Software Requirements Specification

for

Edubuddy Platform

Prepared by:
Sanskriti Saxena & Arnav Rathi
School of Computer Science & Engineering
Manipal University Jaipur

October 16, 2025

Version 1.0

Contents

Revision History

Name	Date	Reason For Changes	Version
Sanskriti Saxena Arnav Rathi	20/08/2025	Initial Draft with Student and Teacher Roles	1.0

1 Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document defines the functional and non-functional requirements for **Edubuddy**, version 1.0. This product is a web-based application designed to be a comprehensive academic ecosystem for both students and teachers. Its purpose is to provide a centralized, reliable, and collaborative platform for sharing educational resources, facilitating peer-to-peer learning, and enabling instructor guidance and material verification within an educational institution.

1.2 Document Conventions

This document adheres to the IEEE 830-1998 standard for SRS documentation. All functional requirements are uniquely identified with a tag of the format REQ-X for traceability. The term ”The system” refers to the Edubuddy software.

1.3 Intended Audience and Reading Suggestions

This document is intended for the following audiences:

- **Developers (Project Team):** To understand the complete feature set that needs to be implemented.
- **Project Manager:** To oversee the project’s scope and progress against the defined requirements.
- **Testers:** To develop test cases and quality assurance plans based on the specified requirements.
- **Stakeholders (Students, Teachers):** To review the proposed functionality and ensure it meets their academic needs.

It is recommended to read the document sequentially, paying special attention to the distinct user classes in Section 2.3 and the feature sets in Section 4, which detail role-specific functionalities.

1.4 Product Scope

Edubuddy aims to create a structured digital environment that enhances the learning and teaching experience. The key goals are:

- To build a central repository for course notes, study guides, and other academic materials.
- To distinguish between peer-generated content and instructor-verified materials, adding a layer of authenticity.
- To provide an organized, searchable, and filterable database of all academic resources.
- To foster a community where students can clarify doubts and teachers can provide authoritative answers.
- To reduce the time and effort required for both students and teachers to find and distribute quality educational content.
- To provide a centralized platform for tracking academic progress and managing assignments.

1.5 References

1. *Edubuddy GitHub Repository: https://github.com/Sanskriti0109/Edubuddy*.

2. *Wiegers, Karl E. Software Requirements Specification Template, 1999*.

3. *MERN Stack (MongoDB, Express.js, React, Node.js) Official Documentation*.

2 Overall Description

2.1 Product Perspective

Edubuddy is a new, self-contained product that will function as a standalone web application. It is designed to be a multi-user platform with role-based access, serving as a vital connection point between students seeking knowledge and instructors providing it, and managing various aspects of academic life.

2.2 Product Functions

The major functions of the system are:

- Role-based user registration and authentication.
- Uploading, managing, and sharing of educational resources by all users.
- Verification and endorsement of content by Teacher users.

2.3 User Classes and Characteristics

User Class	Characteristics
Student	The primary consumer and contributor of content. Students can register, log in, manage their profile, upload their notes, download resources shared by peers and teachers, access course materials, and participate in the Q&A forum by asking and answering questions. They are expected to have basic web literacy.
Teacher / Instructor	A privileged user responsible for content quality and guidance. Teachers have all the capabilities of a student but can also upload official course materials, verify the accuracy of student-uploaded notes, provide authoritative answers in the Q&A forum, and manage course materials. This role requires subject matter expertise.

2.4 Operating Environment

- **Client-Side:** The system shall be a responsive web application accessible via modern web browsers (e.g., Google Chrome, Mozilla Firefox, Safari, Microsoft Edge) on both desktop and mobile devices.
- **Server-Side:** The application is developed using the MERN stack (MongoDB, Express.js, React, Node.js) and is designed for deployment on a cloud platform (e.g., Vercel, Heroku, AWS).

2.5 Design and Implementation Constraints

- The project must be developed using the specified MERN stack.
- All development must be version-controlled using Git and hosted on the project's GitHub repository.
- The user interface must be intuitive and responsive on both desktop and mobile devices.
- The system must implement role-based access control to secure teacher-specific functionalities and protect sensitive data.
- The application depends on a configured MongoDB instance for data persistence.

2.6 User Documentation

- A detailed README.md file in the GitHub repository will provide instructions for project setup, configuration, and deployment.

