EduSync
(School Management System)
Application Requirements Specification

Name: Achuth Chandra Moonnamkuttiyil
UMD Email ID: achuth@umd.edu
UMD Directory ID: Achuth
UID: 120311482

Table of Contents

List of Figures

1. Introduction

The EduSync - School Management System (SMS) is a web-based application designed to modernize and streamline educational institution operations. This system aims to replace traditional paper-based methods with a digital solution that enhances efficiency, improves communication, and provides real-time access to crucial information for administrators, teachers, and students.

1.1 Revision History

Date	Comment	Author
2024-09-22	Initial document	Achuth Chandra Moonnamkuttiyil
2024-10-20	Updated requirements document	Achuth Chandra Moonnamkuttiyil
2024-10-24	Updated with Threat Modelling and Misuse Cases	Achuth Chandra Moonnamkuttiyil

2. Overall Description

2.1 Brief Description

EduSync - School Management System is a comprehensive digital platform that facilitates the management of various school operations, including student and teacher information, attendance tracking, grade management, and scheduling. It serves as a centralized hub for educational data and processes, accessible to authorized users based on their roles within the institution.

2.2 Detailed Description

Edusync is built to address the unique needs of different user groups within a school environment. It offers a range of functionalities tailored to administrators, teachers, and students, ensuring that each group can efficiently perform their respective tasks and access relevant information.

Key functionalities include:

1. User Management: The system supports three distinct user roles - Administrator, Teacher, and Student - each with specific permissions and access levels.

2. Student and Teacher Information Management: Administrators can add, view, and manage detailed profiles for students and teachers.

3. Attendance Tracking: Teachers can record and manage student attendance, which can be viewed by students and administrators.

4. Grade Management: Teachers can input and update student grades for various assessments, viewable by students and administrators.

5. Timetable Management: Administrators can create and manage class schedules, which can be viewed by teachers and students.

