BILKENT UNIVERSITY

CS491 - ANALYSIS AND REQUIREMENTS REPORT

T2423 - Evalio

Eren Hayrettin Arım	22002306
Ahmet Reşat Demir	22002299
Mehmet Burak Demirel	22003396
Dilara Mandıracı	22101643
Yusuf Toraman	22002885

Table of Contents

1. Introduction

Even if the traditional paper-based exam method is widely used, it often falls short of accurately measuring student success and meeting modern educational needs. Research from Purdue University shows the importance of mapping and analyzing questions during the exam preparation phase to enhance exam quality and performance [1]. In response, our team aims to develop Evalio, an advanced exam database and management application designed to streamline exam creation and improve assessment effectiveness. Evalio enables instructors to create practical, high-quality exams through a shared, searchable question database that supports multiple courses. With features like question usage tracking, editing, versioning, export options, and detailed performance analytics, Evalio helps create balanced assessments while managing assignments and exams. Additionally, it offers secure, multi-university collaboration. It ultimately contributes to improved teaching and learning outcomes.

2. Proposed System

2.1. Overview

The Evalio platform aims to ease multiple aspects of the examination, objection to exam results, and document the course information. A multi-institutional designed question database allows the sharing and storing of questions and exams among different universities. Authorized entities (i.e., instructors, teaching assistants, and university administrative staff) can later create detailed analysis reports and ABET reports. Additionally, the system aims to provide a robust objection session manager to arrange objection sessions efficiently.

2.2. Functional Requirements

The functional requirements of this project focus on creating a secure, efficient, and user-friendly system to manage question databases, build exams, analyze performance records, and organize exam result viewing sessions. Each module is designed to streamline workflows for instructors, teaching assistants, and students. The system also supports accreditation needs and inter-university collaboration.

2.2.1. Questions Database and Exam Builder:

- Maintain a question database for each course, categorized by type and topic classification.
- Instructors can add new questions to the database.
- Log the history of question usage, including statistical data.
- Allow the addition, editing, and deprecation of questions.
- Implement permission controls to restrict or allow access to specific course-related question pools.
- Provide an exam builder that supports the following:
  - Question selection and arrangement.
  - Editable space and points allocation for questions.
  - Header section customization (course name, date, name/ID fields, score table).
  - Export to LaTeX, Google Docs, or other visual formatting editors, with optional PDF export.
  - In-editor question editing and creation.
- Enable instructors to build modular exams by selecting questions from the database using filters such as difficulty level, topic, and type.
- Include functionality to auto-generate balanced exams based on predefined criteria, such as equal topic distribution or desired difficulty levels.

2.2.2. Analysis Tools

- University and course instructors can create custom reports based on selected metrics:
  - Correlation between attendance and academic performance.
  - Success rate per question.
  - Success rate comparison with other universities per question.
  - etc.…
- University and course instructors can create ABET/MÜDEK reports according to global standards.

2.2.3. Portfolio Builder

- Enable TAs to upload scans of selected exams (best, average, worst), quizzes, homework, projects, and exam seating plans.
- Upload attendance sheets and grading files.
- Generate course summary forms from uploaded and system data.
- Export a merged PDF containing all portfolio elements, including exam solution keys.

2.2.4. Exam Result Viewing Session Organizer:

- Enable teaching assistants to create exam result viewing sessions.
- Enable teaching assistants to create time slots for created sessions, which allows configurable numbers.
- Allow teaching assistants to cancel or rearrange sessions.
- Allow users to check schedules via the institution's local system to avoid conflicts.
- Allow students to access the session with the provided session key and password.
- Announce and manage session sign-ups on a first-come, first-served basis.

2.3. Nonfunctional Requirements

2.3.1. User Friendliness / Usability

Evalio aims to be user-friendly with its easy and understandable interface designed to facilitate users while creating exams by taking advantage of the modularity of the questions. We know that currently, instructors can create exams using different tools, so this product focuses on providing advanced ease of use. Therefore, the user interface should be as simple as possible while providing the functionality of the basic modules.

2.3.2. Maintainability

Since Evalio targets various types of end users, such as instructors, students, and teaching assistants, and includes different modules, we aim to make it as sustainable as possible by splitting the architecture into services. A modular codebase will enable isolated updates without breaking other services or modules.

2.3.3. Reliability

Evalio aims to be reliable. Given the variety of end-user types and use cases, a single point of failure should not affect other modules, and any failure should be quickly recoverable. We aim to increase reliability by splitting the application's architecture into separate services.

2.3.4. Security

Since we aim to support different user types (e.g., instructors and institutions), the application should include a user authentication and authorization system to ensure that each user type has the correct permissions to access specific modules. Additionally, passwords and potentially sensitive information should be stored in a coded form in the database to ensure security.

2.4. System Models

2.4.1. Scenarios

General Scenarios

Scenario 1: Creating a Question in the Database

- **Participating Actors:** Instructor, System
- **Entry Conditions:**
    - The instructor is logged in and authenticated.
    - The instructor has access to the relevant course.
- **Exit Conditions:**
    - The new question is successfully added to the database with all required metadata.
- **Main Flow Events:**
- The instructor navigates to the "Questions Database" module.
- The instructor selects the option to add a new question.

    - The system displays a form to enter question details:
    - Question text
    - Keywords or topic classification
    - Type (quiz/exam/both)
    - Recommended points and space for answers
    - Solution key
    - The instructor fills in the form and submits it.
    - The system validates the data and saves the new question in the database.
- **Alternative Flows:**
    1) **Invalid Data Input:**
    - The system notifies the instructor of missing or incorrect fields.
    - The instructor corrects the fields and resubmits.

Scenario 2: Building an Exam

- **Participating Actors:** Instructor, System
- **Entry Conditions:**
    - The instructor is logged in and authenticated.
    - A course and its question database exist.
- **Exit Conditions:**
    - The exam is created successfully and exported in the desired format.
- **Main Flow Events:**
    - Instructor accesses the "Exam Builder" module.
    - The instructor selects a course and retrieves its question database.
    - The system displays the list of available questions with filters (e.g., topic, difficulty).
    - The instructor selects questions to include in the exam.
    - The instructor customizes points, space allocation, and the header section.
    - The instructor previews the exam and selects the export format (LaTeX, Google Docs, PDF).
    - The system generates and provides the exam file.
- **Alternative Flows:**

1) **Insufficient Questions:**
   - The system notifies the instructor if selected questions need to meet the desired exam criteria.
2) **Invalid Export:**
   - If the export fails, the system prompts the instructor to retry.

Scenario 3: Organizing Exam Result Viewing Sessions

- **Participating Actors:** Teaching Assistant (TA), Student, System
- **Entry Conditions:**
    - TA is logged in and authenticated.
    - Exam results are available.
- **Exit Conditions:**
    - Viewing sessions are successfully created and published for students.
- **Main Flow Events:**
    - TA navigates to the "Exam Result Viewing" module.
    - TA specifies time slots, the number of students per slot, and the location.
    - The system verifies slot availability.
    - TA publishes the schedule, and the system notifies students.
    - Students sign up for slots on a first-come, first-served basis.
    - TA can monitor and edit the schedule as needed.
- **Alternative Flows:**
    1) **Time Conflict:**
    - The system notifies the TA of schedule conflicts (if API integration with AIRS exists).
    2) **Low Sign-ups:**
    - TA adjusts session timings or extends the slots.

Scenario 4: Building a Portfolio

- **Participating Actors:** Teaching Assistant, System
- **Entry Conditions:**
    - TA is logged in and authenticated.
    - Exam and grading files are available.

- **Exit Conditions:**
    - The portfolio is successfully created and exported as a PDF.
- **Main Flow Events:**
    - TA accesses the "Portfolio Builder" module.
    - TA uploads required files (exam scans, attendance sheets, grading files).
    - TA inputs letter grade ranges or calculates them from the grading file.
    - TA generates the course summary form.
    - The system compiles all uploaded data into a single merged PDF.
    - TA previews and exports the portfolio file.
- **Alternative Flows:**
    1) **Missing Files:**
    - The system notifies the TA of missing required uploads.

Scenario 5: Generating Analytics Reports

- **Participating Actors**: Instructor, Teaching Assistant, System
- **Entry Conditions:**
    - The user is logged in and authenticated.
    - Sufficient exam or attendance data exists.
- **Exit Conditions:**
    - The analytics report is generated and displayed.
- **Main Flow Events:**
    - The user navigates to the "Analysis Tools" module.
    - User selects the desired report type:
    - Success rate per question
    - Grades vs. attendance correlation
    - ABET/MÜDEK compliance reports
    - The system retrieves the relevant data and generates visualized reports.
    - The user views and exports the generated reports in a preferred format.
- **Alternative Flows:**
    1) **Insufficient Data:**

  - The system notifies the user that insufficient data exists for the selected report.

Scenario 6: User Authentication

- **Participating Actors:** User (Instructor, TA, Student), System
- **Entry Conditions:**
  - The user opens the application and attempts to log in.
- **Exit Conditions:**
  - Users are successfully authenticated and granted access based on their role.
- **Main Flow Events:**
  - The user opens the login page.
  - Users input their credentials (username and password).
  - The system verifies the credentials against the database.
  - The system generates a JWT token and grants access if credentials are valid.
  - Users are redirected to the dashboard based on their role (e.g., Instructor dashboard).
- **Alternative Flows:**
  1) **Invalid Credentials:**
  - The system notifies the user of invalid credentials.
  - The user retries or uses the "Forgot Password" option.
  - Two-Factor Authentication (2FA):
  - The system sends a verification code to the user (email/SMS).
  - The user inputs the verification code to complete the login.

Scenario 7: User Role-Based Access Control

- **Participating Actors:** User (Instructor, TA, Student), System
- **Entry Conditions:**
  - The user is logged in and authenticated.
- **Exit Conditions:**
  - Users access only the functionalities allowed by their role.
- **Main Flow Events:**

    - User attempts to access a feature (e.g., Exam Builder).
    - The system checks the user's role (Instructor, TA, Student).
    - System grants or denies access to the requested feature based on permissions:
    - Instructor: Full access to question database, exam builder, and analytics tools.
    - TA: Portfolio Builder, Exam Result Viewing Organizer.
    - Student: Exam result viewing, objection submission.
- **Alternative Flows:**
    1) **Unauthorized Access:**
    - The system displays an error message indicating insufficient permissions.

Scenario 8: Password Reset

- **Participating Actors:** User, System
- **Entry Conditions**:
    - The user cannot log in and selects "Forgot Password."
- **Exit Conditions:**
    - The user successfully resets their password.
- **Main Flow Events:**
    - The user clicks the "Forgot Password" link on the login page.
    - The system prompts the user to enter their registered email.
    - The system verifies the email and sends a password reset link.
    - The user clicks the reset link and enters a new password.
    - The system validates the password and updates it in the database.
    - Users can log in using the new password.
- **Alternative Flows:**
    1) **Unregistered Email:**
    - The system notifies the user that the email has not been found.

Scenario 9: Viewing the Dashboard

- **Participating Actors:** User (Instructor, TA, Student), System
- **Entry Conditions:**

    - The user is authenticated and logged in.
- **Exit Conditions:**
    - Users successfully access their dashboards.
- **Main Flow Events:**
    - The user logs in successfully.
    - The system redirects the user to their role-specific dashboard:
    - Instructor: Overview of courses, questions, and analytics.
    - TA: Portfolio management and exam result sessions.
    - Student: Exam results and objection status.
- **Alternative Flows:**
    1) **No Data Available:**
    - The system displays an empty dashboard with action prompts (e.g., "Add Questions").

Scenario 10: Logging Out

- **Participating Actors:** User, System
- **Entry Conditions:**
    - The user is logged in and authenticated.
- **Exit Conditions:**
    - The user is logged out and redirected to the login page.
- **Main Flow Events:**
    - The user clicks the "Logout" button.
    - The system clears the user's session or invalidates the JWT token.
    - The system redirects the user to the login page.
- **Alternative Flows:**
    1) **Network Issues:**
    - If the logout request fails, the system notifies the user and retries when the network is available.

Registration & Login

Scenario 1: Login

- **Participating Actors:** Admin, Instructor, Teaching Assistant, Institution
- **Entry Conditions:**

    - The user has a valid account with appropriate credentials.
- **Exit Conditions:**
    - The user is successfully logged into the system and gains access to their respective functionalities.
- **Main Flow Events:**
    - The actor accesses the login page.
    - The system prompts for the username and password.
    - The actor enters the correct credentials.
    - The system validates the credentials.
    - The actor is logged in, and the dashboard is displayed based on their role (Admin, instructor, TA, institution, or student).
- **Alternative Flows**
    1. **Invalid Credentials:**
    - If the user enters incorrect credentials, the system displays an error message.
    - The user can retry entering the credentials.
    2. **Token Authorization Failure**
    - If a Token Authorization process fails, the system prompts the actor to re-authenticate.

Scenario 2: Logout

- **Participating Actors:** Admin, Instructor, Teaching Assistant, Institution
- **Entry Conditions:**
    - The user is logged into the system.
- **Exit Conditions:**
    - The user is logged out, and the system displays the login page.
- **Main Flow Events:**
    - The actor clicks on the Logout button.
    - The system terminates the user session.
    - The system redirects the actor to the login page.

Scenario 3: Send Register Request

- **Participating Actors:** Institution

- **Entry Conditions:**
  - The actor does not yet have an active account.
- **Exit Conditions:**
  - A registration request is sent to the Admin for review.
- **Main Flow Events:**
  - The actor selects Send Register Request on the system.
  - The system prompts the actor to fill out registration details.
  - The actor submits the registration request.
- **Alternative Flows**
  1. **Incomplete Form Submission:**
  - If obligatory fields are missing, the system displays an error message.
  - The actor is prompted to complete the form.

Scenario 4: Review Register Requests

- **Participating Actors:** Admin
- **Entry Conditions:**
  - The admin is logged in.
  - There are pending registration requests in the system.
- **Exit Conditions:**
  - The registration request is either accepted or rejected.
- **Main Flow Events:**
  - The admin accesses the Review Register Requests functionality.
  - The system displays a list of pending registration requests.
  - The admin selects a request to review details.
  - The admin decides to:
    - Accept Register Request: The system creates a user account and notifies the requester.
    - Reject Register Request: The system notifies the requester that their request was rejected.
- **Alternative Flows**
  1. **System Error During Account Creation:**
  - The system displays an error message if an error occurs while creating the account.

    - The Admin retries or contacts the requester if an error occurs with the information.

Scenario 5: Register User

- **Participating Actors:** Admin
- **Entry Conditions:**
    - A valid registration request for a user (instructor or TA) exists.
- **Exit Conditions:**
    - The user account (instructor or TA) is successfully created, and the user is notified.
- **Main Flow Events:**
    - The Admin accesses the Register User functionality.
    - The system displays user registration options, such as Instructor or Teaching Assistant registrations.
    - The admin selects the appropriate role:
        - Instructor (extends “Register User”).
        - Teaching Assistant (TA) (extends “Register User”).
    - The system prompts for required details in a form (e.g., name, email, role).
    - The admin submits the registration form.
    - The system validates the input and creates the user account.
    - The system notifies the user (instructor or TA) about successful registration.

Scenario 6: Token Authorization

- **Participating Actors:** Student
- **Entry Conditions:**
    - A student clicks the link sent via email.
- **Exit Conditions:**
    - The student is authenticated and gains access to the requested functionality.
- **Main Flow Events:**
    - The system sends the student a link (authentication token) via email.