2.7 Assumptions and Dependencies

- It is assumed that there will be a mechanism (manual or automated) to assign the ”Teacher” role to legitimate instructors, ensuring the integrity of verified content.
- The system's performance depends on the stability of the hosting provider and the MongoDB service.
- Users are assumed to have a stable internet connection.
- Third-party API integrations (e.g., for calendar, AI-generated quotes) are assumed to be stable and available.

3 External Interface Requirements

3.1 User Interfaces

The system will feature a clean and modern Graphical User Interface (GUI). Key screens will include:

- **Homepage:** An overview of the platform and its features, potentially showcasing recent activities or verified content.
- **Login/Registration Pages:** Secure forms for user authentication, including role selection during registration.
- **Resource Dashboard:** A central feed displaying all resources, with clear labels for verified or official content.
- **Upload Form:** A modal or page for uploading files, their descriptions, and assigning subjects/tags.
- **User Profile Page:** For managing personal information and viewing one's own contributions.

3.2 Hardware Interfaces

No direct hardware interfaces are required. The application interacts with standard client devices (desktops, laptops, mobile phones) through their web browsers.

3.3 Software Interfaces

- **Database:** The system will interface with a MongoDB database via the Mongoose ODM (Object Data Modeling) for all data persistence and retrieval operations.
- **Node.js Environment:** The backend relies on the Node.js runtime and various npm packages (e.g., Express.js for routing, JWT for authentication, bcryptjs for password hashing, multer for file uploads) for its operation.
- **External Calendar API:** Potential integration with a calendar service (e.g., Google Calendar API) for AC Calendar Integration.
- **AI Quote Generation API:** Potential interface with a third-party API for generating AI quotes.

3.4 Communications Interfaces

All communication between the client (web browser) and the server (backend API) will occur over the standard HTTPS protocol to ensure data encryption, integrity, and security. The frontend will interact with the backend via a RESTful API, sending and receiving JSON data.

4 Functional Requirements

This section outlines the key functional requirements for the Edubuddy platform.

4.1 User Authentication and Authorization System

4.1.1 Description and Priority

This feature provides secure mechanisms for users to register, log in, and manage their access based on their roles (Student or Teacher). Priority: **High**.

4.1.2 Requirements

REQ-1 The system shall allow users to register with a unique email, name, password, and designated role (Student or Teacher).

REQ-2 The system shall allow users to manage their profile information (e.g., update name, change password).

4.2 Academic Resource Management

4.2.1 Description and Priority

This feature enables users to upload, view, download, search, and manage educational resources, distinguishing between peer-contributed and teacher-verified content. Priority: **High**.

4.2.2 Requirements

REQ-3 Any authenticated user shall be able to upload a resource (e.g., notes, study guides) with a title, description, subject/tag, and associated file.

REQ-4 All users shall be able to view and download any shared resource.

REQ-5 Resources uploaded by Teachers and resources verified by Teachers shall be visually distinguished in the user interface.

REQ-6 Teachers shall be able to upload and manage official course materials.

REQ-7 The system shall provide a Canvas Board functionality for collaborative brainstorming or visual explanations.

4.3 Search and Filtering Capabilities

4.3.1 Description and Priority

This feature allows users to efficiently locate specific resources or information within the platform. Priority: **High**.

4.3.2 Requirements

REQ-8 The system shall provide a comprehensive search bar to find resources based on keywords in the title, description, or subject/tag.

4.4 Mobile-Responsive Design

4.4.1 Description and Priority

This feature ensures the application is accessible and usable across various devices and screen sizes. Priority: **High**.

4.4.2 Requirements

REQ-9 The system's user interface shall be fully responsive, adapting seamlessly to different screen sizes on desktop, tablet, and mobile devices.

REQ-10 All functionalities available on desktop shall be accessible and usable on mobile devices without degradation of experience.

5 Non-Functional Requirements

This section details the quality attributes and constraints that the Edubuddy platform must satisfy.

5.1 Performance

5.1.1 Description and Priority

The system's responsiveness and efficiency under various loads. Priority: **High**.

5.1.2 Requirements

NFR-1 Page load times shall be under 5 seconds on a standard broadband connection.

5.2 Compatibility

5.2.1 Description and Priority

The system's ability to function correctly across different environments. Priority: **High**.