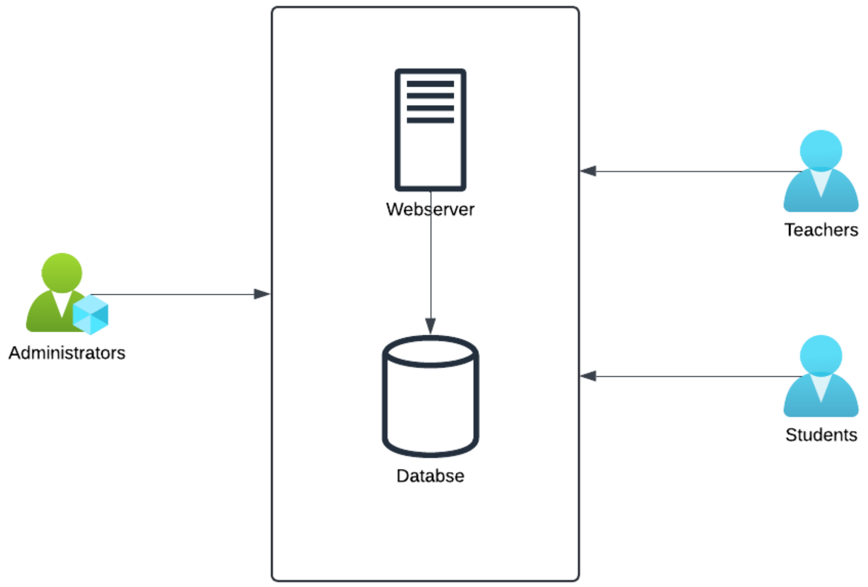

Figure 1: System Environment

3. Use Cases

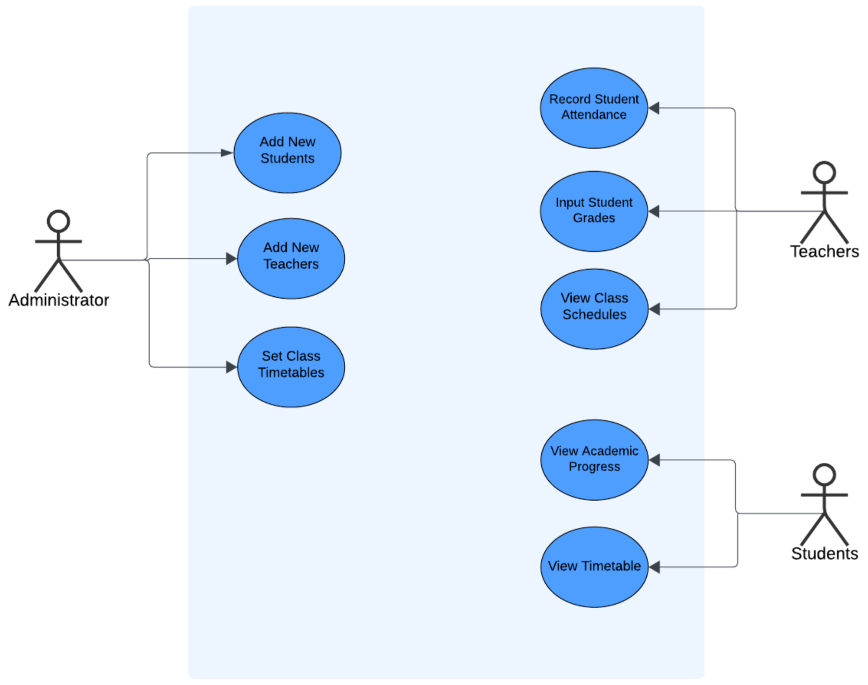

Figure 2: User Types and their Use cases

3.1 Administrator Use Cases

3.1.1 Manage Roles

- **Actor:** Administrator
- **Description:** The administrator can assign and change roles for registered users.
- **Normal Flow:**
    1. Administrator logs into the system.
    2. Navigates to the "Manage Roles" section.
    3. Selects a user and assigns or updates their role.
    4. System validates the changes and updates the user's role in the database.
    5. System displays a confirmation message.
- **Error Flow:**
    - If the system encounters an error during the update, it logs the issue and displays an error message.
    - If the role is invalid or unavailable, the administrator is prompted to select a valid role.

3.1.2 Add New Student

Actor: Administrator
Description: The administrator adds a new student to the system.

Normal Flow:

1. Administrator logs into the system.
2. Navigates to the "Manage Students" section and selects "Add Student."
3. Administrator enters the student's details, including Name, Date of Birth, and other information.
4. Administrator submits the form.
5. System validates the input (e.g., ensuring required fields are filled, validating ID uniqueness).
6. System creates the student record in the database.
7. System displays a confirmation message that the student has been added successfully.

Error Flow:
- If required fields are missing, the system prompts the administrator to complete them.

3.1.3 Add New Teacher

Actor: Administrator
Description: Process of adding a new teacher to the system.
Normal Flow:

1. Navigates to the "Manage Teachers" section and selects "Add Teacher.".
2. System presents a comprehensive form for teacher details.
3. Administrator enters required information.
4. System validates the entered data and makes a teacher profile.
5. Administrator receives confirmation of successful teacher addition.

Error Flow:

- If any required field is left blank, system prompts administrator to complete it.

3.1.4 Set Class Timetable

Actor: Administrator
Description: The administrator creates or updates a class timetable.
Normal Flow:
1. Administrator logs into the system
2. Navigates to the "Timetable Management" section
3. Selects the class for which to set the timetable
4. Assigns subjects, teachers, and time slots for each day of the week
5. System saves the timetable
6. System displays a success message

3.1.5 Manage Student

Actor: Administrator
Description: The administrator manages an existing student's information in the system.
Normal Flow:

1. Administrator logs into the system
2. Navigates to the "Manage Students" section
3. Searches for a specific student using name, ID, or class
4. Selects the student record to manage
5. Updates student details as needed
6. System validates the input
7. System saves the updated student record
8. System displays a success message

Error Flow:

- If required fields are left empty after editing, the system prompts the administrator to complete them
- If the system encounters an error while saving the data, it displays an error message and logs the issue

3.1.6 Manage Teacher

Actor: Administrator
Description: The administrator manages an existing teacher's information in the system.
Normal Flow:

1. Administrator logs into the system
2. Navigates to the "Manage Teachers" section
3. Searches for a specific teacher using name, ID, or subject
4. Selects the teacher record to manage
5. Updates teacher details as needed
6. System validates the input
7. System saves the updated teacher record
8. System displays a success message

Error Flow:

- If required fields are left empty after editing, the system prompts the administrator to complete them
- If the system encounters an error while saving the data, it displays an error message and logs the issue

3.1.7 Remove Student

Actor: Administrator
Description: The administrator removes a student from the system.
Normal Flow:

1. Administrator logs into the system
2. Navigates to the "Manage Students" section
3. Searches for a specific student using name, ID, or class
4. Selects the student record to remove
5. Initiates the remove process

6. System prompts for confirmation
7. Administrator confirms the removal
8. System removes the student record and associated data
9. System displays a success message

Error Flow:
- If the system encounters an error during the removal process, it displays an error message.

3.1.8 Remove Teacher

Actor: Administrator
Description: The administrator removes a teacher from the system.

Normal Flow:
1. Administrator logs into the system
2. Navigates to the "Manage Teachers" section
3. Searches for a specific teacher using name, ID, or subject
4. Selects the teacher record to remove
5. Initiates the remove process
6. System prompts for confirmation
7. Administrator confirms the removal
8. System removes the teacher record and associated data
9. System displays a success message

Error Flow:
- If the system encounters an error during the removal process, it displays an error message.

3.2 Teacher Use Cases

3.2.1 Record Student Attendance

Actor: Teacher
Description: A teacher records attendance for a class.
Normal Flow:
1. Teacher logs into the system
2. Navigates to the "Manage Attendance" section
3. Selects the class and date
4. Marks each student as present or absent
5. Submits the attendance record
6. System saves the attendance data

Error Flow:

- If the teacher attempts to submit incomplete attendance data, the system prompts for completion

3.2.2 Input Student Grades

Actor: Teacher
Description: A teacher inputs grades for a specific assessment.
Normal Flow:
1. Teacher logs into the system
2. Navigates to the "Manage Grade" section
3. Selects the class, subject, and assessment type (e.g., quiz, midterm, final)
4. Enters grades for each student
5. Submits the grades
6. System saves the grade data

Error Flow:
- If entered grades are outside the valid range, the system alerts the teacher and prevents submission.