    - The student clicks the link in the email (or enters the token into the system).
    - The system validates the token.
    - If valid, the student is authenticated and granted access.
- **Alternative Flows:**
    1. **Invalid or Expired Token:**
    - If the token is invalid or expired, the system displays an error and prompts the student to request a new token.
    2. **Link Not Clicked:**
    - If the student verifies their email within a set timeframe, the token becomes valid, and the registration remains complete.

Scenario 7: Ban an Institution:

- **Participating Actors:** Admin
- **Entry Conditions:**
    - The admin identifies an institution that needs to be banned due to misuse or other issues.
- **Exit Conditions:**
    - The institution is banned, and their account is deactivated.
- **Main Flow Events:**
    - The admin accesses the Ban Institution functionality.
    - The system displays a list of institutions.
    - The admin selects the institution to ban.
    - The system deactivates the institution's account and notifies the institution.

Scenario 8: Delete an Institution

- **Participating Actors:** Admin
- **Entry Conditions:**
    - The admin decides to permanently delete an institution's account.
- **Exit Conditions:**
    - The institution's account and associated data are permanently deleted.
- **Main Flow Events:**

    - The admin accesses the delete institution functionality.
    - The system prompts for confirmation to delete the institution's account.
    - Upon confirmation, the system deletes the institution's account and related records.
- **Alternative Flows:**
    1. **Cancellation:**
        - If the admin cancels the deletion, no changes are made.
    2. **System Error:**
        - If the deletion fails due to a system error, the system notifies the admin and logs the error.

Question Database and Exam Builder

Scenario 1: Create a Template

- **Participating Actors:** Instructors
- **Entry Conditions:**
    - The Instructors is logged into the system
- **Exit Conditions:**
    - A new exam template is created successfully and is ready for further editing.
- **Main Flow Events:**
    - The Instructor accesses the Create Template feature.
    - The system provides options to create a blank template or Import LaTeX content.
    - The instructor selects Import LaTeX to upload an existing LaTeX file.
    - The system validates the uploaded LaTeX file and integrates it into the template.
    - The instructor proceeds to save the created template.
    - A success message is displayed, and the new template becomes available for editing.
- **Alternative Flows:**
    1. **Invalid LaTeX File:**

    - The system displays an error message if the uploaded LaTeX file is invalid.
    - The Instructor is prompted to retry or upload the correct file.

Scenario 2: Edit an Existing Exam

- **Participating Actors: I**nstructor
- **Entry Conditions:**
    - The Instructor is logged in and has access to an existing exam.
- **Exit Conditions:**
    - The edited exam is saved successfully.
- **Main Flow Events:**
    - The instructor selects Edit Existing Exams from the Exam Builder interface.
    - The system displays a list of existing exams for the Instructor.
    - The instructor selects an exam to edit.
    - The instructor uses the Edit On LaTeX Editor feature to modify the exam content.
    - After editing, the instructor saves the changes.
    - A confirmation message is displayed, and the system reflects the changes.

Scenario 3: Add a New Question

- **Participating Actors:** Instructor
- **Entry Conditions:**
    - The instructor is logged into the system.
    - An exam template is open for editing.
- **Exit Conditions:**
    - A question is successfully added to the exam template.
- **Main Flow Events:**
    - The instructor accesses the Add Question feature.
    - The system provides options for adding questions in different forms:

        - Add From Database, which displayed a list of available questions from the shared database.
        - Add Image, which allows the instructor to upload an image file.
    - The instructor selects a method to add questions among the options.
    - The instructor selects a question from the database or uploads an image.
    - The system integrates the selected question or adds the uploaded image.
    - A success message is displayed, and the exam template shows the new question.
- **Alternative Flows:**
    1. **Invalid Image Upload:**
    - If the uploaded image is in an unsupported format, the system displays an error message.
    - The instructor retries with a valid file.

Scenario 4: Exporting Exams

- **Participating Actors:** Instructor
- **Entry Conditions:**
    - The instructor has completed designing the exam template.
- **Exit Conditions:**
    - The exam is exported successfully in the pdf or LaTeX format.
- **Main Flow Events:**
    - The instructor accesses the Export feature.
    - The system provides two options for export:
        - Export As PDF
        - Export As LaTeX
    - The instructor selects one of the export options.
    - The system generates the exam in the chosen format and provides a download link.
    - The instructor downloads the exported file, and a confirmation message is displayed.
- **Alternative Flows**

1. **Export Failure:**
   - An error message is displayed if the system encounters an issue during export.
   - The instructor retries to export.

Objection Session Arranger

Scenario 1: Set Up Objection Sessions

- **Use Case Name:** Setting Up Objection Sessions
- **Participating Actors:** TA, Instructor, Admin
- **Entry Conditions:**
  - The TA is logged into the system with appropriate permissions.
  - The objection session management module is accessible.
- **Exit Conditions:**
  - Objection session slots are successfully generated and finalized.
  - A unique password for the session is provided to the TA.
- **Main Flow Events:**
  - The TA accesses the objection session management module.
  - The TA inputs scheduling filters, such as:
    - Dates and hours for sessions.
    - Slot duration.
    - Expected number of students.
  - The system calculates the total number of slots required based on the provided inputs.
  - The system generates a draft schedule in Excel-like format and displays it to the TA.
  - The TA reviews and confirms the generated schedule.
  - The system finalizes the schedule and creates a unique password for the session.
  - The password is shared with the TA for distribution to students.
- **Alternative Flows:**

  1. **Invalid Filters:**
     - The system detects incomplete or conflicting filter inputs (e.g., insufficient hours for the expected number of students).

    - The system displays an error message and prompts the actor to adjust the filters.

2. **Session Overflow:**
    - The system identifies insufficient slots for all students within the given constraints.
    - The TA is prompted to extend hours or add new dates to resolve the issue.

Scenario 2: Cancel an Objection Session

- **Use Case Name:** Canceling an Objection Session
- **Participating Actors:** TA, Instructor, Admin
  Entry Conditions:
    - The objection session has been previously created and finalized.
    - The TA, Instructor, or Admin is logged into the system with appropriate permissions.
- **Exit Conditions:**
    - The objection session is canceled, and all associated data (e.g., slots, schedules, passwords) is deleted from the system.
- **Main Flow Events:**
    - The actor (TA, Instructor, or Admin) accesses the objection session management module.
    - The actor selects the objection session they wish to cancel from the list of active sessions.
    - The system prompts the actor to confirm the cancellation.
    - Upon confirmation, the system:
        - Deletes all associated slots, schedules, and passwords.
        - Notifies any logged-in students (if applicable) that the session has been canceled.
        - Archives the cancellation log for future reference.
    - A success message is displayed to the actor.
- **Alternative Flows:**
  1. **Cancellation Denied Due to System Error:**

    - An error message is displayed if the system encounters an issue during cancellation (e.g., a session lock).
    - The actor is prompted to retry or contact support.

2. **Cancellation Prompt Ignored:**
    - If the actor does not confirm the cancellation, the session remains active, and no changes are made.

Scenario 3: Select an Available Slot

- **Use Case Name:** Selecting a Slot
- **Participating Actors:** Student, Admin
- **Entry Conditions:**
    - The objection session schedule is live and accessible.
    - The student has received a valid password for the session.
- **Exit Conditions:**
    - The student has successfully selected and reserved a slot.
- **Main Flow Events:**
    - The student enters the session password to access the platform.
    - The system validates the password and displays available slots.
    - The student selects a desired slot.
    - The system locks the selected slot to prevent others from further selecting it.
    - A confirmation message is displayed, and the selected slot is shown on the student's dashboard.
- **Alternative Flows:**

1. **Invalid Password:**
    - The student enters an incorrect password.
    - The system displays an error message and denies access.
    - The student can retry entering the password.
2. **Slot Unavailable:**
    - The selected slot is no longer available (already locked by another student).
    - The system notifies the student and prompts them to choose a different slot.

Scenario 4: Change a Previously Chosen Slot

- **Use Case Name:** Changing a Slot
- **Participating Actors:** Student, Admin
- **Entry Conditions:**
    - The student has already reserved a slot.
- **Exit Conditions:**
    - The student’s slot is updated, and the previous slot is unlocked.
- **Main Flow Events:**
    - The student accesses their dashboard and clicks the "Change Slot" option.
    - The system unlocks the previously selected slot, making it available to other students.
    - The student selects a new slot from the remaining options.
    - The system locks the new slot and updates the student’s reservation.
    - A confirmation message is displayed, and the new slot is shown on the student’s dashboard.
- **Alternative Flows:**

    1. **No Slots Available:**
        - If no slots are available, the system notifies the student.
        - The student is prompted to keep their current slot or wait for additional slots.

Scenario 5: Session Completion and Review

- **Use Case Name:** Session Completion and Review
- **Participating Actors:** TA, Instructor, Admin
- **Entry Conditions:**
    - The objection session is complete, and no further slots can be selected.
- **Exit Conditions:**
    - The TA has access to a detailed report of the session.
- **Main Flow Events:**
    - The system logs all student slot selections and changes.

    - The TA accesses the session management module to view the summary.
    - The system generates a detailed report, including:
        - Student names and selected slots.
        - Any slot changes made during the session.
    - The TA downloads the report.
- **Alternative Flows:**

  1. **Incomplete Log:**
    - The system detects incomplete or missing data for sure students.
    - The TA is notified and provided with options to resolve the issue.

Portfolio Builder

Scenario 1: Upload Documents for a Course Portfolio

- **Use Case Name:** Uploading Documents for a Course Portfolio
- **Participating Actors:** TA, Instructor, Admin
- **Entry Conditions:**
    - The TA or Instructor is logged into the system with appropriate permissions.
    - The course is registered in the system for the current term.
- **Exit Conditions:**
    - Documents are successfully uploaded and associated with the selected course.
- **Main Flow Events:**
    - The TA or Instructor accesses the portfolio builder module.
    - The user selects the relevant course.
    - The system displays an upload interface, prompting users to add required documents.
    - The user uploads documents individually or in batches.
    - The system validates the uploaded files for format and size.
    - Upon successful validation, the files are stored in the system and linked to the selected course.

    - The system confirms the upload and displays the updated document list.
- **Alternative Flows:**
  1. **File Format or Size Error:**
     - The system detects an unsupported file format or exceeds the size limit.
     - An error message prompts the user to retry using a valid file.

Scenario 2: Review Uploaded Documents

- **Use Case Name:** Reviewing Uploaded Documents
- **Participating Actors:** TA, Instructor, Admin
- **Entry Conditions:**
    - Documents have been previously uploaded for the course.
- **Exit Conditions:**
    - Documents are reviewed and confirmed for accuracy.
- **Main Flow Events:**
    - The TA or Instructor accesses the course portfolio.
    - The system displays a list of all uploaded documents, including file names and upload dates.
    - The user reviews the list and verifies completeness and accuracy.
    - If necessary, the user flags files for updates or deletion.
- **Alternative Flows:**
  1. **Missing Document Warning:**
     - The system detects that required documents are missing.
     - A notification prompts the user to complete the uploads.

Scenario 3: Delete a Document

- **Use Case Name:** Deleting a Document
- **Participating Actors:** TA, Instructor, Admin
- **Entry Conditions:**
    - At least one document has been uploaded for the course.
- **Exit Conditions:**
    - The selected document is deleted from the course portfolio.

- **Main Flow Events:**
    - The user accesses the course portfolio and identifies the document to delete.
    - The user selects the document and clicks the "Delete" button.
    - The system prompts the user for confirmation.
    - Upon confirmation, the document is deleted.
    - A success message is displayed, and the updated document list is shown.

Scenario 4: Generate the End-of-Term Report

- **Use Case Name:** Generating the End-of-Term Report
- **Participating Actors:** TA, Instructor, Admin
- **Entry Conditions:**
    - All required documents for the course have been uploaded.
- **Exit Conditions:**
    - A comprehensive end-of-term report is generated and available for download.
- **Main Flow Events:**
    - The user selects the course and clicks the "Generate Report" button.
    - The system compiles all uploaded documents into a single PDF file.
    - The system arranges the documents in a predefined order.
    - The generated report is displayed for review.
    - The user downloads the report.
- **Alternative Flows:**

    1. **Missing Documents Detected:**
        - The system notifies the user before generating the report if the required documents still need to be included.