5.2.2 Requirements

NFR-2 The system shall support all major modern web browsers, including Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge.

NFR-3 The system shall provide a consistent and fully functional user experience across various devices (desktops, tablets, mobile phones) due to its mobile-responsive design.

5.3 Maintainability

5.3.1 Description and Priority

The ease with which the system can be modified, corrected, or enhanced. Priority: **Medium**.

5.3.2 Requirements

NFR-4 The codebase shall be modular, well-commented, and adhere to consistent coding standards to facilitate future modifications and bug fixes.

6 Other Requirements

6.1 Business Rules

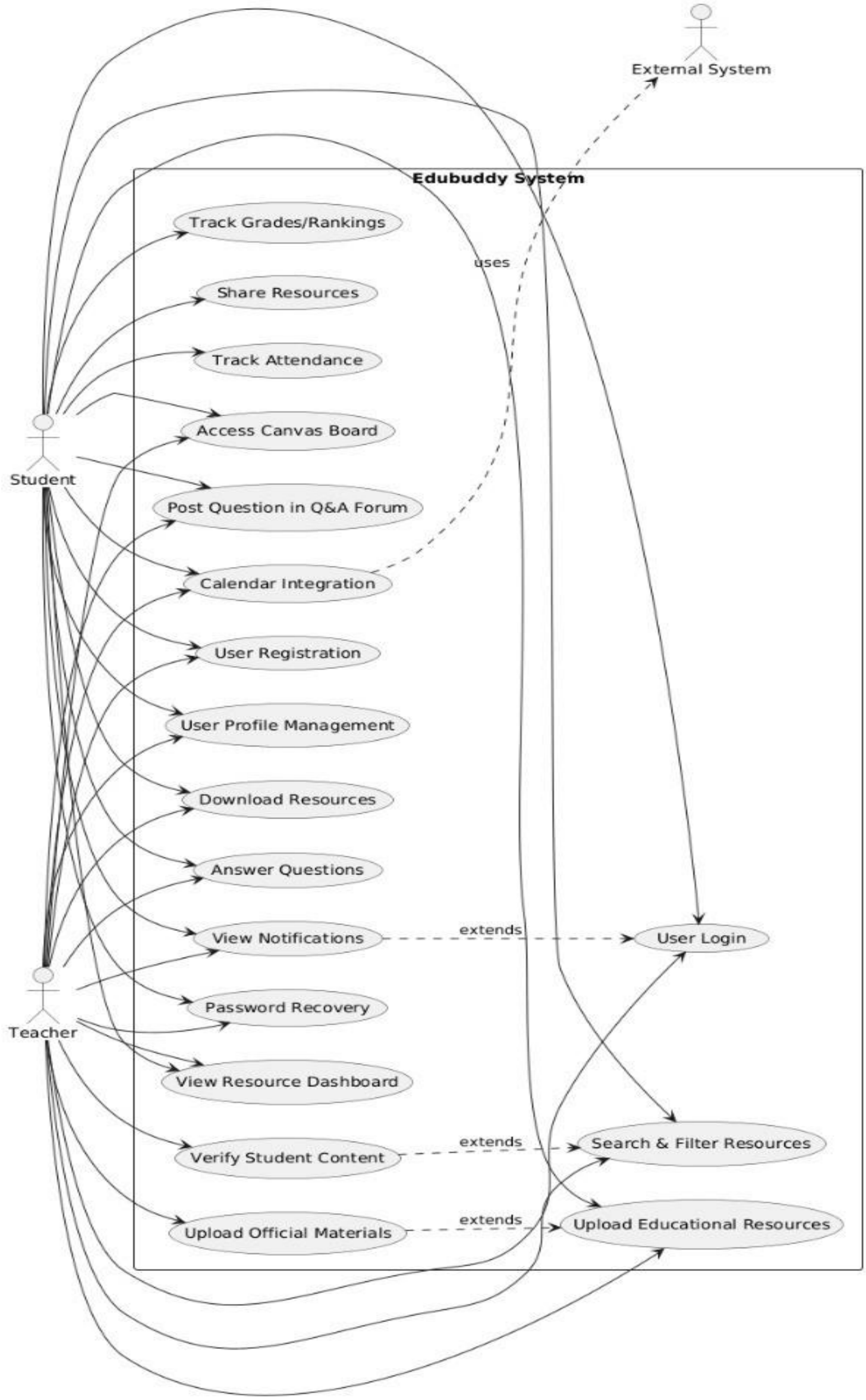
- A user must be authenticated to contribute content (upload resources, ask questions, post replies).
- The ”Teacher” role can only be assigned through a trusted administrative process to maintain the integrity and authority of content verification and official course materials.
- Users are responsible for adhering to copyright laws and academic honesty policies when uploading and sharing materials.
- The platform acts as a facilitator for resource sharing and communication; it is not responsible for the content of user-uploaded files beyond the teacher verification process.
- Content flagged by users or identified as inappropriate by administrators will be reviewed and potentially removed.
- Resource listings will remain active indefinitely unless deleted by the owner or an administrator.