3.2.3 View Class Timetable

Actor: Teacher
Description: Process of viewing the teacher's class timetable.
Normal Flow:
1. Teacher logs into the system
2. Teacher selects "View Timetable" option from their dashboard.
3. System displays the class schedules.

3.3 Student Use Cases

3.3.1 View Academic Progress

Actor: Student
Description: Student checks their current academic standing.
Normal Flow:
1. Student selects "View Academic Progress" option.
2. System retrieves student's current enrolled classes.
3. Student can search a specific class for detailed view.
4. System presents graphical representation of student's performance over time.

Error Flow:
- If grades for a class are not yet available, the page will be empty.

3.3.2 View Personal Timetable

Actor: Student

Description: A student views their personal class timetable.
Normal Flow:
1. Student logs into the system
2. Navigates to the "View Timetable" section
3. System displays the timetable.

Error Flow:
- If the timetable hasn't been set or there's an error retrieving it, the system displays an appropriate message

4. Requirements Specification

4.1 External Interface Requirements

The EduSync system does not have direct external system interfaces. However, it will need to interface with the following:

1. **Database System**: EduSync will interface with a relational database management system (e.g., Microsoft SQL Database) to store and retrieve all system data.
2. **Email System**: For sending email verification to users, the system will interface with an SMTP server.

4.2 Functional Requirements

4.2.1 Add New Student

Use Case Name: Add New Student

XRef: Section 3.1.1, Add New Student

Trigger: Administrator selects "Add New Student" option in “Manage Students” section.

Precondition: Administrator is logged into the system

Basic Path:

1. System presents a form for entering student details
2. Administrator enters required information (name, ID, class, contact information, etc.)
3. System validates the input
4. System saves the new student record and creates a student profile
5. System displays a success message

Alternative Paths:

None

Exception Paths:
1. If any required field is left blank, the system prompts the administrator to complete it
2. If the system encounters an error while saving the data, it displays an error message and logs the issue
3. Administrator may abandon the process at any time

Postcondition: A new student record is created in the system.

4.2.2 Add New Teacher

Use Case Name: Add New Teacher
XRef: Section 3.1.2, Add New Teacher

Trigger: Administrator selects "Add New Teacher" option in "Manage Teachers" section.

Precondition: Administrator is logged into the system

Basic Path:
1. System presents a form for entering teacher details
2. Administrator enters required information (name, ID, subjects, qualifications, contact information, etc.)
3. System validates the input
4. System saves the new teacher record and creates a teacher profile
5. System displays a success message

Alternative Paths:
None

Exception Paths:
1. If any required field is left blank, the system prompts the administrator to complete it
2. If the system fails to create the teacher profile, it rolls back any partial changes and displays an error message.
3. Administrator may abandon the process at any time.

Postcondition: A new teacher record is created in the system.

4.2.3 Set Class Timetable

Use Case Name: Set Class Timetable

XRef: Section 3.1.3, Set Class Timetable

Trigger: Administrator selects "Manage Class Timetable" option

Precondition: Administrator is logged into the system, classes and teachers are already in the system.

Basic Path:
1. System presents a grid-like interface for the timetable
2. Administrator selects a class
3. For each time slot, administrator assigns a subject and a teacher
4. System checks for conflicts (e.g., teacher already assigned elsewhere at the same time)
5. Administrator submits the completed timetable
6. System saves the timetable

Alternative Paths:
1. Administrator sets recurring patterns (e.g., same schedule every Monday) to speed up the process

Exception Paths:
1. Administrator may abandon the process at any time

Postcondition: A new timetable is set for the selected class.

4.2.4 Record Student Attendance

Use Case Name: Record Student Attendance

XRef: Section 3.2.1, Record Student Attendance

Trigger: Teacher selects "Manage Attendance" option
Precondition: Teacher is logged into the system and Students already exists in the system.

Basic Path:
1. System displays the list of students in the selected class
2. Teacher marks each student as present or absent
3. Teacher submits the attendance record
4. System saves the attendance data

Alternative Paths:

None

Exception Paths:
1. If the teacher attempts to submit incomplete attendance data, the system prompts for completion
2. Teacher may abandon the process at any time

Postcondition: Attendance record for the class is updated.

4.2.5 Input Student Grades

Use Case Name: Input Student Grades
XRef: Section 3.2.2, Input Student Grades

Trigger: Teacher selects "Manage Grades" option
Precondition: Teacher is logged into the system and Courses and Students already exists in the system.

Basic Path:

1. System displays a list of students in the selected class
2. Teacher enters grades for each student
3. System validates that grades are within the acceptable range
4. Teacher submits the completed grade sheet
5. System saves the grade data

Alternative Paths:

None

Exception Paths:

1. If entered grades are outside the valid range, the system alerts the teacher and prevents submission
2. Teacher may abandon the process at any time

Postcondition: Grades for the selected assessment are recorded in the system.