2.4.2. Use Case Model

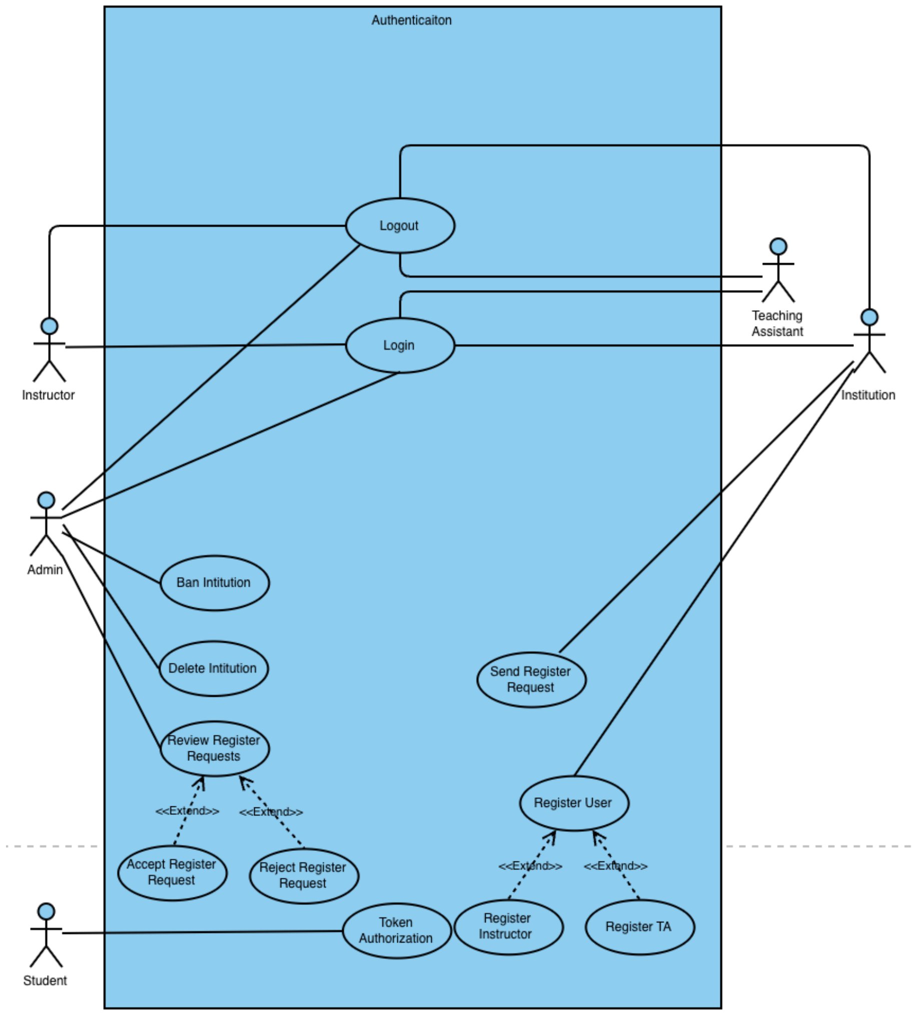


Figure 1: Authentication Use-Case Diagram

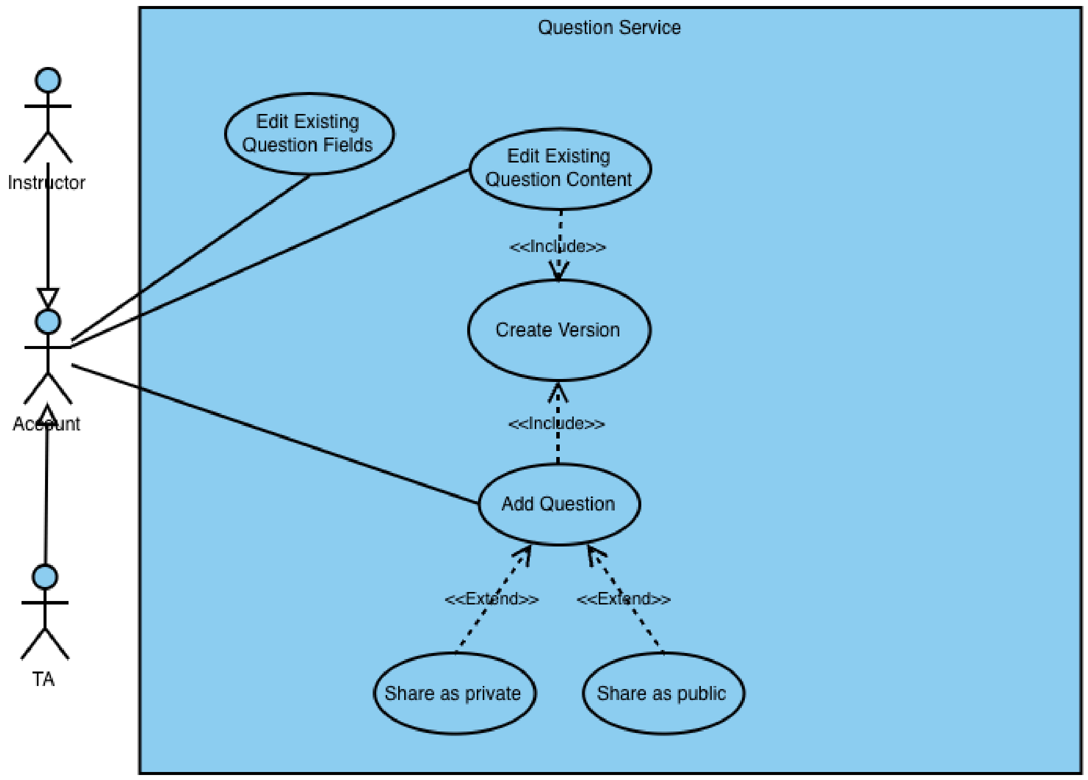


Figure 2: Question Service Use-Case Diagram

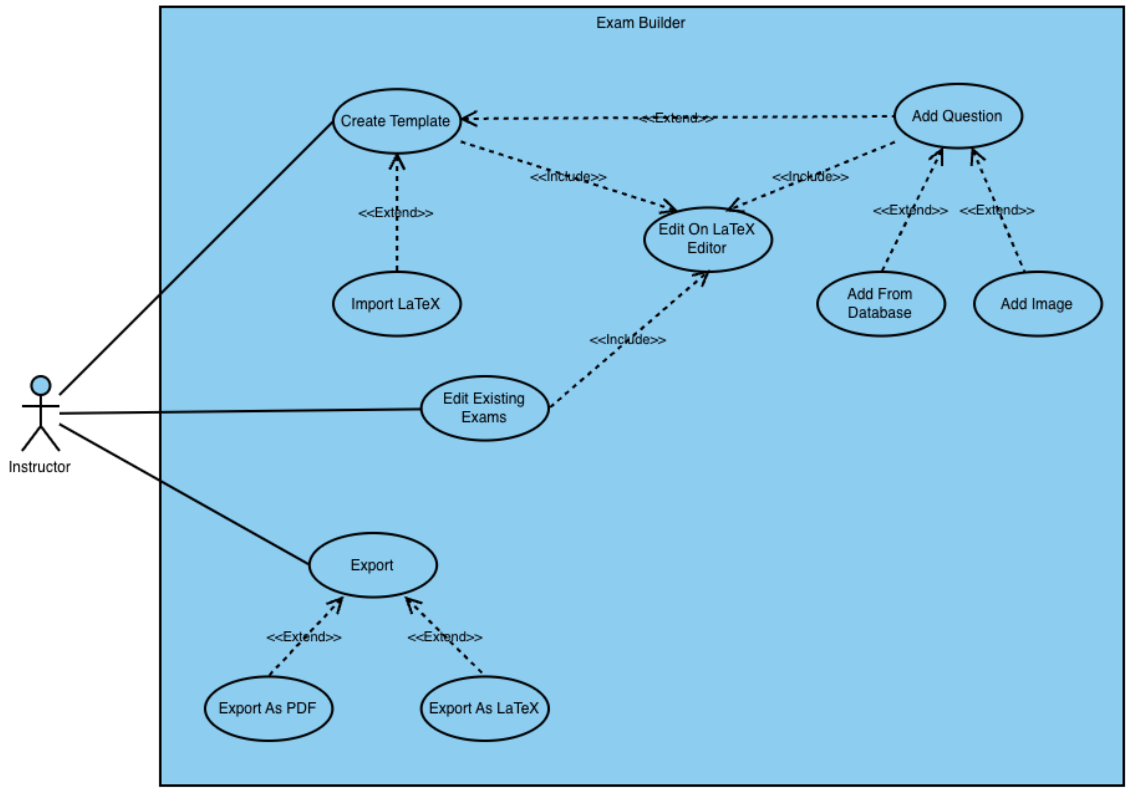


Figure 3: Exam Builder Use-Case Diagram

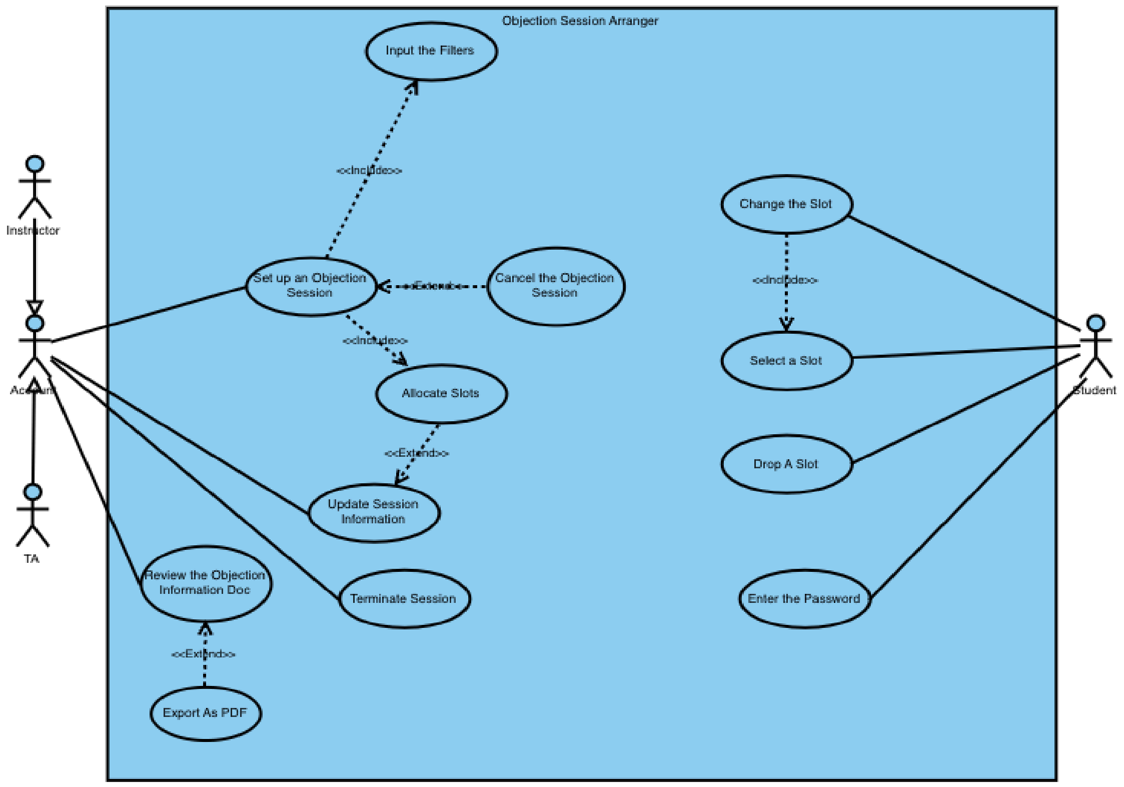


Figure 4: Objection Session Arranger Use-Case Diagram

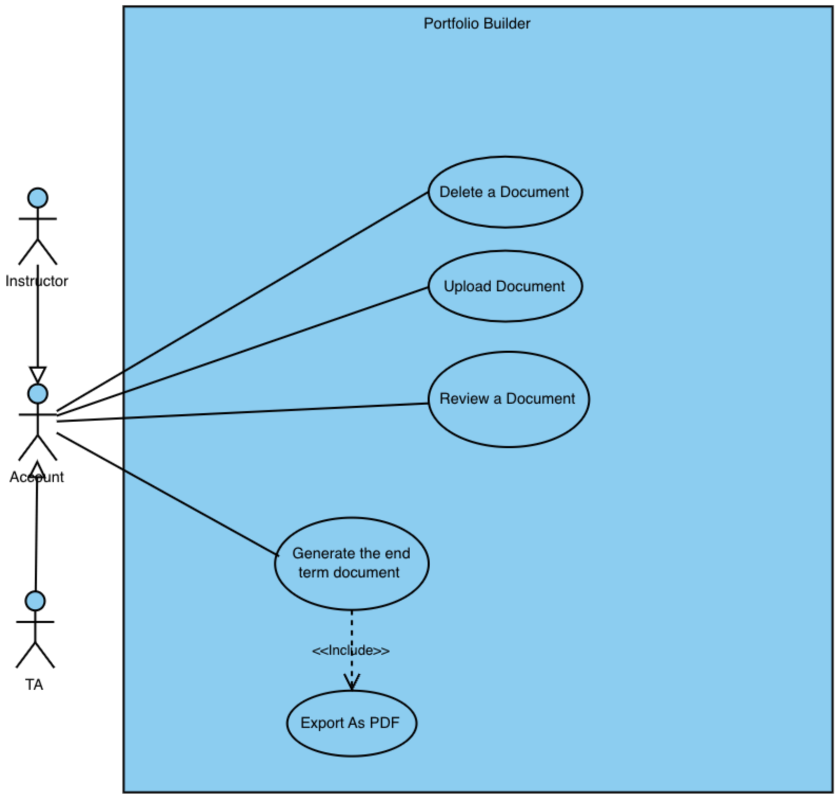


Figure 5: Portfolio Builder Use-Case Diagram

2.4.3. Object and Class Model

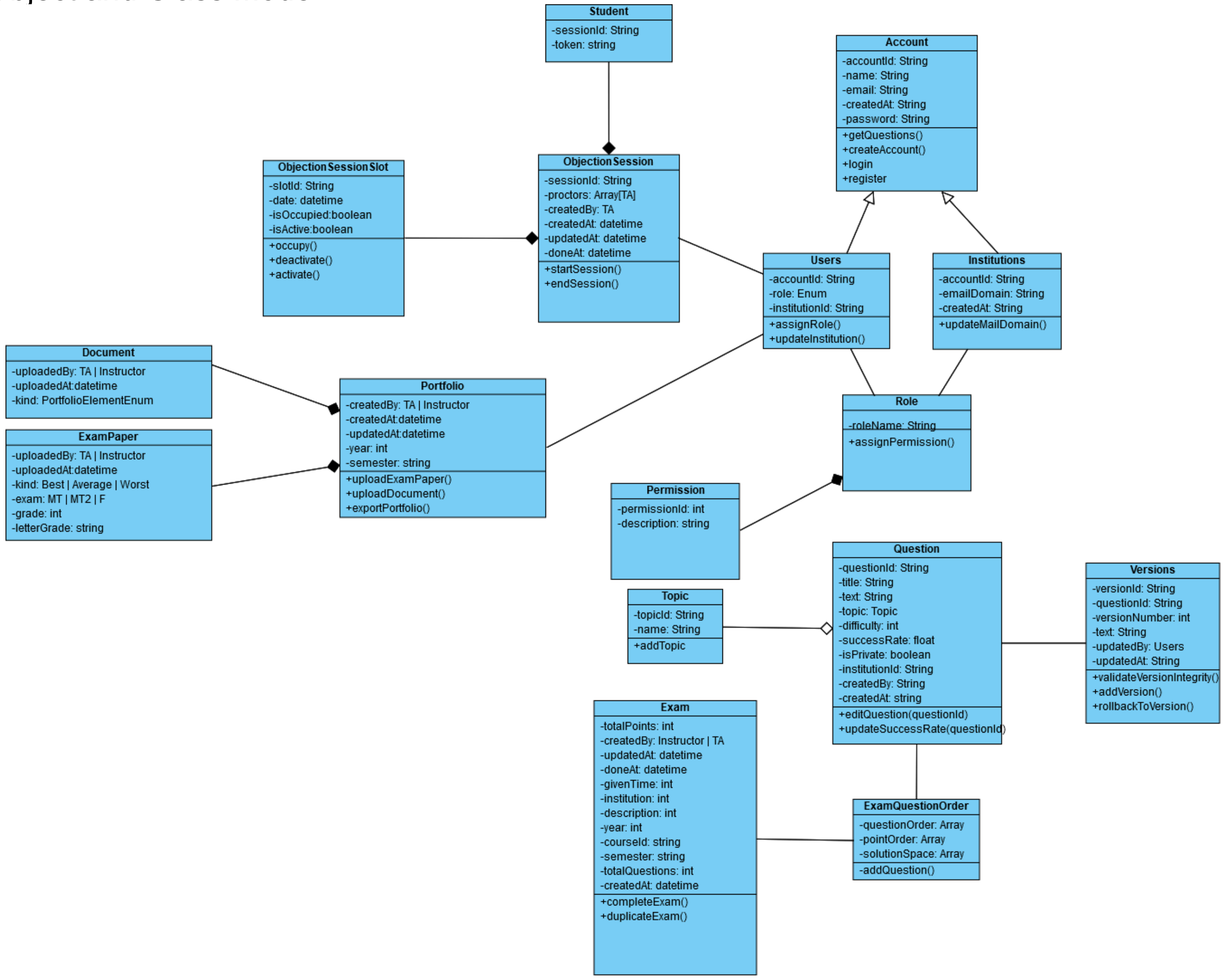


Figure 6: Class Diagram for Exam Builder and Session Organizer