A Analysis Models

This appendix contains the analysis models for the Edubuddy platform. These models provide a visual representation of the system’s structure, behavior, and data flow.

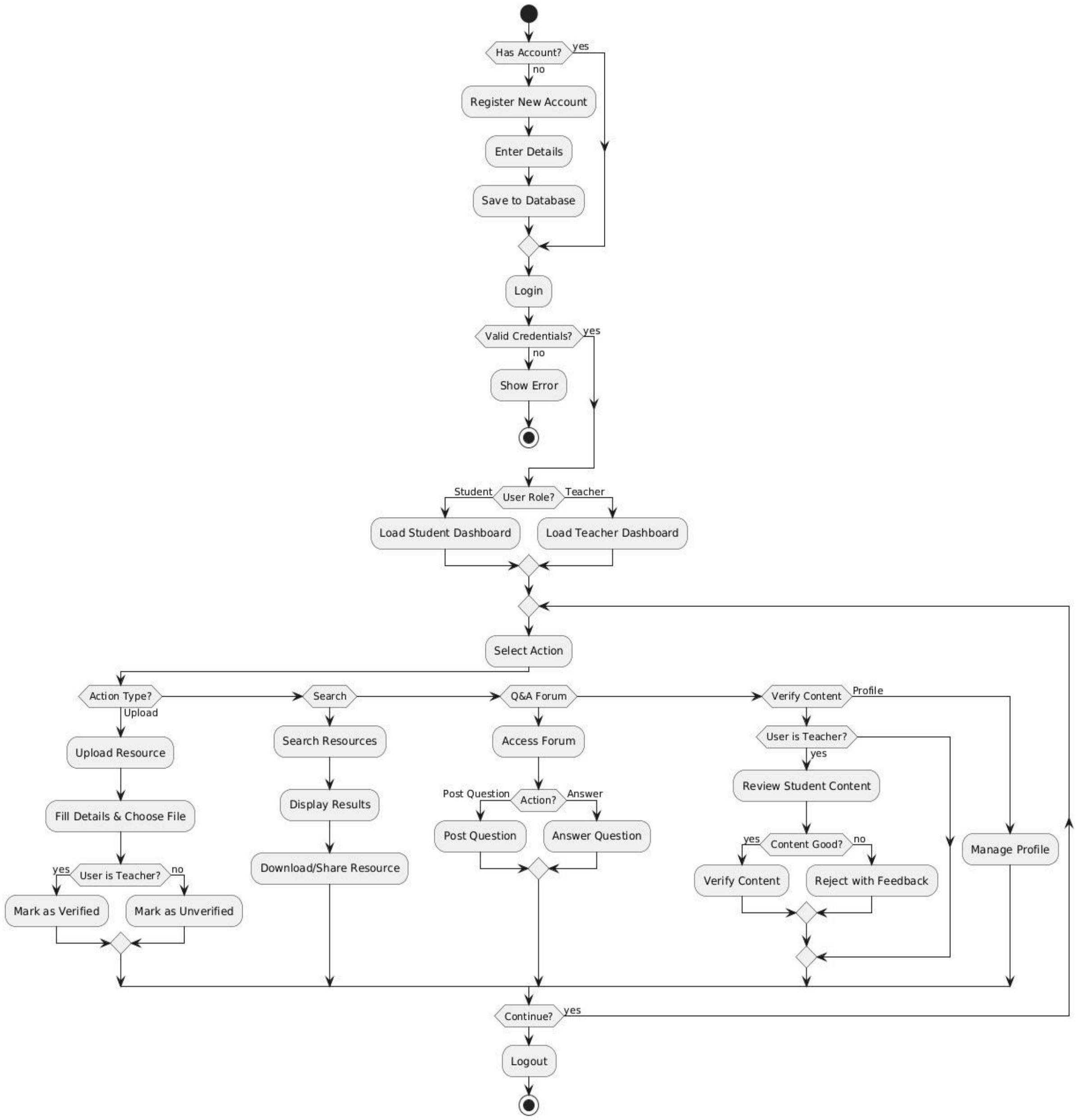
A.1 Use Case Diagram

The Use Case Diagram illustrates the functional requirements of the system by showing