4.2.6 View Class Timetable (Teacher)

Use Case Name: View Timetable (Teacher)

XRef: Section 3.2.3, View Timetable

Trigger: Teacher selects "View Timetable" option
Precondition: Teacher is logged into the system

Basic Path:

1. System retrieves the teacher's schedule
2. System displays a weekly view of the teacher's classes

Alternative Paths:

1. Teacher can filter the timetable to show only specific subjects or classes

Exception Paths:

1. If the timetable hasn't been finalized, the system displays a message indicating this
2. If the system fails to load the timetable, it displays an error message and offers to retry
3. Teacher may exit the view at any time

Postcondition: Teacher views their class schedule.

4.2.7 View Class Timetable (Student)

Use Case Name: View Timetable (Student)

XRef: Section 3.3.2, View Timetable

Trigger: Student selects "View Timetable" option
Precondition: Student is logged into the system

Basic Path:
1. System retrieves the student's class schedule
2. System displays a weekly view of the student's classes

Exception Paths:
1. If the student's timetable hasn't been assigned yet, the system displays an appropriate message
2. If the system fails to load the timetable, it displays an error message and offers to retry
3. Student may exit the view at any time

Postcondition: Student views their class schedule.

4.2.8 View Academic Progress

Use Case Name: View Academic Progress
XRef: Section 3.3.1, View Academic Progress

Trigger: Student selects "View Academic Progress" option
Precondition: Student is logged into the system

Basic Path:
1. System retrieves student's current enrolled classes
2. For each class, system displays current grade and attendance record
3. Student can select a specific class for detailed view
4. System presents graphical representation of student's performance over time

Alternative Paths:

1. Student can view progress reports for previous semesters/years

Exception Paths:
1. If the system fails to load the progress data, it displays an error message and offers to retry
2. Student may exit the view at any time

Postcondition: Student views their current academic standing.

4.2.9 Manage Student

Use Case Name: Manage Student
XRef: Section 3.1.4, Manage Student

Trigger: Administrator selects "Manage Students" option

Precondition: Administrator is logged into the system

Basic Path:
1. System presents a search interface for finding students
2. Administrator enters search criteria (name, ID, or class)
3. System displays matching student records
4. Administrator selects a student record to manage
5. System displays the student's information in an editable form
6. Administrator makes necessary changes
7. System validates the input
8. Administrator submits the changes
9. System saves the updated student record

Alternative Paths:
None

Exception Paths:
1. If any required field is left blank, the system prompts the administrator to complete it
2. Administrator may abandon the process at any time

Postcondition: The selected student's information is updated in the system

4.2.10 Manage Teacher

Use Case Name: Manage Teacher
XRef: Section 3.1.5, Manage Teacher

Trigger: Administrator selects "Manage Teachers" option

Precondition: Administrator is logged into the system

Basic Path:

1. System presents a search interface for finding teachers
2. Administrator enters search criteria (name, ID, or subject)
3. System displays matching teacher records
4. Administrator selects a teacher record to manage
5. System displays the teacher's information in an editable form
6. Administrator makes necessary changes
7. System validates the input
8. Administrator submits the changes
9. System saves the updated teacher record

Alternative Paths:

1. System provides an option to update multiple teachers' information based on department-wide changes

Exception Paths:

1. If any required field is left blank, the system prompts the administrator to complete it
2. If the system fails to save the changes, it retains the entered data and prompts the administrator to try again
3. Administrator may abandon the process at any time

Postcondition: The selected teacher's information is updated in the system.

4.2.11 Remove Student

Use Case Name: Remove Student
XRef: Section 3.1.6, Remove Student

Trigger: Administrator selects "Remove Student" option
Precondition: Administrator is logged into the system

Basic Path:

1. System presents a search interface for finding students
2. Administrator enters search criteria (name, ID, or class)
3. System displays matching student records
4. Administrator selects a student record to remove
5. System displays a confirmation dialog with the student's details
6. Administrator confirms the removal
7. System removes the student record and associated data
8. System displays a success message

Exception Paths:

1. If the system encounters an error during the removal process, it halts the operation and displays the error as pop up alert.
2. Administrator may abandon the process at any time

Postcondition: The selected student is removed from the system.

4.2.12 Remove Teacher

Use Case Name: Remove Teacher
XRef: Section 3.1.7, Remove Teacher

Trigger: Administrator selects "Remove Teacher" option

Precondition: Administrator is logged into the system

Basic Path:

1. System presents a search interface for finding teachers
2. Administrator enters search criteria (name, ID, or subject)
3. System displays matching teacher records
4. Administrator selects a teacher record to remove
5. System displays a confirmation dialog with the teacher's details
6. Administrator confirms the removal
7. System removes the teacher record and associated data
8. System displays a success message

Alternative Paths:

None

Exception Paths:

1. If the system encounters an error during the removal process, it halts the operation and displays the error as pop up alert.
2. Administrator may abandon the process at any time

Postcondition: The selected teacher is removed from the system.

