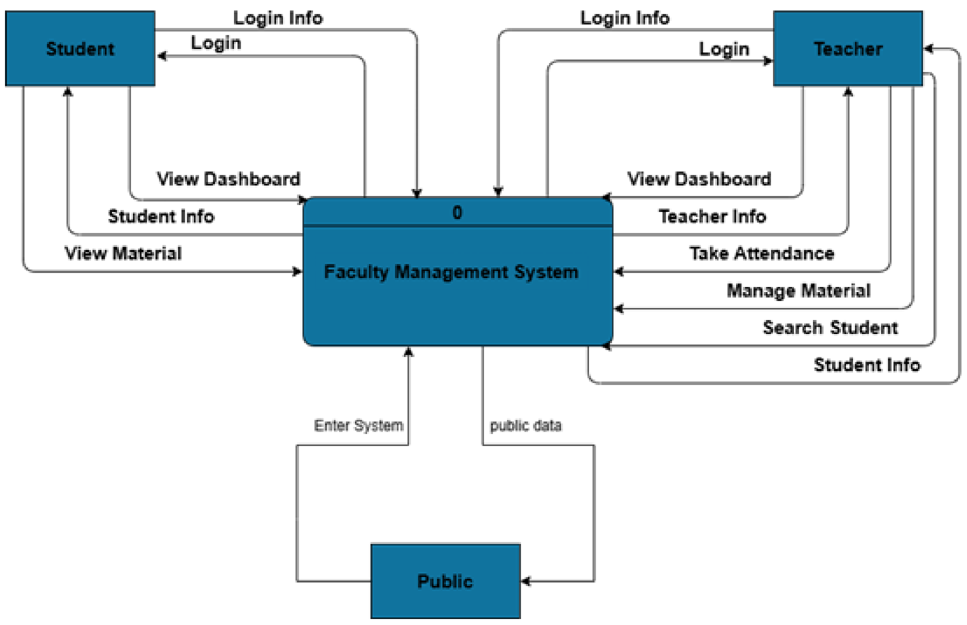
Regular automatic backups to secure student, teacher, and course data. Recovery plans for any disaster to restore data and operations quickly in the event of a catastrophic failure.

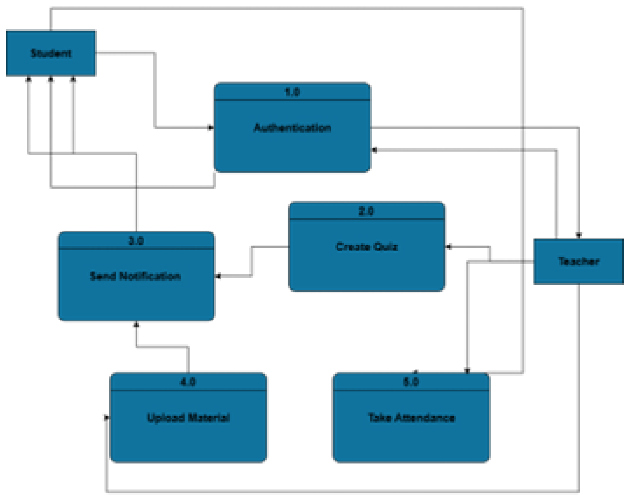
x. **Analytics and Reporting**

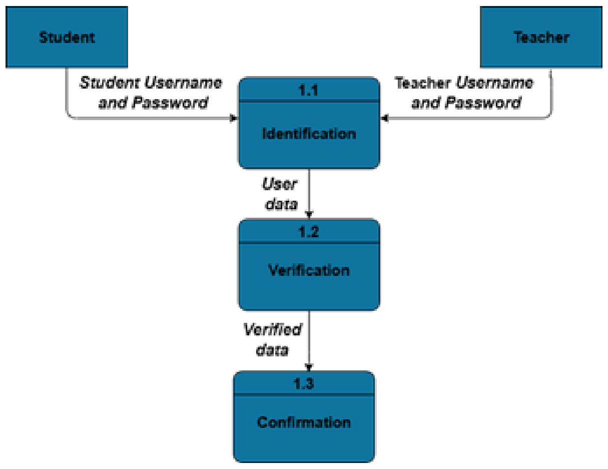
Track platform usage for monitoring engagement and improving the user experience. Aggregate reports on student performance and Regular monitoring of system performance.