the interactions between actors (Student, Teacher, External System) and the system's use cases.



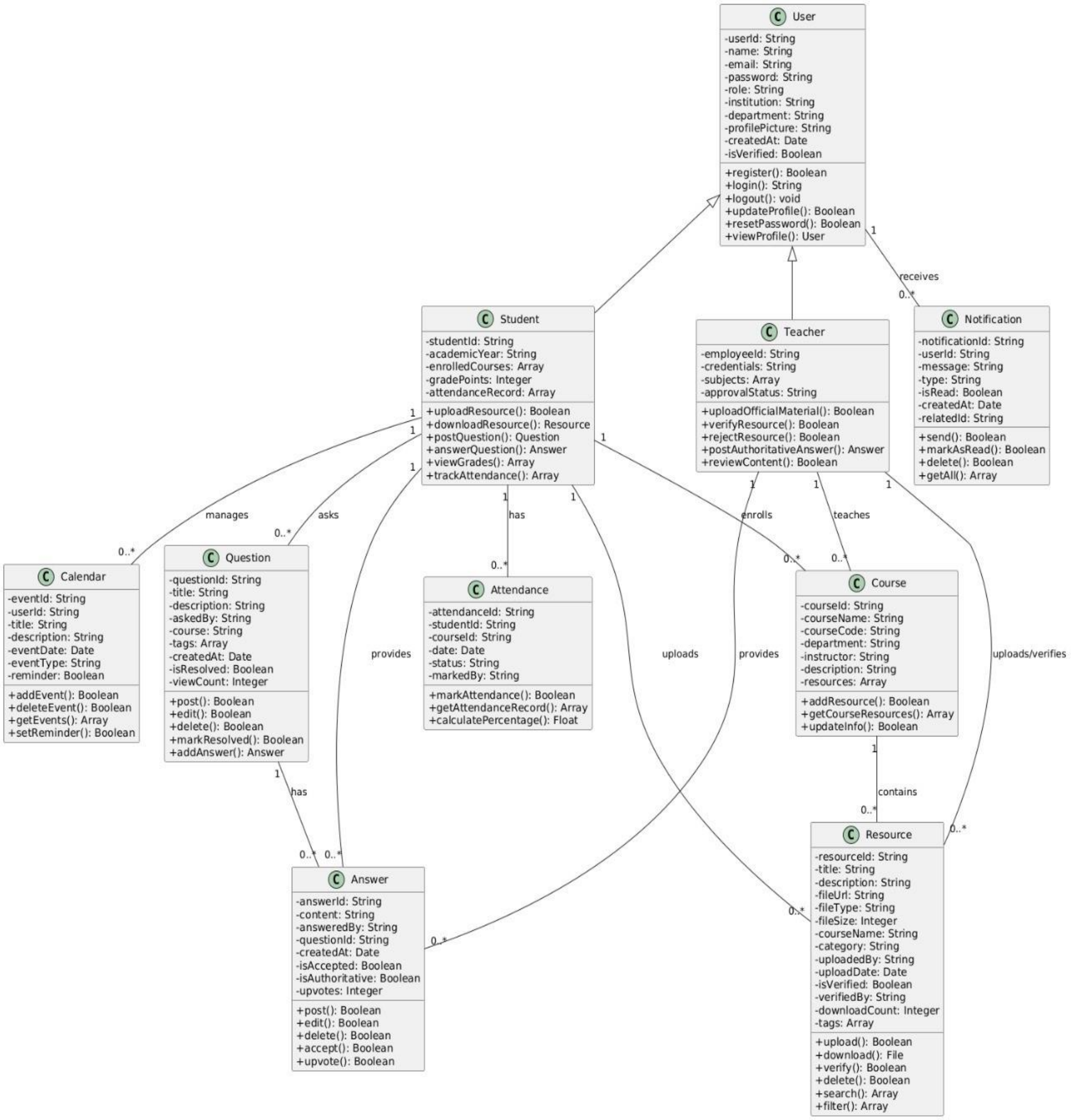
A.2 Activity Diagram

The Activity Diagram depicts the workflow and business process logic within the Edubuddy system, showing the flow from user registration through various system operations.