4.2.13 Manage Roles

Use Case Name: Manage Roles
XRef: Section 3.1, Manage Roles

Trigger: Administrator selects "Manage Roles" option from the navigation menu.

Precondition:

- Administrator is logged into the system.
- At least one user is registered in the system.

Basic Path:

1. The system displays a list of all registered users, including their usernames, email addresses, and current roles.
2. The administrator selects a user to modify their role.
3. The system displays the user's current role and a dropdown menu with available roles (Admin, Teacher, Student).
4. The administrator selects a new role from the dropdown.
5. The system updates the user’s role in the database.
6. The system logs the role change event for auditing.
7. The system displays a success message to the administrator.

Alternative Paths:

1. **No Users Found:** If there are no users registered, the system displays a message indicating no users are available for role assignment.
2. **Role Already Assigned:** If the selected role is already assigned to the user, the system displays a message indicating the roles already assigned.

Exception Paths:

1. **Invalid Role Selection:**
    - If the administrator selects an invalid role (e.g., a role not defined in the system), the system displays an error message and prompts the administrator to select a valid role.
    - The system logs the error event.
2. **Database Error:**
    - If the system encounters an error while updating the role in the database, it displays an error message.
    - The system logs the error and advises the administrator to retry the operation.
3. **Session Timeout:**
    - If the administrator's session times out during the process, the system redirects to the login page and prompts the administrator to log in again.

Postcondition:

- The user's role is successfully updated in the system.
- The role change is logged in the audit trail for future reference.

4.3 Non-Functional Requirements

1. **Performance**:

    - The system should support up to 1000 concurrent users without significant degradation in performance
    - Page load times should not exceed 3 seconds under normal network conditions
2. **Usability**:
    - The user interface should be intuitive and require minimal training for users to become proficient
    - The system should be accessible on various devices, including desktops, tablets, and smartphones
3. **Reliability**:
    - The system should have an uptime of at least 99.9%
    - Data backups should be performed daily
4. **Scalability**:
    - The system should be able to handle a 50% increase in user base without requiring major architectural changes
5. **Maintainability**:
    - The system should be built using modular architecture to facilitate easy updates and maintenance.
    - Comprehensive documentation should be maintained for all system components.
6. **Compatibility**:
    - The web application should be compatible with the latest versions of major web browsers (Chrome, Firefox, Safari, Edge)

4.4 Security-Based Non-Functional Requirements

Based on the identified misuse cases and the threat model, the following security requirements are added to the **Non-Functional Requirements** section:

4.4.1 Authentication and Authorization

- The system must implement user authentication for all users (admin, teacher, student).
- Role-based authorization must ensure that only authorized users can access sensitive functions (e.g., grade modification, student/teacher record deletion).
- Two-factor authentication should be added for admins and teachers to prevent unauthorized access.

4.4.2 Data Integrity and Confidentiality

- All sensitive data, including student personal details and grades, should be encrypted both in transit and at rest.

- Ensure that **audit trails** are in place for all data modifications, especially for actions related to grades, attendance, and user records.

4.4.3 Non-Repudiation

- Implement **detailed logging** to track all changes made by users to ensure that no user can deny their actions.

4.4.4 Availability

- The system must implement rate-limiting and session management to prevent denial of service attacks.

4.4.5 Data Backup and Recovery

- Implement **automatic daily backups** of critical data (student records, grades, timetables) to minimize the impact of data loss during security incidents.

4.4.6 Cross-Site Scripting (XSS) Prevention

- All user inputs must be properly **sanitized and escaped** to prevent XSS attacks.
- A **Content Security Policy (CSP)** must be enforced to restrict where scripts can be loaded from.
- Error messages displayed to users should not expose sensitive data to avoid XSS payloads being reflected back to the browser.

4.4.7 SQL Injection Prevention

- All database queries should use **parameterized queries** or **stored procedures** to avoid direct embedding of user input in SQL statements.
- **Input validation** should be enforced to restrict the format of inputs for SQL queries.
- Detailed error messages about SQL operations should be hidden from the end-users to avoid leaking database structure information.

4.4.8 Session Security

- All sessions must be encrypted using **HTTPS** to protect session tokens during transmission.
- Implement **secure cookies** with the attributes `HttpOnly`, `Secure`, and `SameSite=Strict` to reduce the risk of session hijacking.
- Sessions should be **invalidated** after a period of inactivity, and users should be logged out.

4.4.9 Strong Input Validation

- The system should perform **both client-side and server-side validation** on all input fields.
- Input data should be restricted to valid formats through **whitelisting**.
- Ensure **error messages** do not reveal too much information that can be used by an attacker (e.g., database or system error details).

4.4.10 Logging and Monitoring

- Implement **comprehensive logging** for all important activities (e.g., grade modifications, student record changes, login attempts).
- Set up **real-time monitoring** for unusual activities, such as repeated login failures or unauthorized access attempts.
- Ensure **logs are protected from tampering** and are securely stored, with restricted access only to authorized administrators.

4.4.11 Regular Security Audits

- The system should undergo **regular security audits** to ensure compliance with security best practices and to identify potential vulnerabilities.
- Penetration tests should be performed to test the robustness of the system against attacks like SQL injection, XSS, and session hijacking.