2.4.4. Dynamic Models

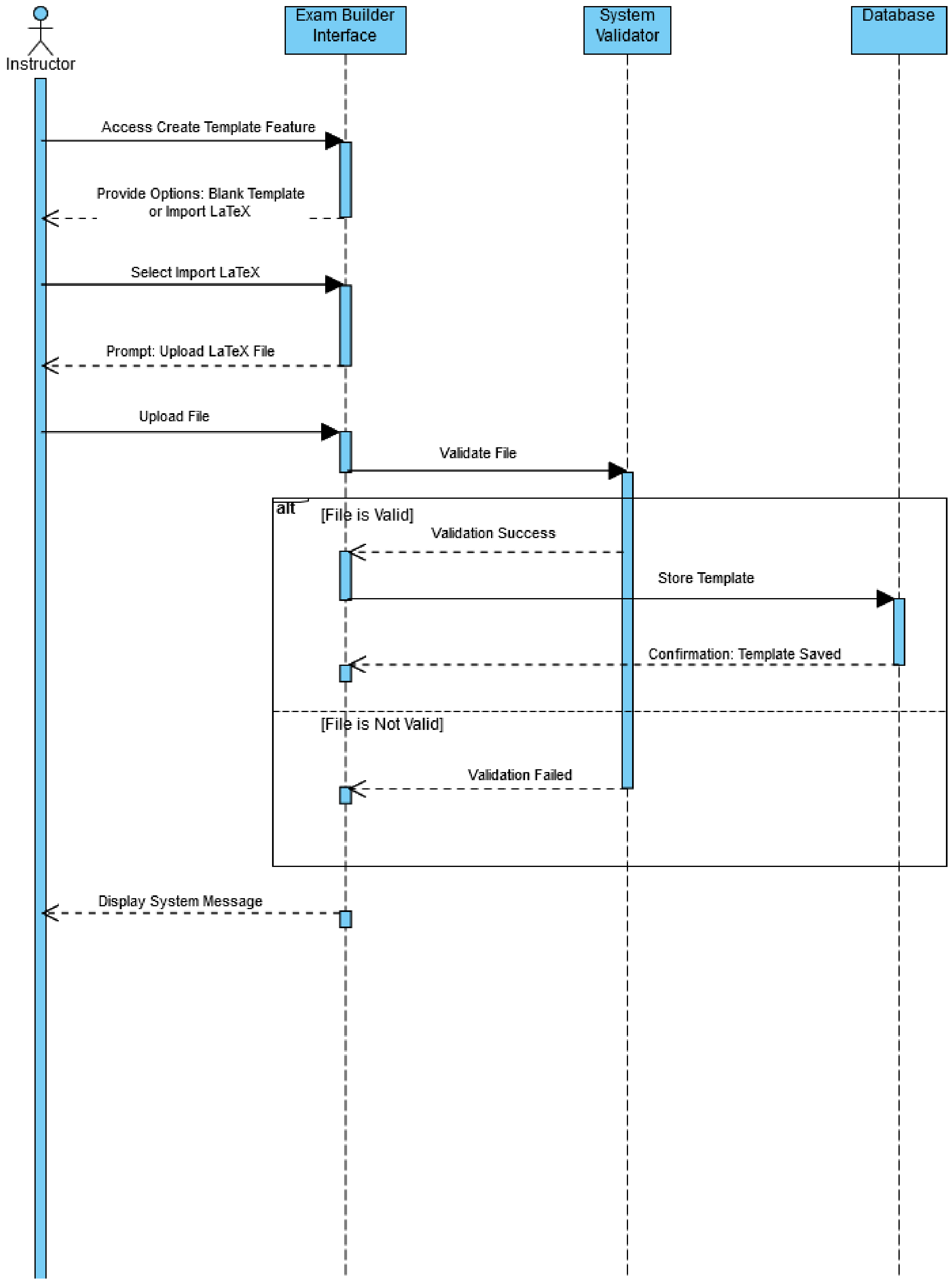


Figure 7: Exam Builder Sequence Diagram

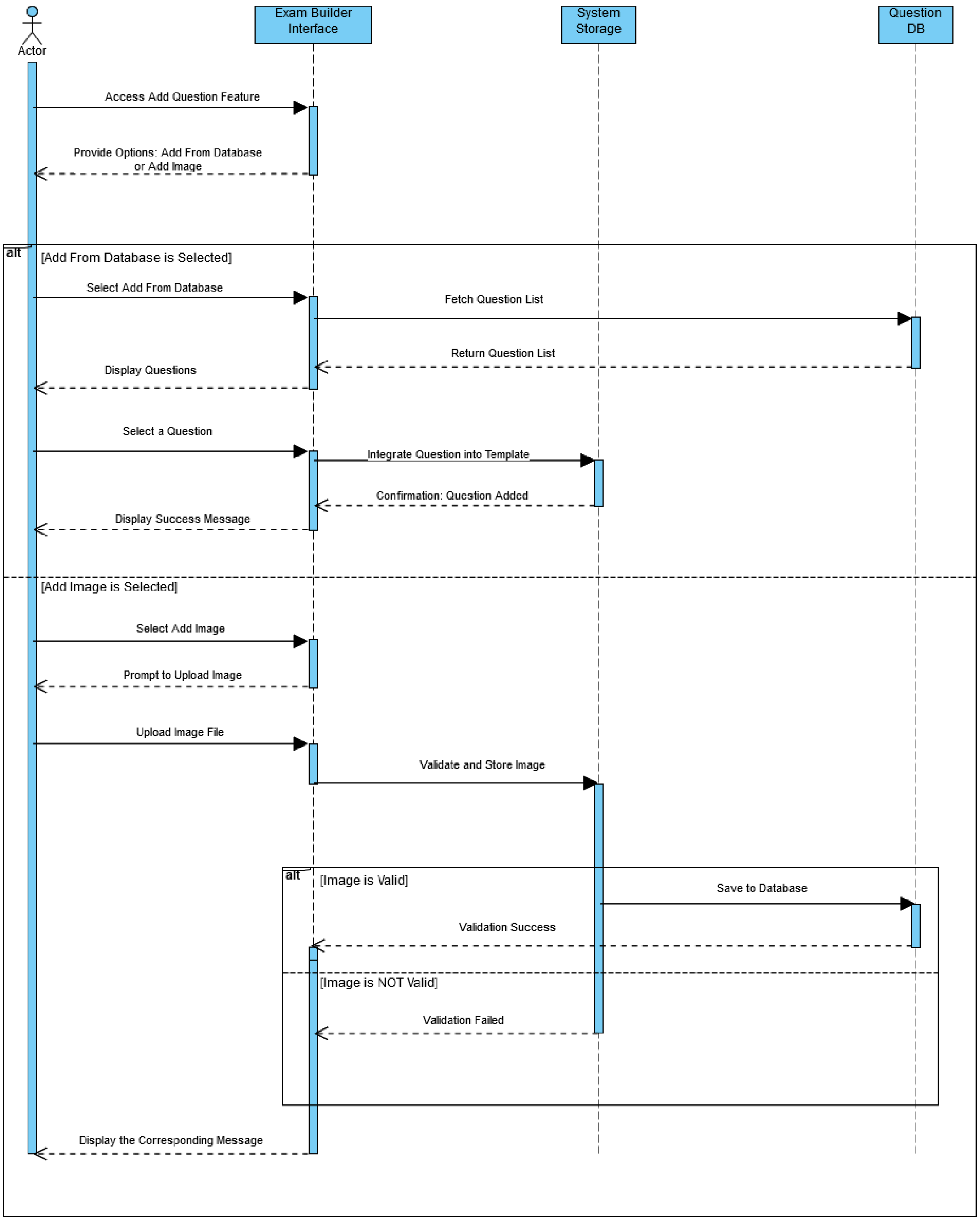


Figure 8: Adding Question Sequence Diagram

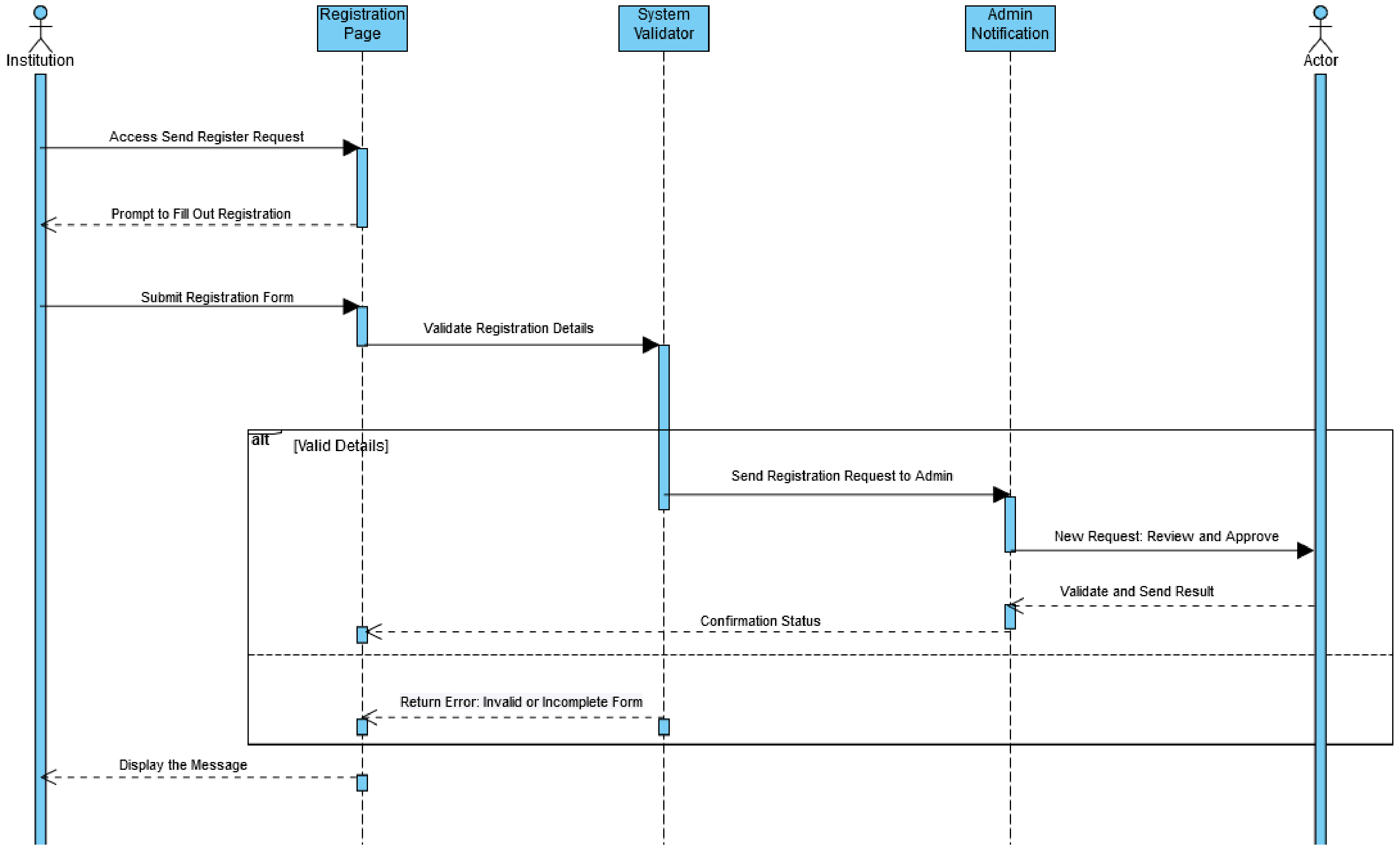

Institution
Registration Page
System Validator
Admin Notification
Actor
Access Send Register Request
Prompt to Fill Out Registration
Submit Registration Form
Validate Registration Details
alt
[Valid Details]
Send Registration Request to Admin
New Request: Review and Approve
Validate and Send Result
Confirmation Status
Return Error: Invalid or Incomplete Form
Display the Message