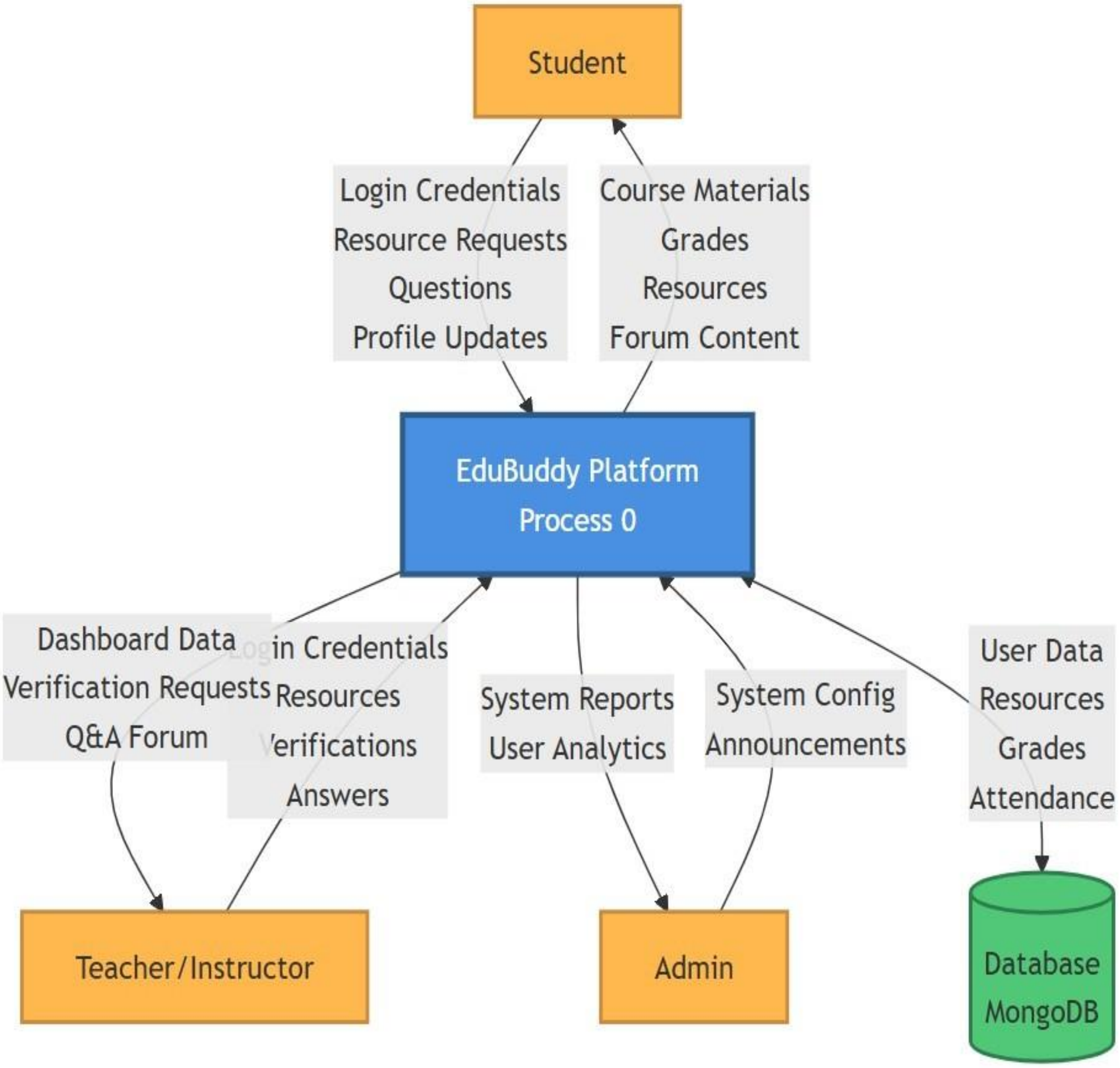
A.3 Class Diagram

The Class Diagram shows the static structure of the system, including the main entities (User, Student, Teacher, Resource, Question, Answer, etc.) and their relationships.