5. Misuse Cases and Threat Model

This section outlines the misuse cases for the Edusync application based on the functionality provided for administrators, teachers, and students, along with the associated threats. The threat model follows the STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege) framework to identify and categorize potential threats.

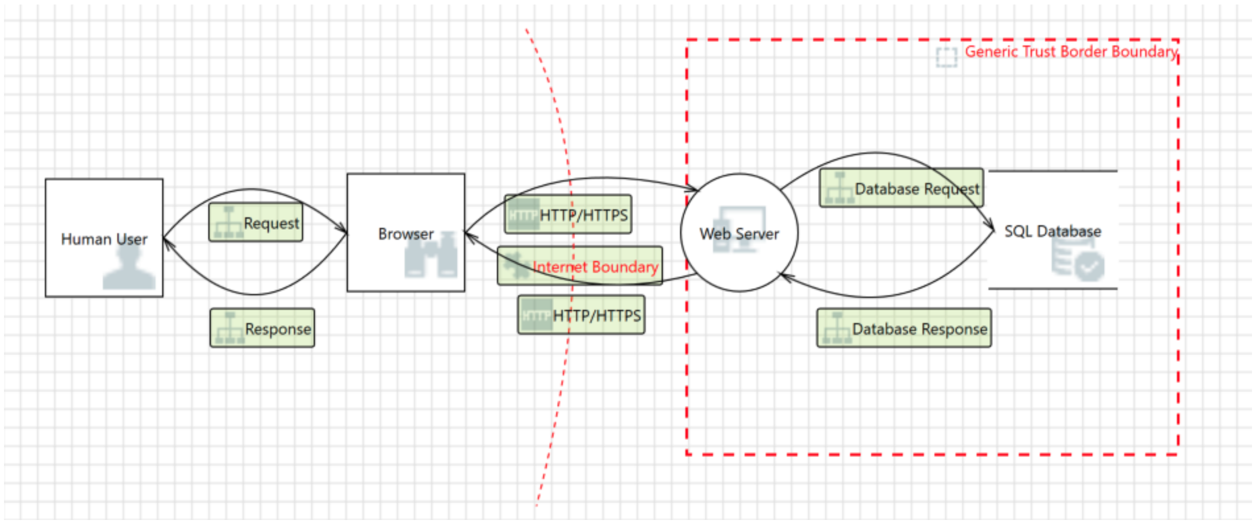

Fig 3: Threat Model diagram created in Microsoft TMT

5.1 Misuse Cases

5.1.1 Misuse Case: Unauthorized Access to Administrative Functions

- **Description**: A malicious actor attempts to gain access to the administrator's dashboard to add, remove, or modify teacher and student records or class timetables.
- **Impact**: If successful, the attacker can manipulate critical data, including deleting students or teachers, altering timetables, or modifying student grades, which can disrupt the academic system significantly.
- **Countermeasures**: Implement multi-factor authentication (MFA), role-based access control (RBAC), and session management to limit access to administrator-only functions.
- **Category**: Elevation of Privilege (E), Spoofing (S).

5.1.2 Misuse Case: Tampering with Grades

- **Description**: A malicious actor (either a student or an external attacker) gains unauthorized access to the grading system and alters the grades of one or more students.
- **Impact**: Compromising academic integrity, this could lead to incorrect academic evaluations, affecting a student's academic progress and potentially career opportunities.
- **Countermeasures**: Implement authentication, restrict access to the grade submission page only to authorized teachers, add audit trails for grade modifications, and encrypt the grade data to ensure integrity.
- **Category**: Tampering (T), Elevation of Privilege (E).

5.1.3 Misuse Case: Information Disclosure of Student Data

- **Description**: A malicious actor gains unauthorized access to student personal data, including contact information, grades, or attendance records.
- **Impact**: Disclosure of sensitive student information could lead to identity theft, privacy violations, and significant harm to students' personal lives.
- **Countermeasures**: Implement data encryption both in transit (SSL/TLS) and at rest, role-based access control to limit access to personal data, and ensure logging of all data access activities.
- **Category**: Information Disclosure (I).

5.1.4 Misuse Case: Denial of Service (DoS) Attack on Edusync Platform

- **Description**: A malicious actor performs a denial-of-service (DoS) attack, flooding the server with excessive requests to render the system unavailable to legitimate users.
- **Impact**: Students, teachers, and administrators will be unable to access the platform during the attack, disrupting classes, attendance tracking, and student progress monitoring.
- **Countermeasures**: Implement rate-limiting, use load balancers, and monitor traffic for suspicious activity. Ensure regular backups are taken to restore services quickly after an attack.
- **Category**: Denial of Service (DoS).

5.1.5 Misuse Case: Spoofing User Identity

- **Description**: A malicious actor pretends to be a legitimate user, such as a teacher or student, by hijacking session cookies or obtaining credentials through phishing.
- **Impact**: Spoofing could lead to unauthorized access to restricted areas like grade management or attendance records, allowing attackers to tamper with student data or impersonate teachers.
- **Countermeasures**: Implement strong password policies, enforce multi-factor authentication (MFA), use HTTPS for secure communications, and introduce session management with token expiration.
- **Category**: Spoofing (S), Elevation of Privilege (E).