Figure 9: Exam Builder Sequence Diagram

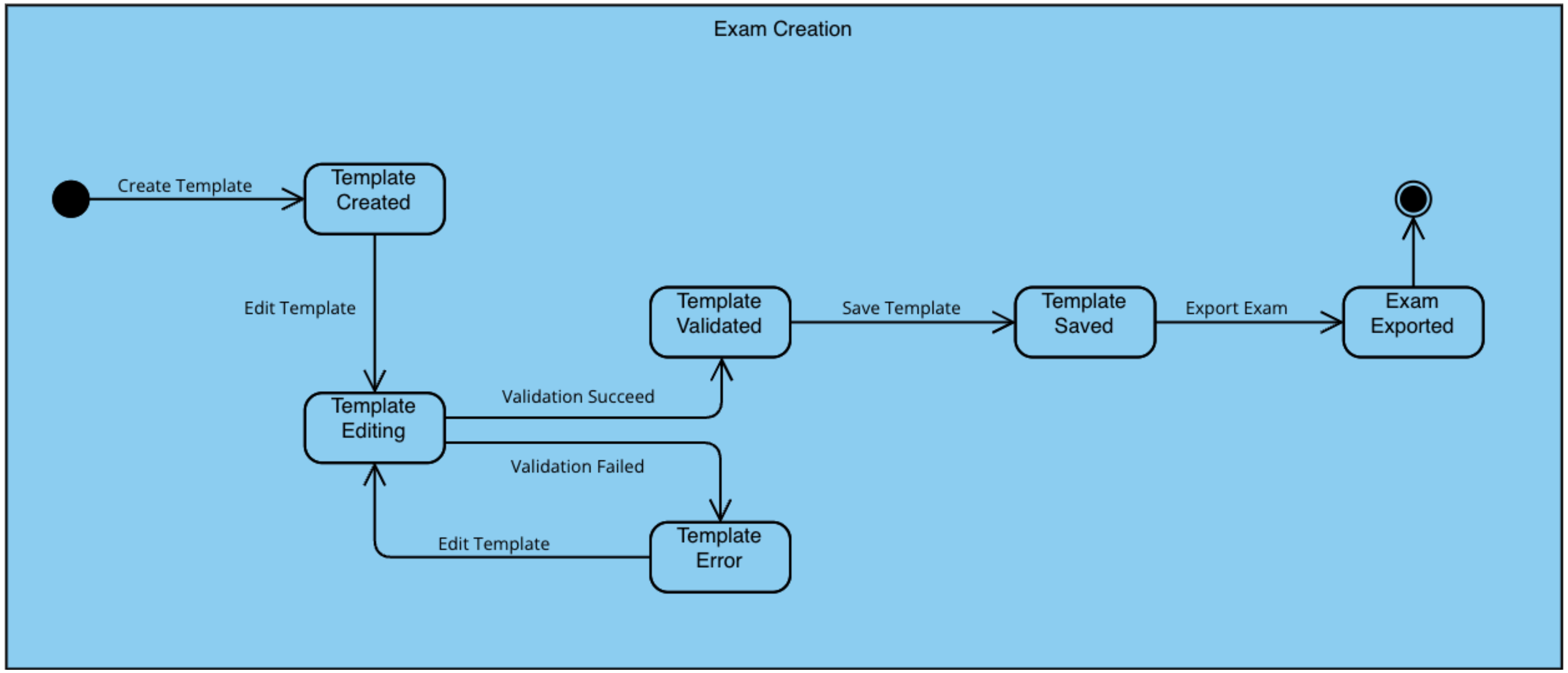


Figure 10: Exam Creation State Diagram

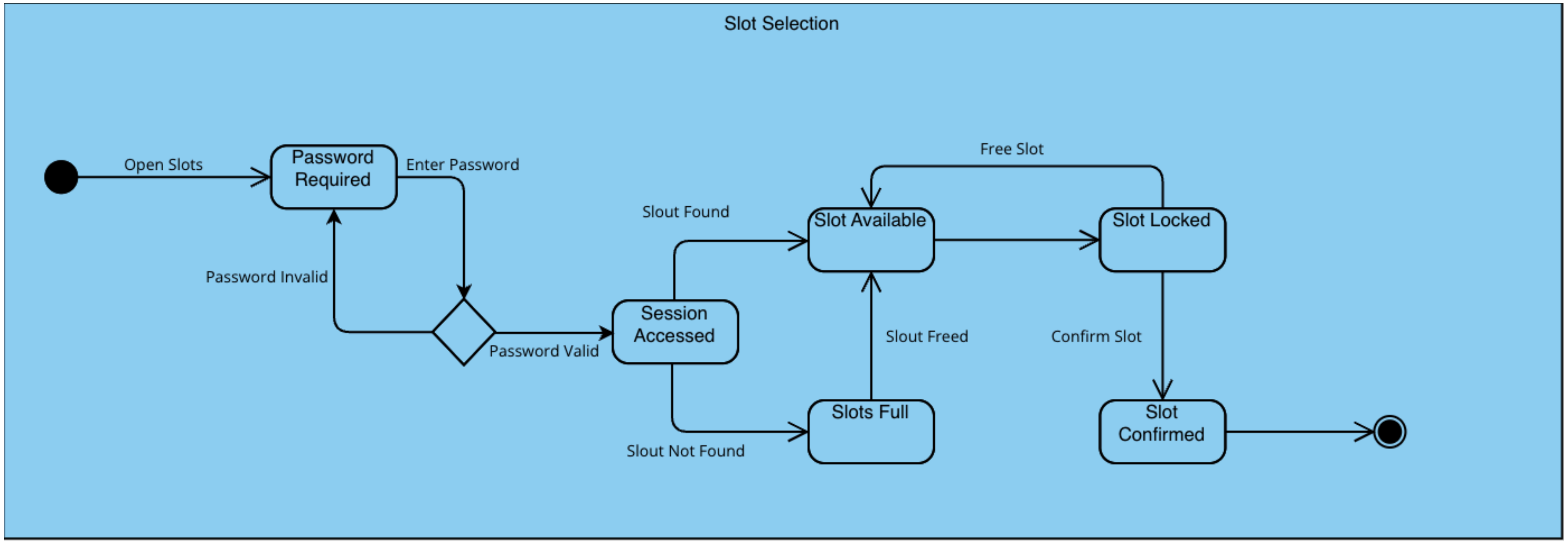


Figure 11: Slot Selection State Diagram

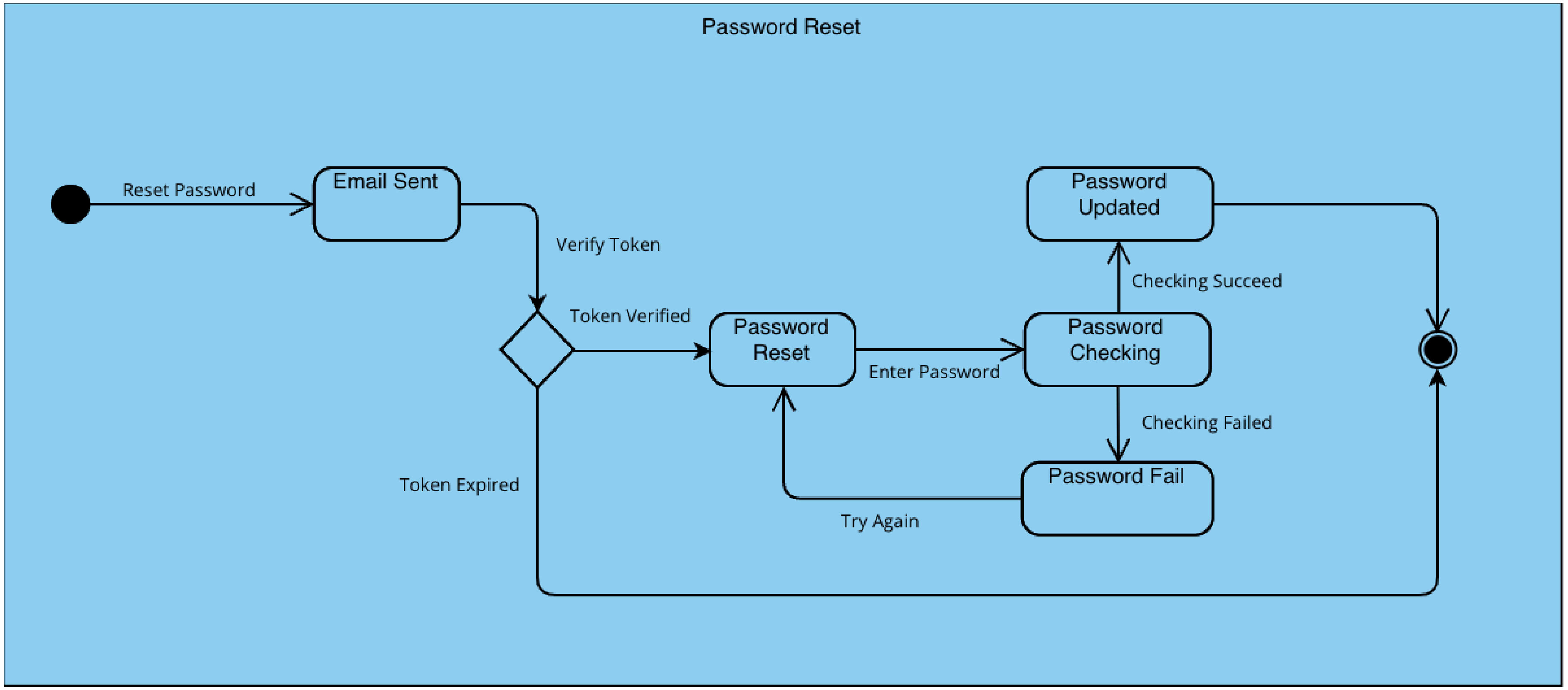


Figure 12: Password Reset State Diagram

2.4.5. User Interface - Navigational Paths and Screen Mock-ups

2.4.5.1. Login

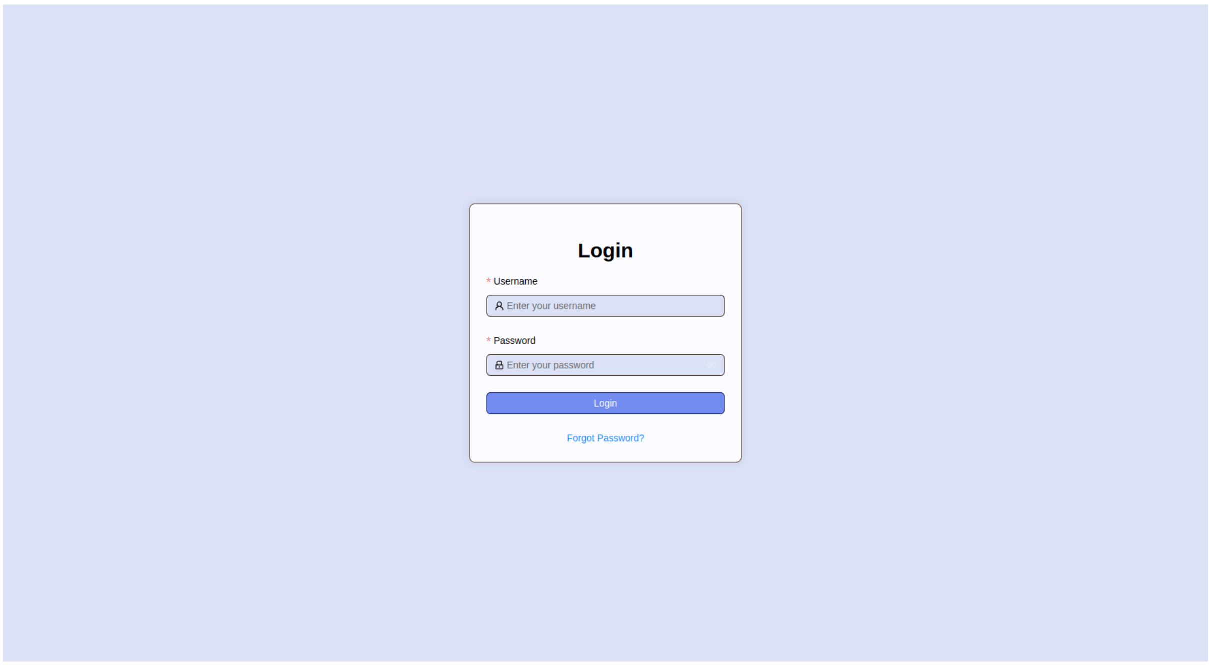


Figure 13: Login Screen

Users (instructors, teaching assistants, institutions) can log in to the system entering their credentials, or change their passwords.

2.4.5.2. Register

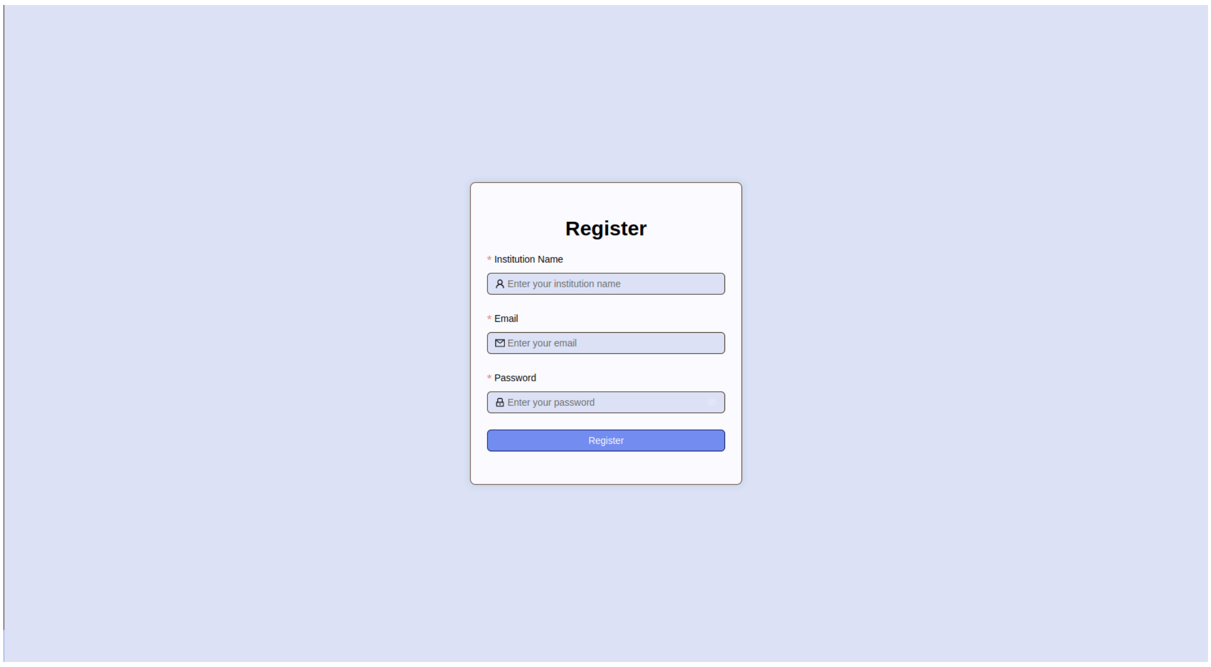
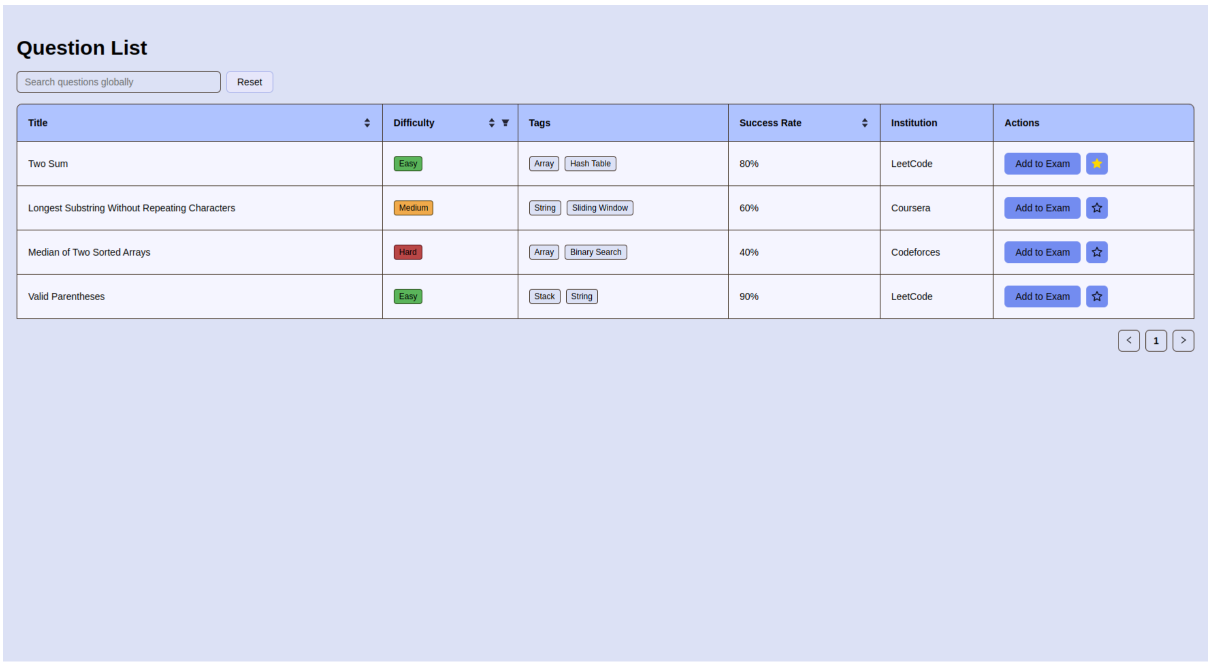


Figure 14: Register Screen

Institutions send registration requests using this page by providing the institution name, username, password, and university's email domain.

2.4.5.3. Question List



Question List

Search questions globally | Reset

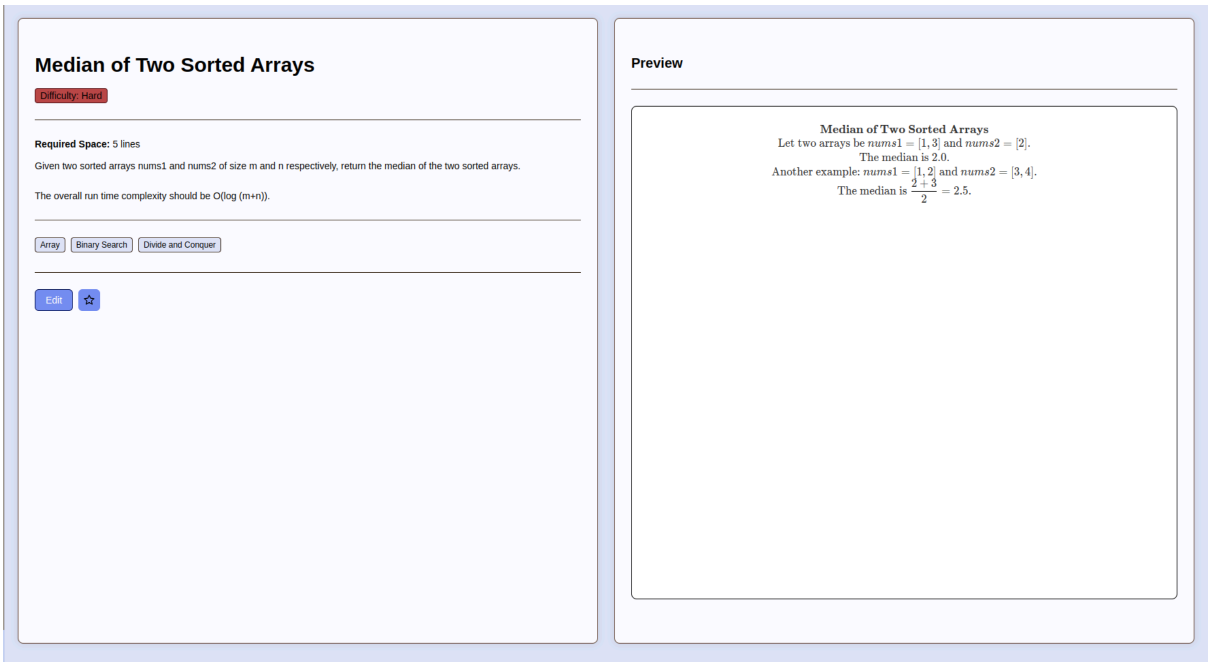
Title	Difficulty	Tags	Success Rate	Institution	Actions
Two Sum	Easy	Array, Hash Table	80%	LeetCode	Add to Exam ★
Longest Substring Without Repeating Characters	Medium	String, Sliding Window	60%	Coursera	Add to Exam ☆
Median of Two Sorted Arrays	Hard	Array, Binary Search	40%	Codeforces	Add to Exam ☆
Valid Parentheses	Easy	Stack, String	90%	LeetCode	Add to Exam ☆

< 1 >

Figure 15: Question List Page

Instructors and TAs can see the list of questions on this page. The title, difficulty level, tags, success rate, and the institution information of the questions are shown in a grid. Users can search questions based on their titles and sort them based on the columns of the given grid.