## 3. Design Diagrams

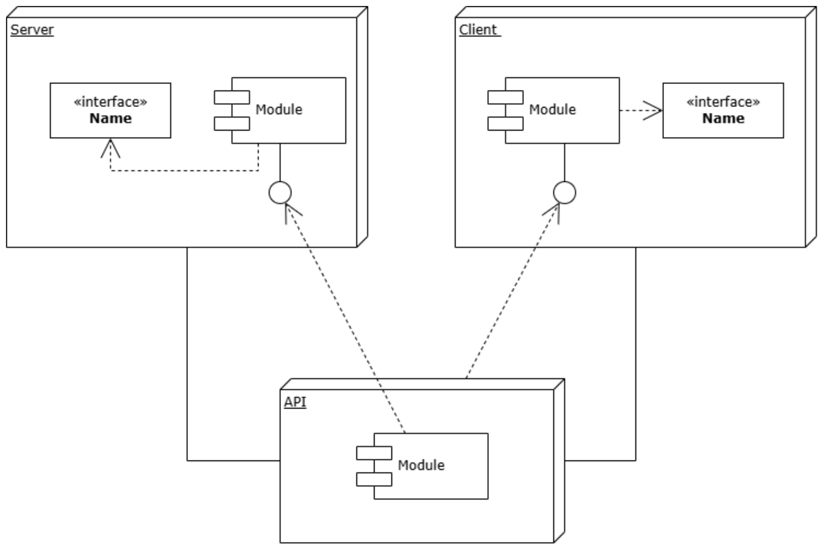
### i. Context Diagram



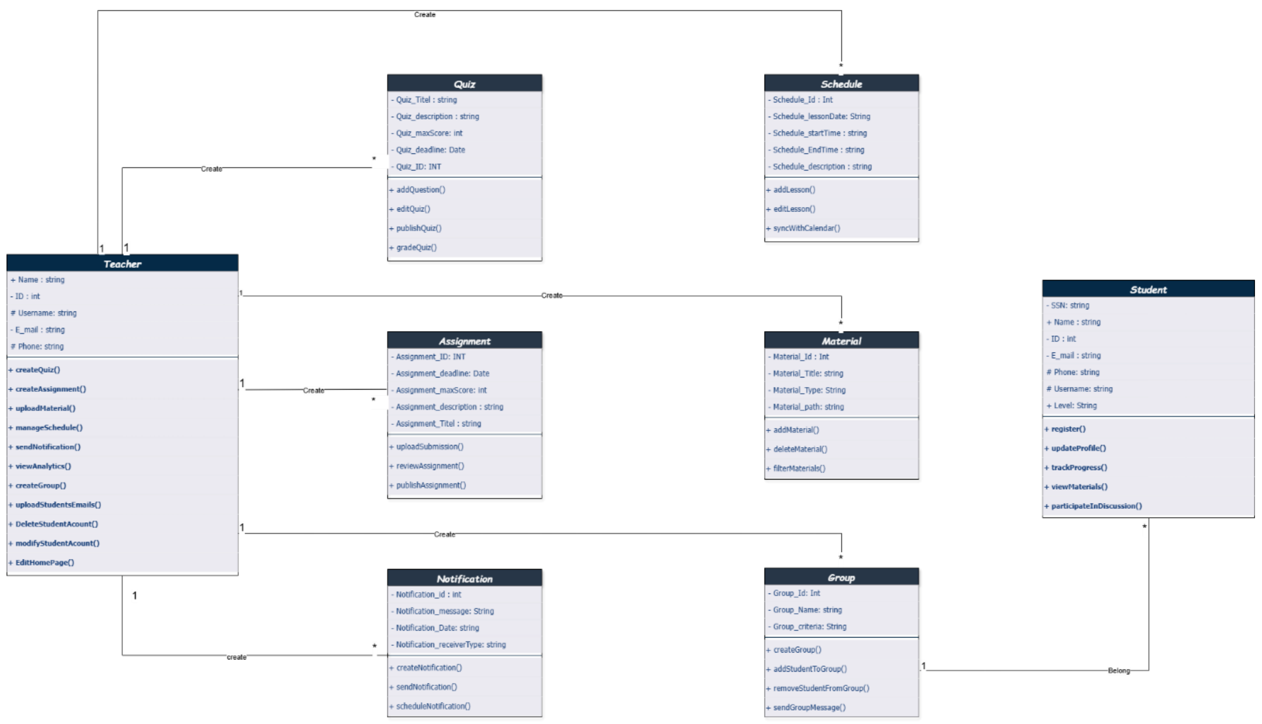




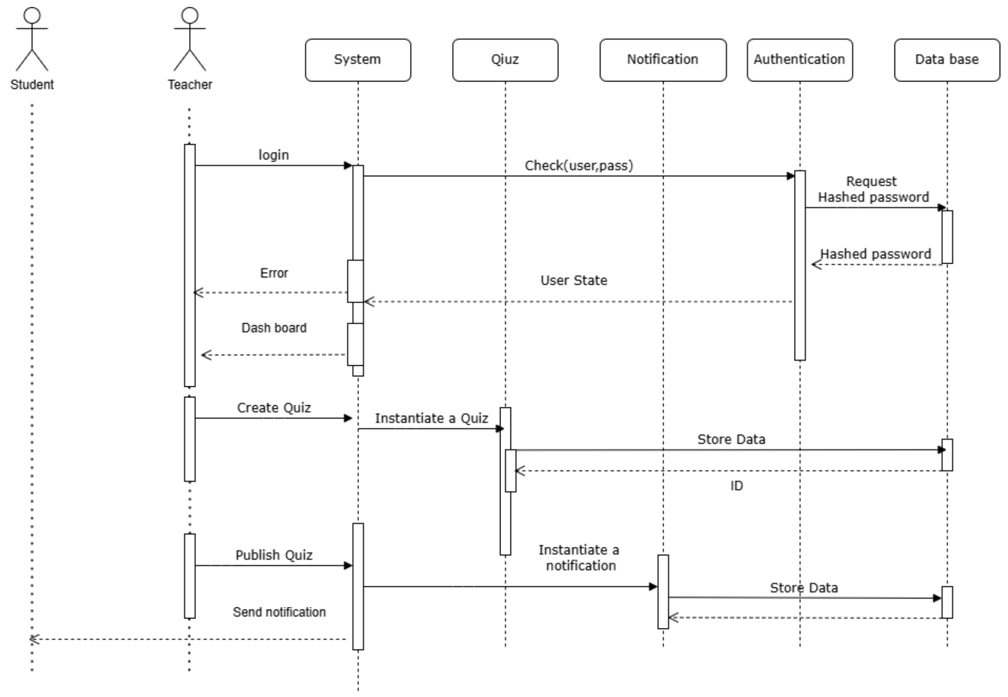
## ii. Component Diagram

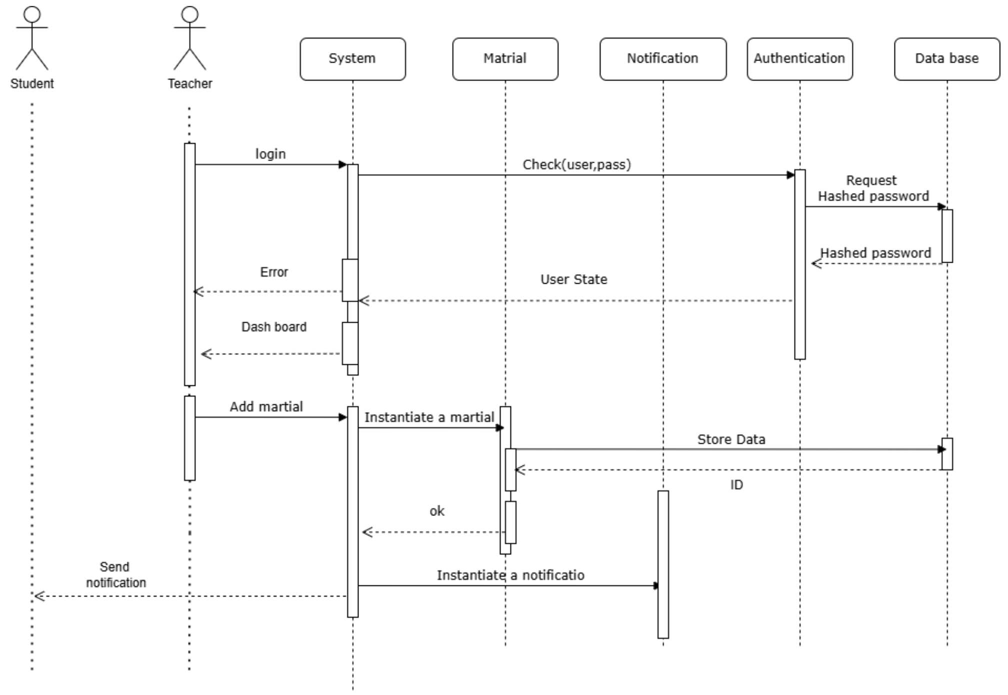


## iii. Class Diagram

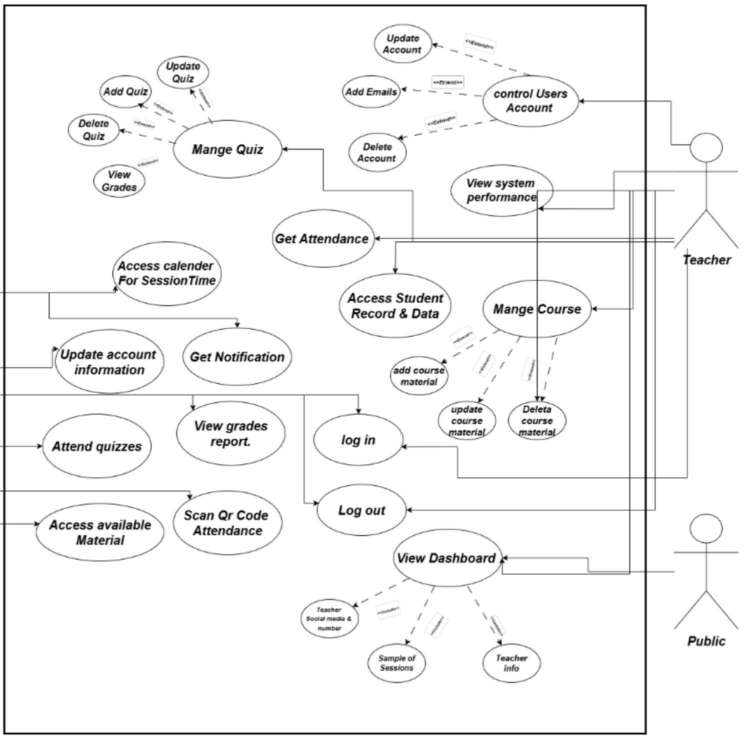


## iv. Sequence Diagram

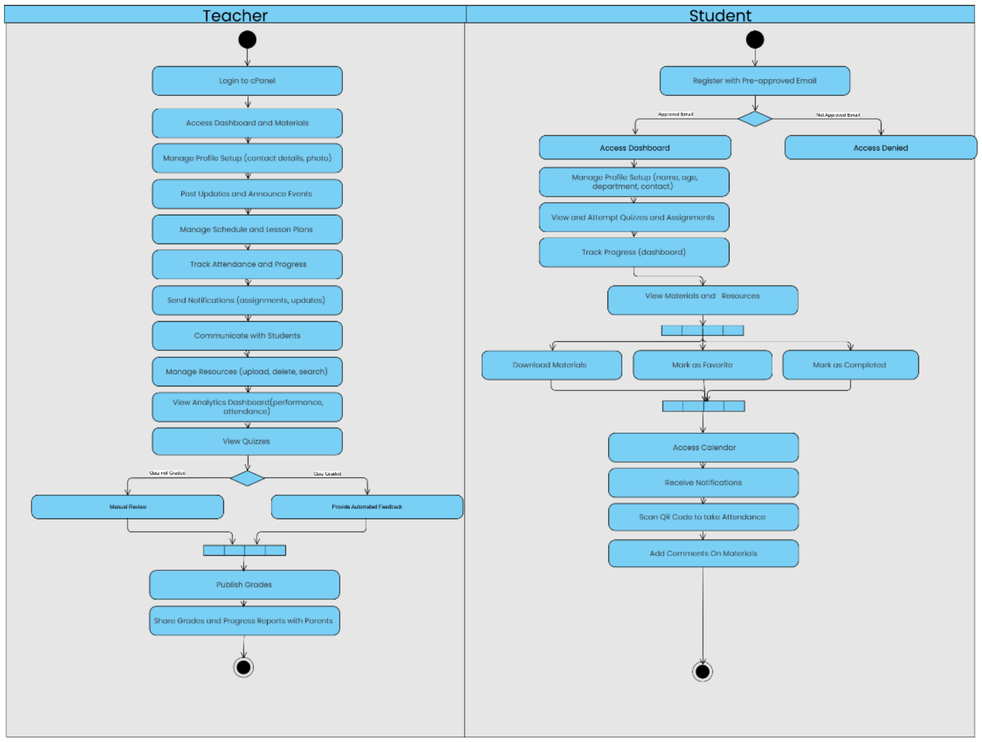