5.1.6 Misuse Case: Repudiation of Actions

- **Description**: A user (teacher, student, or admin) denies performing actions such as modifying grades or updating timetables, claiming that unauthorized changes were made by someone else.
- **Impact**: This threatens the accountability of actions within the system, causing distrust in the platform's data integrity and accuracy.

- **Countermeasures**: Implement detailed logging for all actions performed by users. Use digital signatures or audit trails to link actions with user accounts for non-repudiation.
- **Category**: Repudiation (R).

5.1.7 Misuse Case: Cross-Site Scripting (XSS)

- **Description**: A malicious actor injects client-side scripts into web pages viewed by other users (students, teachers, or admins). This can happen through unvalidated user inputs.
- **Impact**: XSS attacks could allow attackers to steal session cookies, hijack user sessions, or manipulate the displayed content.
- **Countermeasures**:
    - Sanitize and escape all user inputs.
    - Implement Content Security Policy (CSP) to restrict where scripts can be loaded from.
    - Use secure frameworks that handle XSS attacks by default (e.g., ASP.NET Core MVC has built-in XSS protection).
- **Category**: Tampering (T), Information Disclosure (I), Spoofing (S).

5.1.8 Misuse Case: SQL Injection

- **Description**: A malicious actor injects SQL code into input fields (such as search boxes and login forms), potentially allowing them to manipulate or extract data from the database.
- **Impact**: An SQL Injection attack could allow the attacker to view, modify, or delete data from the database, bypassing authentication or compromising sensitive student and teacher information.
- **Countermeasures**:
    - Use parameterized queries and stored procedures to avoid dynamic SQL execution.
    - Validate all user inputs rigorously and avoid directly embedding user-supplied data into SQL queries.
    - Implement proper error handling to prevent exposing database details in case of a failure.
- **Category**: Tampering (T), Information Disclosure (I).

5.1.9 Misuse Case: Session Hijacking

- **Description**: An attacker steals a user's session token (often through XSS or network eavesdropping) and impersonates the user by gaining access to their session.

- **Impact**: If the attacker hijacks the session of an admin or teacher, they could gain unauthorized access to the system, modify student grades, change timetables, or delete records.
- **Countermeasures**:
    - Use **HTTPS** to secure all communications and prevent session tokens from being exposed in plaintext.
    - Implement **secure session management** practices, such as regenerating session tokens upon login and using **SameSite** cookies.
    - Implement **session timeout** and automatic logout after inactivity.
- **Category**: Spoofing (S), Elevation of Privilege (E).

5.1.10 Misuse Case: Insufficient Input Validation

- **Description**: Attackers may exploit input fields (e.g., student names, grades, or contact forms) that are not properly validated to inject malicious data or cause unexpected behavior (e.g., buffer overflow).
- **Impact**: Poor input validation could lead to security vulnerabilities, such as XSS, SQL Injection, or even system crashes.
- **Countermeasures**:
    - Ensure **input validation** on both the client and server sides.
    - Use whitelists for allowed characters and formats (e.g., dates, alphanumeric IDs).
    - Implement automatic error reporting and rejection of suspicious input patterns.
- **Category**: Tampering (T), Information Disclosure (I).

5.1.11 Misuse Case: Insufficient Logging and Monitoring

- **Description**: The system lacks proper logging and monitoring, making it difficult to detect and respond to malicious activities such as unauthorized access, data tampering, or other security breaches.
- **Impact**: Delayed response to security incidents could lead to greater data breaches, loss of student/teacher records, or integrity issues with student grades.
- **Countermeasures**:
    - Implement **comprehensive logging** for all critical actions (e.g., login attempts, grade changes, timetable updates).
    - Set up real-time **monitoring** for unusual activities and trigger alerts for anomalies.
    - Ensure logs are tamper-evident and stored securely to prevent unauthorized access.
- **Category**: Repudiation (R), Information Disclosure (I).

5.2 Threat Model (STRIDE)

The following threat model for the Edusync platform uses STRIDE to categorize threats across different components of the system.

Actors

1. **Administrator**: Manages student, teacher, and class data.
2. **Teacher**: Manages grades, attendance, and class timetables.
3. **Student**: Views academic progress, grades, and personal timetable.

Assets

1. **Student Data**: Names, grades, attendance records, personal details.
2. **Teacher Data**: Personal details, schedules, and assigned classes.
3. **Grades**: Data representing students' academic progress.
4. **Class Timetables**: Data representing schedules for students and teachers.

Threat Surface

1. **Web Application Interface**: Main interface for all actors.
2. **Database (Microsoft SQL Server)**: Stores all user, grade, and timetable data.
3. **Data Transmission**: Between the client (web browser) and server.
4. **Session Management**: Handles user authentication and persistence.