2.4.5.4. Question Detail - Edit Question


Median of Two Sorted Arrays

Difficulty: Hard

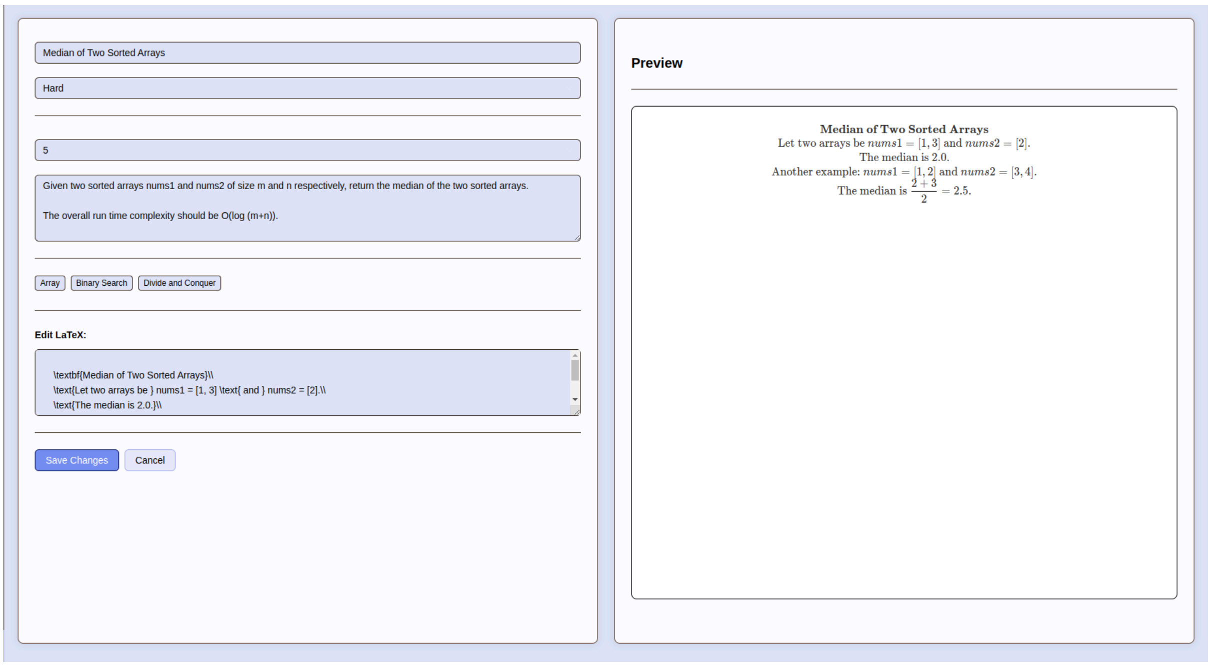
Required Space: 5 lines

Given two sorted arrays nums1 and nums2 of size m and n respectively, return the median of the two sorted arrays.

The overall run time complexity should be O(log (m+n)).

Median of Two Sorted Arrays
Let two arrays be $nums1 = [1, 3]$ and $nums2 = [2]$.
The median is 2.0.
Another example: $nums1 = [1, 2]$ and $nums2 = [3, 4]$.
The median is $\frac{2+3}{2} = 2.5$.

Figure 16: Question Details Page


Median of Two Sorted Arrays

Hard

5

Given two sorted arrays nums1 and nums2 of size m and n respectively, return the median of the two sorted arrays.

The overall run time complexity should be O(log (m+n)).

Edit LaTeX:

```
\textbf{Median of Two Sorted Arrays}\\
\text{Let two arrays be } nums1 = [1, 3] \text{ and } nums2 = [2].\\
\text{The median is 2.0.}\\
```

Median of Two Sorted Arrays
Let two arrays be $nums1 = [1, 3]$ and $nums2 = [2]$.
The median is 2.0.
Another example: $nums1 = [1, 2]$ and $nums2 = [3, 4]$.
The median is $\frac{2+3}{2} = 2.5$.

Figure 17: Question Details Page (2)

On this page, users can edit the content of the question. Changes can be tracked in the LaTeX preview screen. Tags are allocated on this page also.

2.4.5.5. Add Question

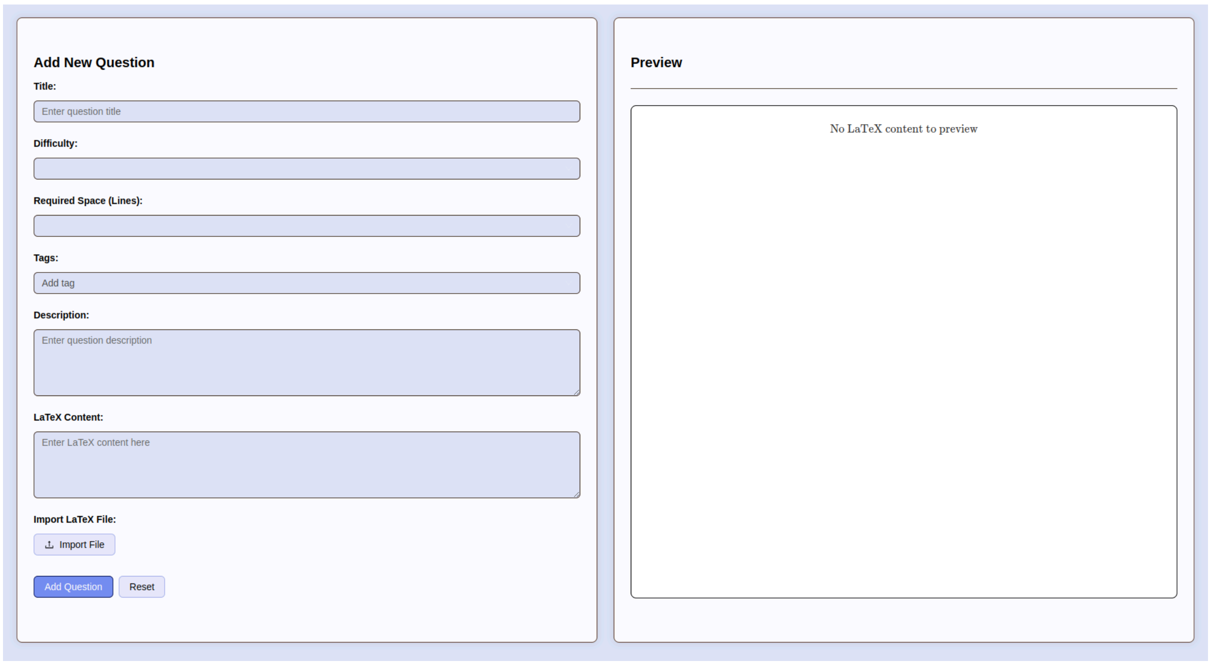
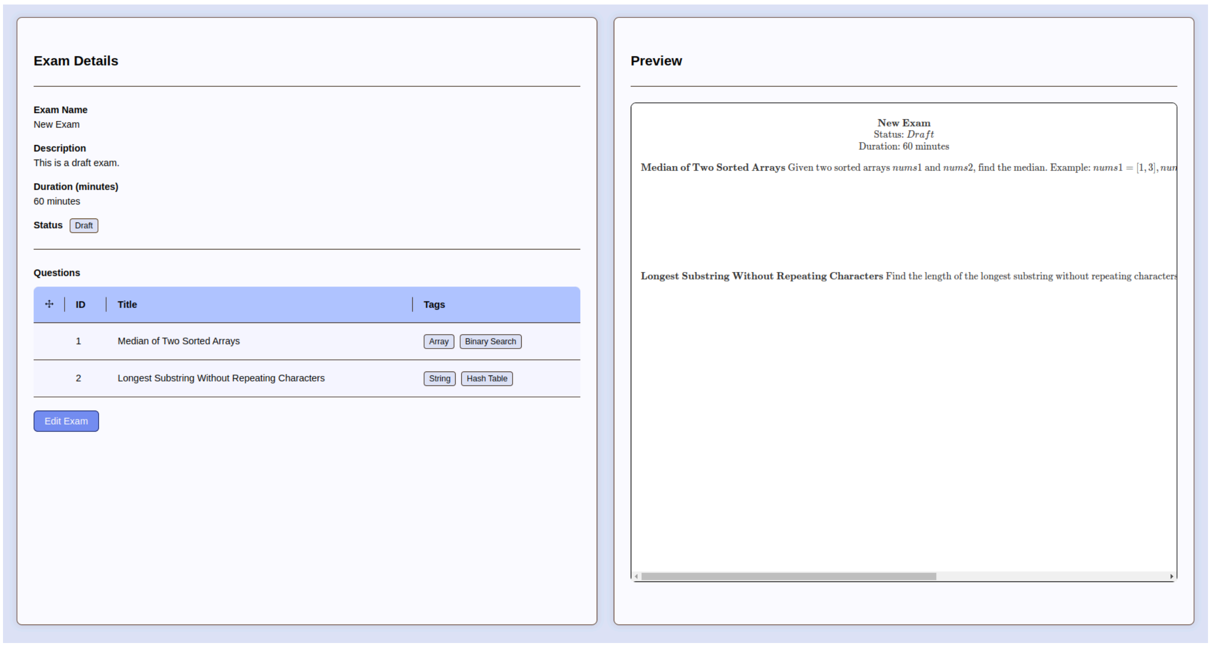


Figure 18: Add Question Page

On this page, users can create a question from scratch. When a question is created, it needs a title, difficulty level, initial tags, description, and the LaTeX content. A question can also have inserted images to provide images if required.

2.4.5.6. Exam Detail


Exam Details

Exam Name
New Exam

Description
This is a draft exam.

Duration (minutes)
60 minutes

Status Draft

Questions

ID	Title	Tags
1	Median of Two Sorted Arrays	Array, Binary Search
2	Longest Substring Without Repeating Characters	String, Hash Table

Edit Exam

Preview

New Exam
Status: $Draft$
Duration: 60 minutes

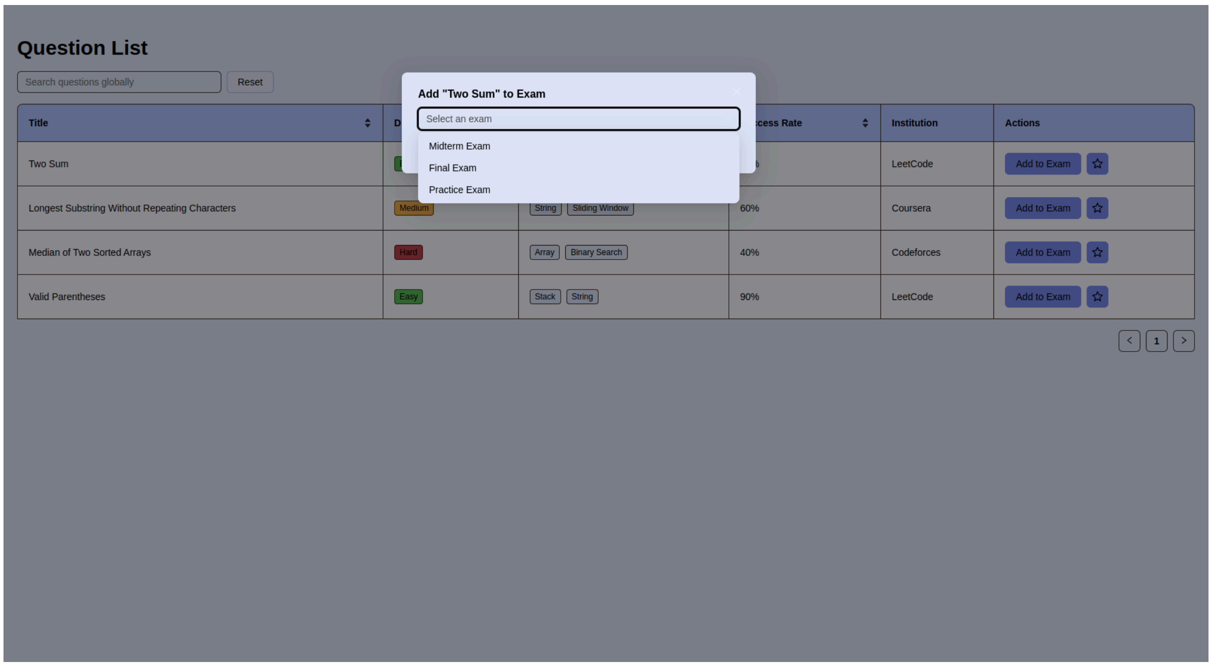
Median of Two Sorted Arrays Given two sorted arrays $nums1$ and $nums2$, find the median. Example: $nums1 = [1, 3], nun$

Longest Substring Without Repeating Characters Find the length of the longest substring without repeating characters

Figure 19: Exam Details Page

This page shows the current status and the content of an exam. Users can see the preview of the exam and which questions are included. Also, if exam creation still needs to be done, it is marked as a draft.