A.4 Data Flow Diagram - Level 0 (Context Diagram)

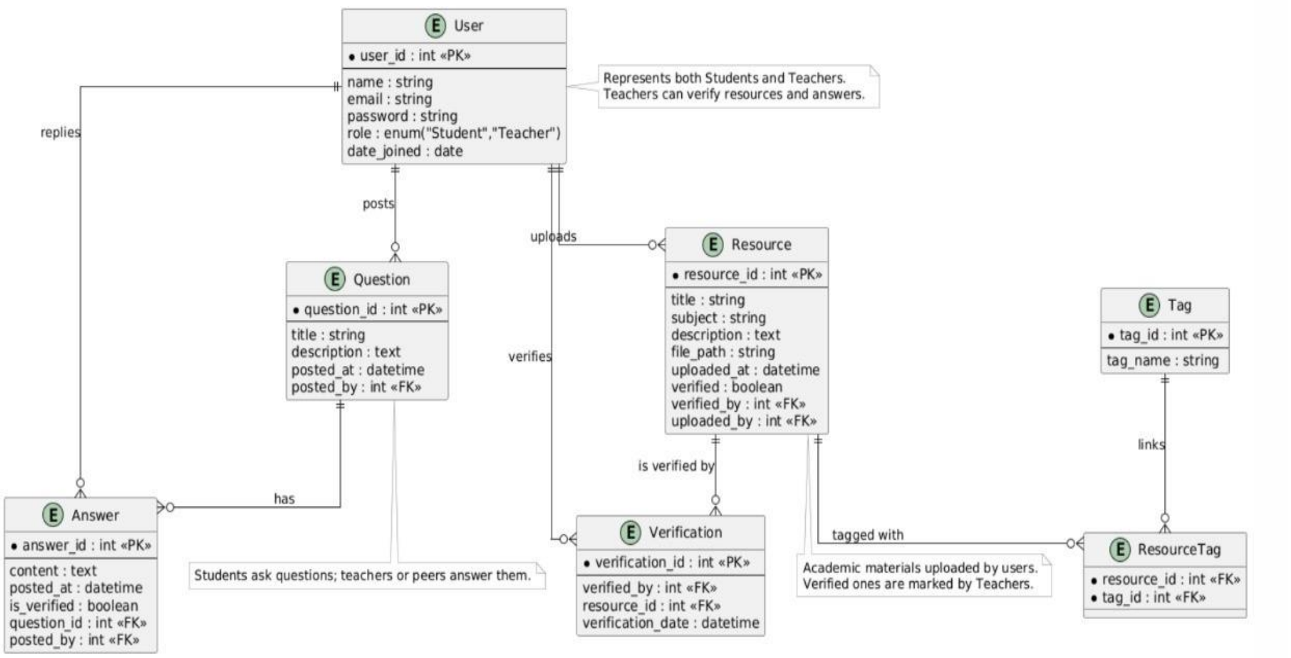
The Context Diagram (DFD Level 0) shows the Edubuddy system as a single process, illustrating the data flow between external entities and the system.



A.5 Entity Relationship Diagram (ERD)

The ERD shows the database schema and relationships between different entities in the Edubuddy system, including users, resources, courses, grades, and attendance.