STRIDE Analysis

Threat	Category	Description	Impact	Countermeasures
Unauthorized access to admin functions	Spoofing (S), Elevation of Privilege (E)	Spoofed credentials or privilege escalation allows unauthorized access to the admin panel.	Compromises the system by allowing unauthorized users to modify critical data, impacting academic integrity.	Implement multi-factor authentication (MFA), role-based access control (RBAC), and proper session handling.
Tampering with student grades	Tampering (T), Elevation of Privilege (E)	Attacker modifies grades through unauthorized access.	Affects academic integrity by allowing grade manipulation.	Limit access to grade management functions to authenticated teachers, implement encryption, and provide audit trails.

Disclosure of student data	Information Disclosure (I)	Personal information of students is disclosed to unauthorized users.	Violates student privacy and potentially leads to identity theft.	Encrypt sensitive data, implement role-based access, and provide detailed audit logs for data access.
Alteration of class timetable	Tampering (T), Denial of Service (DoS)	Unauthorized modification of class schedules disrupts the academic process.	Causes confusion for students and teachers due to overlapping or incorrect class schedules.	Restrict timetable management to authenticated administrators, use tamper-evident logs.
Denial of Service attack	Denial of Service (DoS)	Flooding the server with requests to render the platform unusable.	Disrupts access to the system for students, teachers, and administrators.	Implement rate-limiting, load balancing, and monitoring of incoming traffic for anomalies.
Spoofing user identity	Spoofing (S)	Attacker impersonates a legitimate user to gain unauthorized access.	Results in unauthorized access to sensitive data or actions performed in another user's name.	Implement MFA, HTTPS for secure communications, and session management to prevent session hijacking.
Repudiation of user actions	Repudiation (R)	Users deny making changes or performing certain actions.	Affects accountability and trust in the system’s data integrity.	Implement non-repudiation measures such as logging, digital signatures, and linking all actions to user accounts.

Threat	Category	Description	Impact	Countermeasures
Cross-Site Scripting (XSS)	Information Disclosure, Spoofing, Tampering	Malicious actors inject client-side scripts into web pages viewed by	Hijacking of user sessions, data theft, or content manipulation leading to user	Sanitize and escape user inputs, implement Content Security Policy (CSP) to limit script sources, and use

		other users. These scripts could be used to steal session cookies or manipulate content.	impersonation and unauthorized actions.	secure frameworks with built-in XSS protection.
SQL Injection	Tampering, Information Disclosure	Attackers inject SQL code into input fields, enabling them to manipulate or extract data from the database.	Unauthorized viewing, modification, or deletion of data, potential bypassing of authentication controls.	Use parameterized queries and stored procedures, validate inputs rigorously, and implement error handling to avoid database detail exposure.
Session Hijacking	Spoofing, Elevation of Privilege	Attackers steal a user's session token (often through XSS or network sniffing) and impersonate the user by accessing their session.	Unauthorized access to the system, data manipulation, or impersonation of admin or teacher accounts.	Use HTTPS to secure session tokens, regenerate session tokens on login, implement secure cookies (HttpOnly, Secure, SameSite), and auto-expire sessions.
Insufficient Input Validation	Tampering, Information Disclosure	Lack of robust input validation can lead to injection attacks or system crashes.	Data manipulation, potential exploitation of vulnerabilities, and system instability.	Enforce strong input validation on both client and server, use whitelists for allowed characters/formats, and ensure automatic error handling for invalid inputs.
Insufficient Logging and Monitoring	Repudiation, Information Disclosure	Lack of logging and monitoring makes it difficult to detect and respond to	Delayed response to security incidents, potentially leading to larger data breaches	Implement comprehensive logging for critical actions, set up real-time monitoring and alerts for unusual activity, and secure

		malicious activities.	or data integrity issues.	logs to prevent tampering.

5.3 STRIDE Threat Model Summary for Edusync

Based on the analysis of misuse cases, here is a high-level mapping of threats according to the STRIDE model:

Category	Threat Description
Spoofing (S)	Attackers may impersonate legitimate users, including admin, teacher, or student accounts, potentially leading to unauthorized access.
Tampering (T)	Attackers may modify data, including grades, attendance records, and timetables, without authorization.
Repudiation (R)	Users may deny performing actions (e.g., grade changes), and without sufficient logging, the system may fail to prove their accountability.
Information Disclosure (I)	Sensitive information (e.g., student grades, personal details) may be disclosed to unauthorized users due to poor access control or weak encryption.
Denial of Service (DoS)	The system could be overwhelmed by high traffic, rendering it inaccessible to legitimate users during key academic periods.
Elevation of Privilege (E)	Attackers may gain unauthorized access to privileged functions, such as modifying grades, deleting records, or accessing administrative features.

5. References

Lane, G. Krüger and Charles. (2023, June 17). *How to write an SRS document (software requirements specification document)*. Perforce Software. https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document

IEEE. (1984, February 10). *830-1984 - IEEE Guide for Software Requirements Specifications*. IEEE Xplore. https://ieeexplore.ieee.org/document/278253

Lusov, Kyrylo. *Software Requirements Specification Example and Guide*. Jelvix. https://jelvix.com/blog/software-requirements-specification