2.4.5.7. Adding Questions to An Exam


Question List

Search questions globally | Reset

Add "Two Sum" to Exam

Select an exam
- Midterm Exam
- Final Exam
- Practice Exam

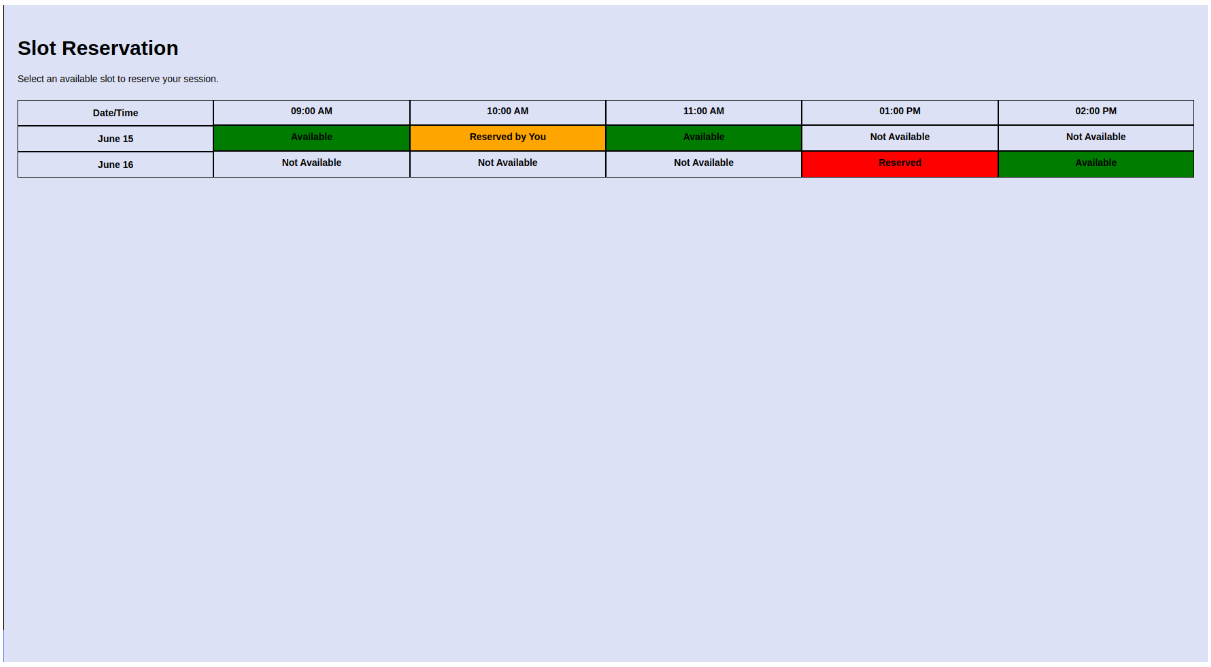
Title	D		ccess Rate	Institution	Actions
Two Sum			%	LeetCode	Add to Exam ☆
Longest Substring Without Repeating Characters	Medium	String, Sliding Window	60%	Coursera	Add to Exam ☆
Median of Two Sorted Arrays	Hard	Array, Binary Search	40%	Codeforces	Add to Exam ☆
Valid Parentheses	Easy	Stack, String	90%	LeetCode	Add to Exam ☆

< 1 >

Figure 20: Adding Question to An Exam Modal

Users can add questions to an exam while searching for questions on the question list page. When a user decides to add a question to an exam, he selects the question and then chooses the target exam.

2.4.5.8. Slot Reservation



Slot Reservation

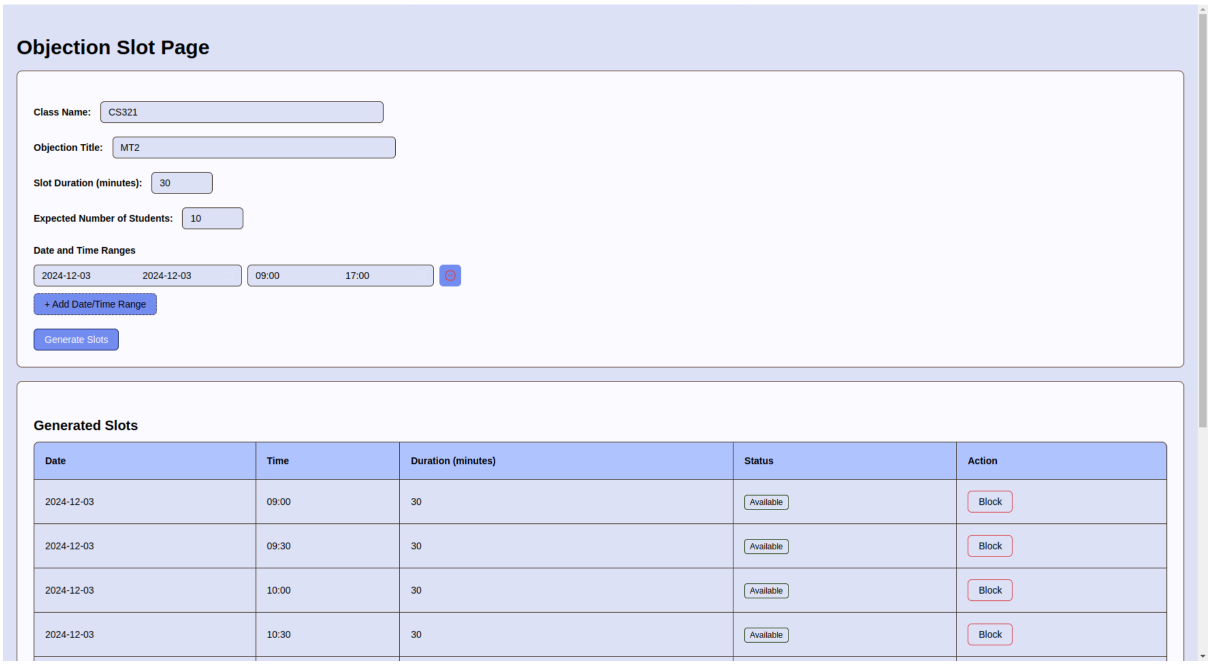
Select an available slot to reserve your session.

Date/Time	09:00 AM	10:00 AM	11:00 AM	01:00 PM	02:00 PM
June 15	Available	Reserved by You	Available	Not Available	Not Available
June 16	Not Available	Not Available	Not Available	Reserved	Available

Figure 21: Slot Reservation Page

On this page, students can see the chart for the objection session. The available or occupied slots are shown with labels and different colors. Students can occupy a slot, drop a slot, or change it via this page.

2.4.5.9. Slot Creation



Objection Slot Page

Class Name: CS321

Objection Title: MT2

Slot Duration (minutes): 30

Expected Number of Students: 10

Date and Time Ranges

2024-12-03 2024-12-03 09:00 17:00

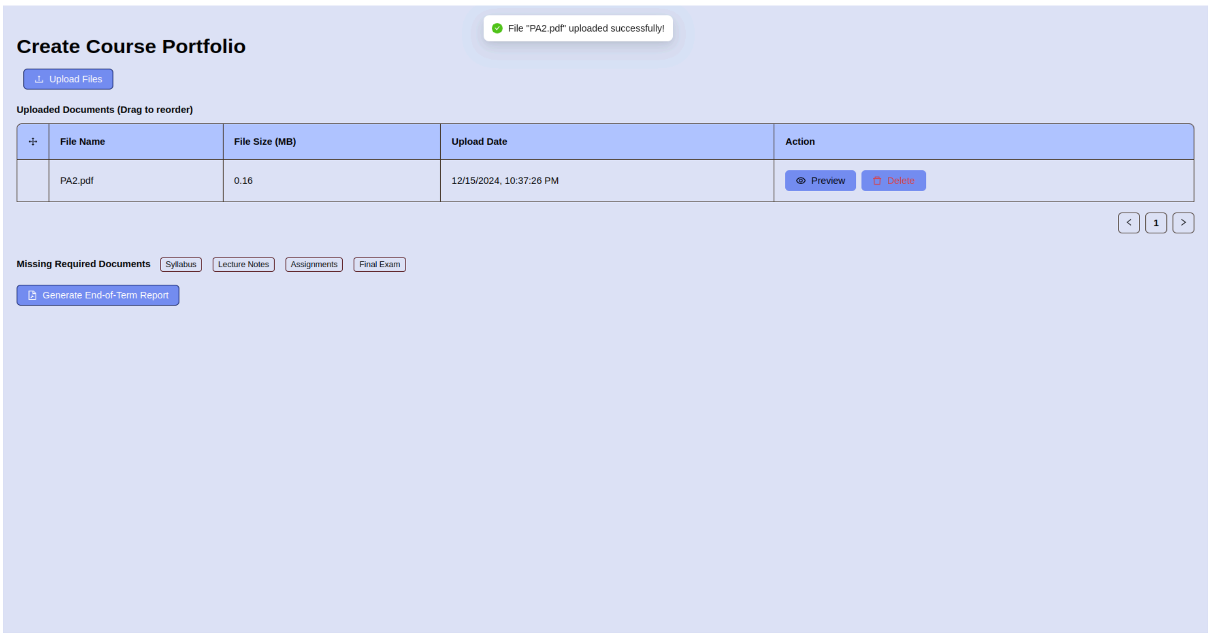
Generated Slots

Date	Time	Duration (minutes)	Status	Action
2024-12-03	09:00	30	Available	Block
2024-12-03	09:30	30	Available	Block
2024-12-03	10:00	30	Available	Block
2024-12-03	10:30	30	Available	Block

Figure 22: Slot Creation Page

Users can create slots for a designated objection session by providing the available slot duration, expected number of students, and objection period. Before creating, a preview of the slots is also shown to the user.

2.4.5.10. Create Portfolio



Create Course Portfolio

File "PA2.pdf" uploaded successfully!

Uploaded Documents (Drag to reorder)

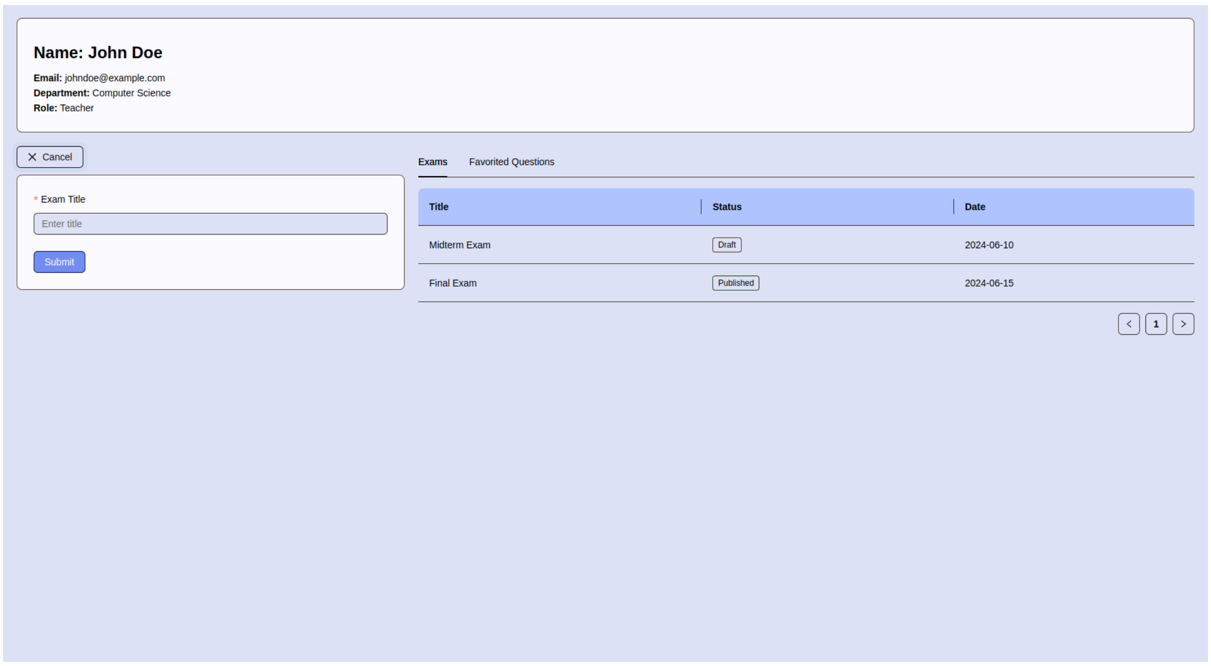
	File Name	File Size (MB)	Upload Date	Action
	PA2.pdf	0.16	12/15/2024, 10:37:26 PM	Preview Delete

Missing Required Documents: Syllabus, Lecture Notes, Assignments, Final Exam

Figure 23: Create Course Portfolio Page

On this page, users can create the course portfolio. The user adds necessary documents, which are shown on the page. After uploading the documents, the user may reorder them and export the report as a PDF. Some parts of the documents can also be gathered from the system.

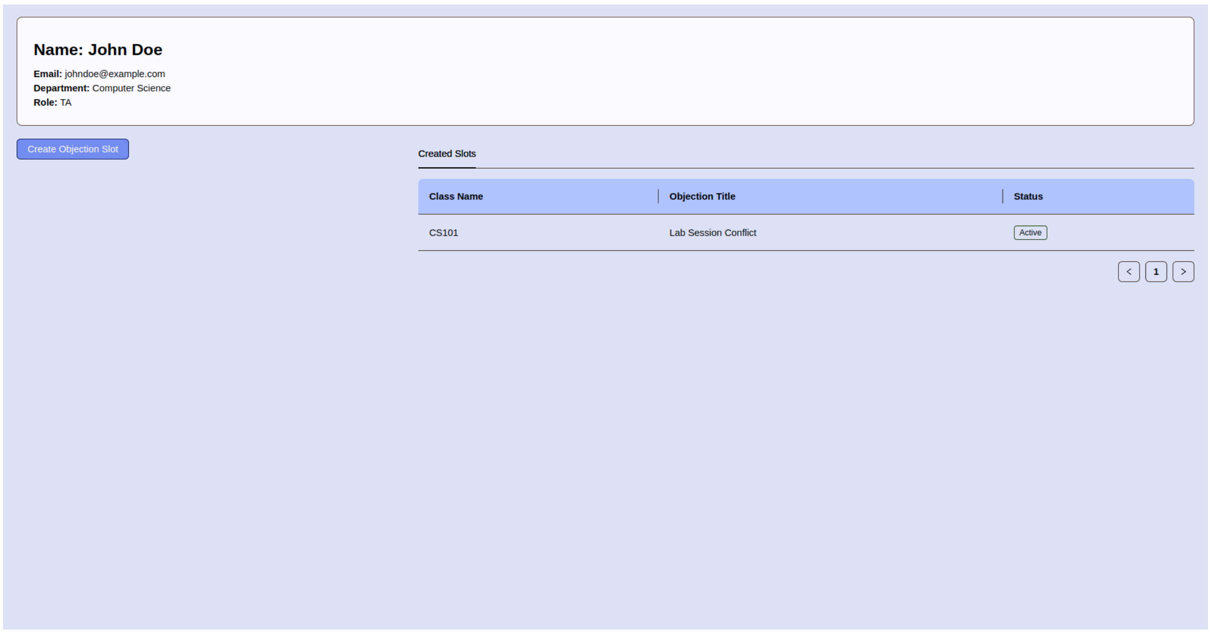
2.4.5.11. Profile Page



Name: John Doe
Email: johndoe@example.com
Department: Computer Science
Role: Teacher

Title	Status	Date
Midterm Exam	Draft	2024-06-10
Final Exam	Published	2024-06-15

Figure 24: Instructor Profile Page



Name: John Doe
Email: johndoe@example.com
Department: Computer Science
Role: TA

Class Name	Objection Title	Status
CS101	Lab Session Conflict	Active

Figure 25: TA Profile Page

The Teacher Profile Page overviews a teacher's personal and professional details. The top section includes the teacher's name, email, and role designation, ensuring that key identification information is readily accessible. Below, a table displays specific details such as exam information, including the title, status, and date of scheduled or completed exams. This layout organizes the data in a clean, structured format, allowing users to access and manage teacher-related activities easily.

The TA (Teaching Assistant) Profile Page serves a similar purpose but is tailored for teaching assistants. At the top, the page highlights the assistant's name and email alongside their designated role. Below this, a detailed table outlines relevant data entries such as the date name, objective title, and the status of activities or objectives assigned.