EduBuddy - Entity Relationship Diagram



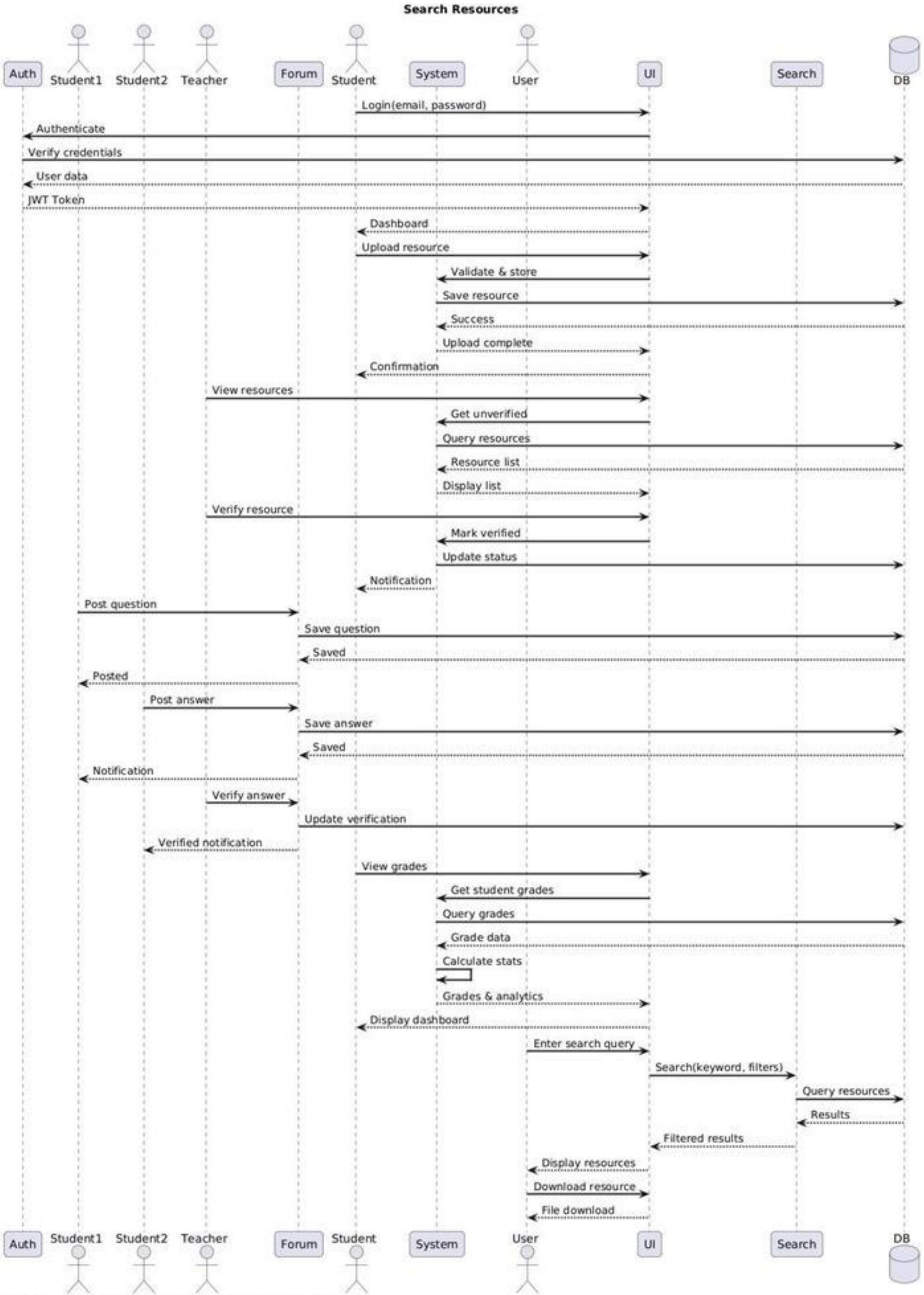
Represents both Students and Teachers.
Teachers can verify resources and answers.

Students ask questions; teachers or peers answer them.

Academic materials uploaded by users.
Verified ones are marked by Teachers.

A.6 Sequence Diagram

The Sequence Diagram illustrates the interaction between different system components over time, showing the message flow for key operations like resource upload, verification, and Q&A interactions.



B To Be Determined List

1. **TBD-1 (Teacher Role Assignment):** The specific process for verifying and granting ”Teacher” privileges to legitimate instructors within the university. This includes who can assign roles and the validation steps.

2. **TBD-2 (Content Verification Criteria):** The exact criteria and weighting used by the system (if any automated checks are implemented) or guidelines for Teachers when ”verifying” student-uploaded resources, ensuring consistency and quality.

3. **TBD-3 (AI Quote Integration Details):** The specific third-party API to be used for AI-generated quotes, including API keys, rate limits, and error handling strategies.

4. **TBD-4 (Calendar Integration Details):** The specific external calendar service (e.g., Google Calendar, Outlook Calendar) to be integrated for AC Calendar functionality, including authorization flows and data synchronization.