## v. Use Case Diagram

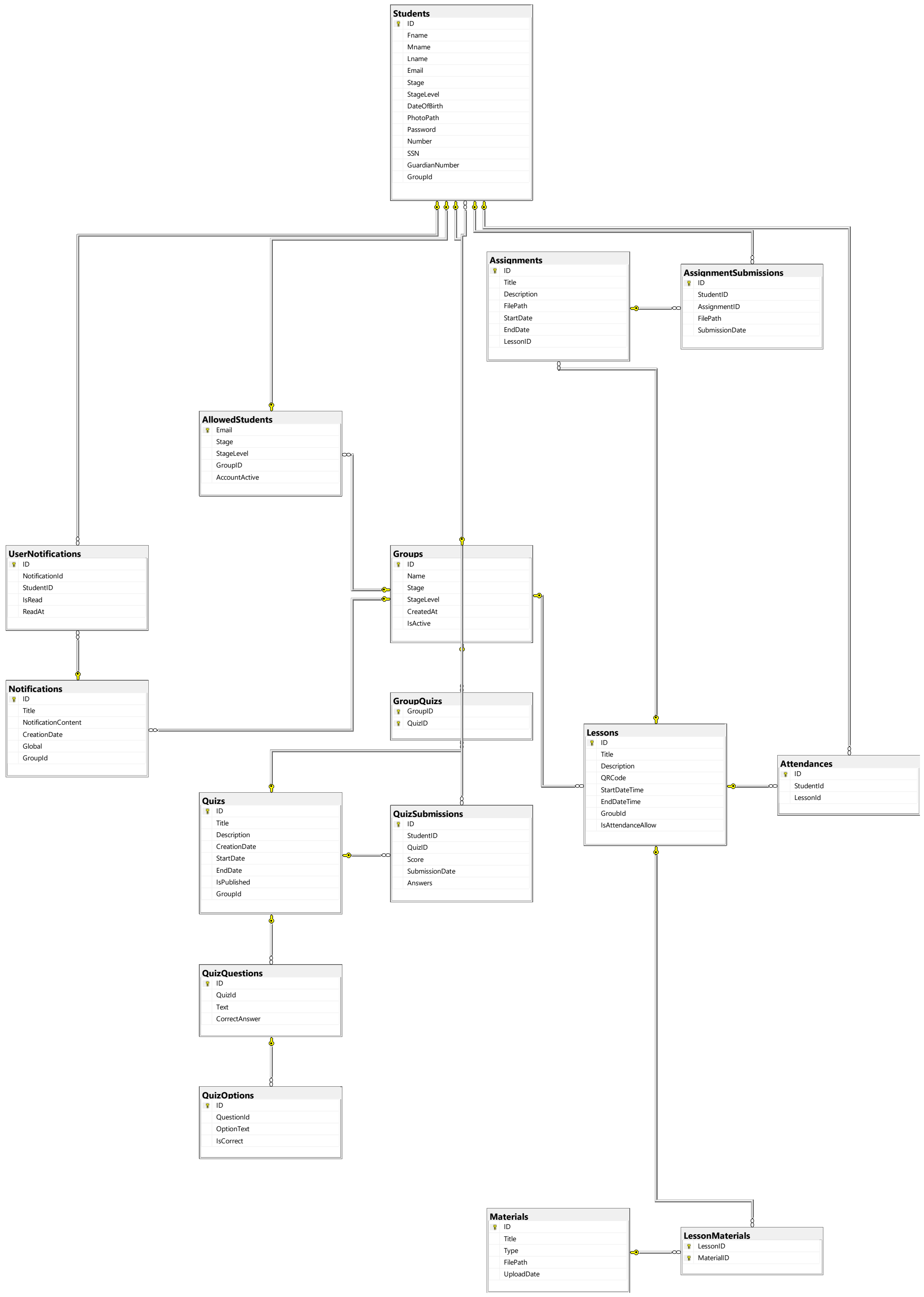


## vi. Activity Diagram


Teacher
Login to cPanel
Access Dashboard and Materials
Manage Profile Setup (contact details, photo)
Post Updates and Announce Events
Manage Schedule and Lesson Plans
Track Attendance and Progress
Send Notifications (assignments, updates)
Communicate with Students
Manage Resources (upload, delete, search)
View Analytics Dashboard(performance, attendance)
View Quizzes
Quiz not Graded
Quiz Graded
Manual Review
Provide Automated Feedback
Publish Grades
Share Grades and Progress Reports with Parents
Student
Register with Pre-approved Email
Approved Email
Not Approved Email
Access Dashboard
Access Denied
Manage Profile Setup (name, age, department, contact)
View and Attempt Quizzes and Assignments
Track Progress (dashboard)
View Materials and Resources
Download Materials
Mark as Favorite
Mark as Completed
Access Calendar
Receive Notifications
Scan QR Code to take Attendance
Add Comments On Materials

## 4. System Design

### ERD



## API Overview

Authentication: Role-based login (teacher, student) using JWT Tokens.

Data Handling: CRUD operations (Create, Read, Update, Delete) for users, materials, assignments, quizzes, and attendance.

Role Management: Each user (teacher, student) has different access rights.

Error Handling: API returns standard HTTP status codes for success (200, 201) and errors (400, 401, 404, 500).

Security: JWT Authentication and RBAC (Role-Based Access Control)

## API Endpoints

| Endpoint | Method | Description | Role |
|---|---|---|---|
| **Teacher Endpoints** | | | |
| **/api/Teacher/Account/LogIn** | **POST** | **Login for teachers** | **Public Access** |
| **/api/Teacher/AllowedStudent** | **GET** | **Get list of allowed students** | **Teacher** |
| **/api/Teacher/AllowedStudent** | **POST** | **Add a new allowed student** | **Teacher** |
| **/api/Teacher/AllowedStudent** | **DELETE** | **Remove an allowed student** | **Teacher** |
| **/api/Teacher/AllowedStudent** | **PUT** | **Update information for an allowed student** | **Teacher** |
| **/api/Teacher/Group** | **GET** | **Get list of groups** | **Teacher** |
| **/api/Teacher/Group** | **POST** | **Create a new group** | **Teacher** |
| **/api/Teacher/Group** | **DELETE** | **Delete a specific group** | **Teacher** |
| **/api/Teacher/Group** | **PUT** | **Update a group’s information** | **Teacher** |
| **/api/Teacher/Notification** | **GET** | **Get list of notifications** | **Teacher** |
| **/api/Teacher/Notification** | **POST** | **Create a new notification** | **Teacher** |
| **/api/Teacher/Notification** | **DELETE** | **Delete a specific notification** | **Teacher** |
| **/api/Teacher/Notification** | **PUT** | **Update a notification** | **Teacher** |
| **/api/Teacher/Material** | **GET** | **Get list of learning materials** | **Teacher** |
| **/api/Teacher/Material** | **POST** | **Upload new learning material** | **Teacher** |
| **/api/Teacher/Material** | **DELETE** | **Delete specific learning material** | **Teacher** |
| **/api/Teacher/Material** | **PUT** | **Update learning material** | **Teacher** |

|  |  |  |  |
|---|---|---|---|
| /api/Teacher/Lesson | GET | Get list of lessons | Teacher |
| /api/Teacher/Lesson | POST | Create a new lesson | Teacher |
| /api/Teacher/Lesson | DELETE | Delete a lesson | Teacher |
| /api/Teacher/Lesson | PUT | Update lesson details | Teacher |
| /api/Teacher/Quiz | GET | Get list of quizzes | Teacher |
| /api/Teacher/Quiz | POST | Create a new quiz | Teacher |
| /api/Teacher/Quiz | DELETE | Delete a specific quiz | Teacher |
| /api/Teacher/Quiz | PUT | Update quiz information | Teacher |
| /api/Teacher/Student | GET | Get list of students | Teacher |
| /api/Teacher/Student | POST | Add a new student | Teacher |
| /api/Teacher/Student | DELETE | Delete a student from the system | Teacher |
| /api/Teacher/Student | PUT | Update student information | Teacher |
| Student Endpoints |  |  |  |
| /api/Student/Account/LogIn | POST | Login for students | Public Access |
| /api/Student/Account/Register | POST | Register new student | Public Access |
| /api/Student/Notification | GET | Get notifications for student | Student |
| /api/Student/Lesson | GET | View available lessons | Student |
| /api/Student/Assignment | POST | Submit an assignment | Student |
| /api/Student/Quiz | GET | View list of available quizzes | Student |
| /api/Student/Quiz | POST | Submit answers for a quiz | Student |
| /api/Student/TakeAttendance | POST | Submit attendance via QR or code | Student |

## Team Member

Malak Alsayed
Flutter Developer

Ahmed Fahmy
Backend Developer

Mohamed Mohsen
Database Developer

Marwa Abdelmonem
Frontend Developer