2.4.5.12. Analysis Reports

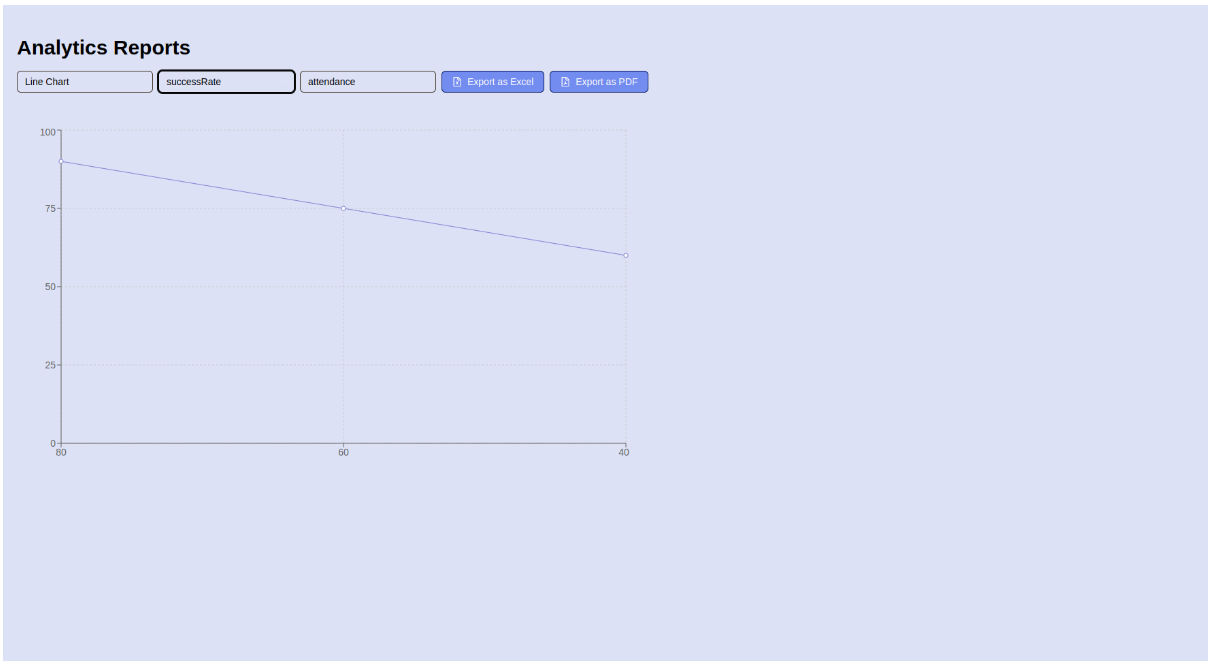
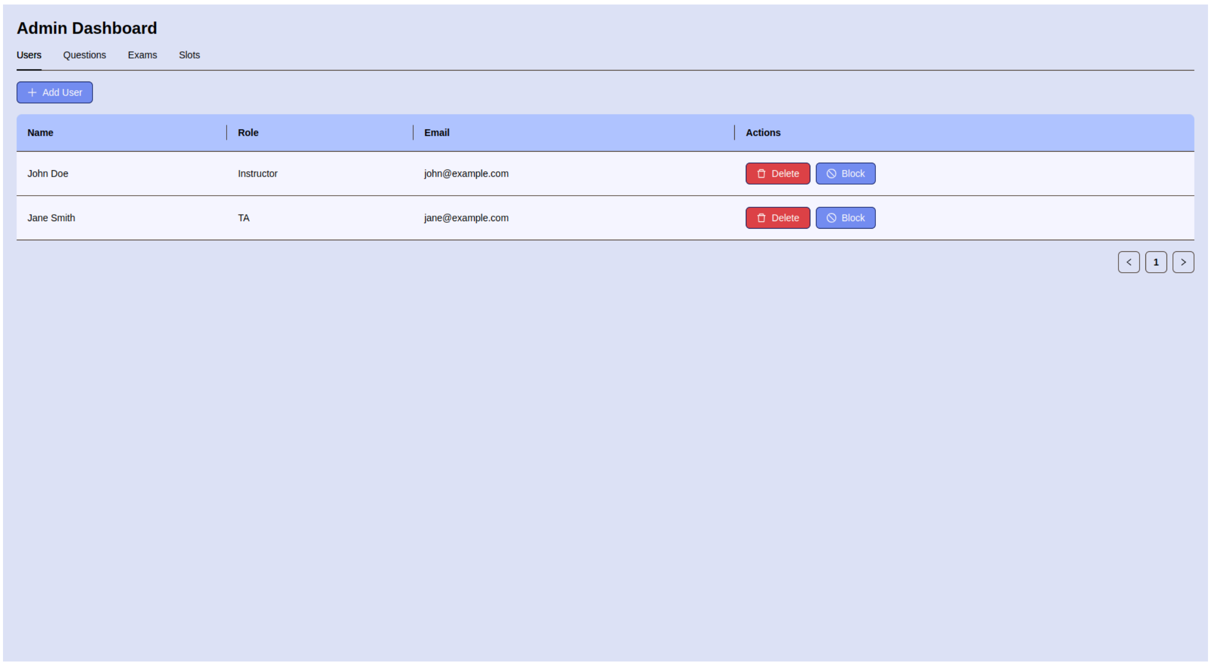


Figure 26: Analytic Reports

Intructors and TAs can create analysis reports based on several metrics of the exams and questions. Different charts are available for better visual interpretation.

2.4.5.13. Admin Dashboard



Name	Role	Email	Actions
John Doe	Instructor	john@example.com	Delete Block
Jane Smith	TA	jane@example.com	Delete Block

Figure 27: Admin Dashboard

Admins can manage the system via the given page. They can evaluate institutions' registry requests and also block or delete them. Admins can also see the questions, exams, and objections on the system.

3. Other Analysis Elements

3.1. Consideration of Various Factors in Engineering Design

3.1.1. Institutional Collaboration

Evalio collaborates with universities by providing a shared question database and modular exam creation tools. By allowing institutions to share non-private academic resources securely, Evalio creates an innovation in exam design.

3.1.2. Educational Efficiency

Evalio enhances educational efficiency by streamlining the exam preparation and management process. By automating repetitive tasks such as question organization, filtering, and report generation, the platform allows instructors to focus more on creating meaningful assessments rather than administrative tasks.

3.1.3. Constraints

To guarantee Evalio’s security, usability, and compliance with educational institutions' requirements, several constraints control its design and implementation. To guarantee Evalio’s security, usability, and compliance with educational institutions' requirements, several constraints control its design and implementation.

3.1.3.1. Time Constraints

The project must be completed within the timeline defined by the academic calendar, including sufficient time for development, testing, and deployment. Each module (question database, exam builder, portfolio manager, etc.) must be finalized in phases, allowing room for feedback and iterative improvements.

3.1.3.2. Budgetary Constraints

Evalio will be hosted using a cloud service to avoid the overhead of establishing and maintaining a dedicated server infrastructure. Cloud services provide a cost-efficient solution that aligns with the varying needs and scale of the platform. However, operational and potential scaling costs must be carefully managed to remain within the project’s budget. Additionally, introducing Evalio to universities and showcasing its benefits will require periodic target group analysis and a reasonable budget for promotional activities, ensuring alignment with the specific demands of academic institutions [2].

3.1.3.3. Data Privacy and Security Constraints:

The system will handle sensitive and confidential information, including exam questions, grading data, and instructor details. Role-based access control (RBAC) [3] must be strictly implemented to ensure that only authorized instructors and TAs can access the platform. Students should have no access to the system or its components. Robust authentication mechanisms, such as email verification and password protection, ensure that only approved users can interact with the platform.

3.1.3.4. Institution-Specific Requirements:

The platform must be flexible enough to support institution-specific guidelines, such as unique formats for uploaded documents, exam designs, and language preferences. These customizations must be implemented without compromising the consistency and core functionality of the system.

3.1.4. Standards

3.1.4.1. System Modeling Standards

UML will ensure clear and consistent visualization of the system's architecture and workflows. Diagrams such as use case, sequence, and class diagrams will follow UML 2.5 [4], offering a standard representation of the system’s design and interactions.

3.1.4.2. Documentation Standards

The structure and format of all requirements documentation will follow IEEE 29148 [5] standards for Systems and Software Engineering Requirements Engineering. This ensures clarity, completeness, and traceability of functional and non-functional requirements throughout the development lifecycle.

3.1.4.3. Data Privacy and Security Standards

Given that Evalio deals with sensitive academic data such as exam questions and instructor records, the system will comply with ISO 27001 [6] standards for Information Security Management. Role-based access control (RBAC), encryption protocols, and secure authentication mechanisms will be implemented to maintain data confidentiality, integrity, and availability.

3.2. Risks and Alternatives

3.2.1. Unauthorized Access to Sensitive Data

- **Risk:**
  Since the system stores sensitive data, such as exam questions and results, there is a risk that unauthorized individuals (e.g., students) could attempt to reach the application. This would cause the potential misuse of confidential information.

- **Alternative:**
  Implement strict role-based access control (RBAC) to ensure only authorized users can access the platform. User accounts for TAs and instructors will not be created through self-registration but registered and verified by the institution's designated admin or head. This process guarantees that only verified personnel can use the system. Additionally, detailed access logs for auditing must be maintained and periodically reviewed to detect and prevent unauthorized activities.

3.2.2. Exam Generation

- **Risk:**
  The exam generation process may need help with syntax errors in LaTeX codes, compatibility issues with differing packages, and performance bottlenecks for large exams.

- **Alternative:**
  To address these challenges, validation checks will be implemented to ensure LaTeX codes are error-free when stored, standardized templates and packages will be used to avoid compatibility issues, asynchronous processing will handle large exam generation efficiently, and a preview feature will allow instructors to review and verify exam layouts before final export [7].

3.2.3. Real-Time Slot Management

- **Risk:**
  Managing the real-time platform for slot selection may face challenges such as ensuring one user's selection cannot be overwritten by another, handling concurrent user actions accurately, and maintaining system responsiveness under heavy traffic.

- **Alternative:**
  To address these challenges, locking mechanisms will be implemented to immediately secure a slot upon selection, ensuring no overwrites occur. Concurrent actions will be handled using transaction-safe database operations and conflict resolution strategies. Additionally, load testing and optimization techniques will be applied to ensure system responsiveness and reliability during peak usage.

3.2.4. Failure to Meet Deadlines

- **Risk:**
  Developing all three planned modules (Question Database, Objection Session Manager, and Portfolio Builder) might not be feasible within the project timeline.

- **Alternative:**
  Focus on delivering a fully functional Question Database and Exam Builder (Module 1) as a priority. The remaining modules can be implemented incrementally, starting with the Objection Session Manager. This ensures that the core functionality of the system is delivered on time.

3.2.5. Data Loss

- **Risk:**
  System or server failures could lead to data loss or corruption, affecting uploaded documents, questions, and generated reports [8].
- **Alternative:**
  Use a cloud-based solution with automated backups [8].

3.3. Project Plan

Table 1: Factors that can affect analysis and design.

	Effect Level	Effect
Public Health	1	Limited relevance: the system can support remote assessments during health crises.
User Safety	9	Secure authentication prevents unauthorized access to question databases and exams.
Public Welfare	2	With a global question database and high-quality assessments, it improves fairness in education.
Global Factors	5	Promotes multi-university collaboration and compatibility across institutions globally.
Cultural Factors	2	Allow to edit and create questions from the global database according to varying educational styles and assessment
Social Factors	3	It meets the growing demand for digital tools in modern education, which enhances exam convenience.
Environmental	4	Reduces paper usage and supports sustainability

Factors		through digital exam processes
Economic Factors	6	Lower exam preparation costs save resources and improve operational efficiency.

Table 2: List of work packages

WP#	**Work package title**	**Leader**	**Members involved**
WP1	CS 491/2 Reports	Ahmet Reşat Demir	All Members
WP2	Microservice Architecture Implementation	Burak Demirel	All Members
WP3	Data Storage Implementation	Yusuf Toraman	All Members
WP4	Frontend Development	Eren Arım	All Members
WP5	Backend Development	Dilara Mandıracı	All Members
WP6	Testing and QA	Ahmet Reşat Demir	All Members

Table 3: Work Package 1 - CS491/2 Reports

WP 1: *CS491/2 Reports*			
Start date: *Mid-Oct 2024*. **End date:** *Mid-May 2025*			
Leader:	Ahmet Reşat Demir	**Members involved:**	*All Team Members*
Objectives: *This work package aims to prepare key project documentation, including the Specification Report and the Analysis & Requirements Report. These reports define the project scope, system requirements, and key modules, ensuring clarity for team members and evaluators.*			

Tasks: ***Task 1.1 Specification Report*****:** *The Specification Report defines the functional and non-functional requirements of the system. It outlines the key modules (Question Service, Exam Builder) and establishes the project's scope, constraints, and deliverables.* ***Task 1.2 Analysis & Requirements Report*****:** *The Analysis & Requirements Report provides a detailed analysis of the system's architecture, design, and use cases. It includes UML diagrams such as class, sequence, use case diagrams and functional workflows.* ***Task 1.3 Documenting*****:** *Documentation of the source code.*
Deliverables ***D1.1:*** *Specification Report – A finalized document outlining project requirements and scope.* ***D1.2:*** *Analysis & Requirements Report – Document detailing system models, use cases, and architectural design.*

Table 4: Work Package 2 - Microservice Architecture Implementation

WP 2: *Microservice Architecture Implementation*			
Start date: *01.12.2025.* **End date:** *Early-Jan 2025*			
Leader:	*Burak Demirel*	**Members involved:**	*All Team Members*
Objectives: *This work package includes implementing and maintaining the microservice architecture. Construction of the services and their connection to the gateway service is done in this package.*			
Tasks: ***Task 2.1 Gateway Service Establishment*****:** *Construct the gateway service and make the appropriate configuration.* ***Task 2.2 Consul Integration*****:** *Add consul dependency to register services.* ***Task 2.3 Service Registry*****:** *Construct the main skeleton of the services and register them to the consul.* ***Task 2.4 Docker Integration*****:** *Add docker configuration to set a dockerized working environment*			

***Task 2.5 Scripts**: Add bash scripts to run docker containers robustly.*
Deliverables ***D2.1:*** *Source Code*

Table 5: Work Package 3 - Data Storage Implementation

WP 3: *Data Storage Implementation*			
Start date: *Early-Jan 2025*. **End date:** *Mid-Feb 2025*			
Leader:	*Yusuf Toraman*	**Members involved:**	*Burak Demirel, Ahmet Reşat Demir, Dilara Mandıracı, Eren Arım*
Objectives: *This work package includes implementing and maintaining the DBMS and Redis. The entity-relationship model is implemented in this work package.*			
Tasks: ***Task 3.1 Construct Postgresql Container**: Write appropriate docker files to run the Postgresql database in a docker container.* ***Task 3.2 Construct Deployment DBMS**: Establish a deployment database on an appropriate cloud service* ***.Task 3.3 Construct Redis Container**: Write appropriate docker files to run Redis dependency in a docker container.* ***Task 3.4 Scripts**: Prepare necessary scripts to properly run the database and docker containers. Includes local development scripts and deployment scripts.*			
Deliverables ***D3.1:*** *Source Code*			

Table 6: Work Package 4 - Frontend Development

WP 4: *Frontend Development*			
Start date: *27.11.2024.* **End date:** *In Progress*			
Leader:	*Eren Arım*	**Members involved:**	*Burak Demirel, Ahmet Reşat Demir, Dilara Mandıracı, Yusuf Toraman*

Objectives: *The objective is to design and implement the user interface for Evalio. This includes developing user-friendly screens using React, ensuring interaction with the backend services, and delivering a functional and visually attractive frontend.*
Tasks: ***Task 4.1 UI Design***: *Design the user interface components for all required screens.* ***Task 4.2 Frontend Implementation***: *Implement the designed UI components using React. Develop functional screens such as the question management interface, exam builder dashboard, and login page.* ***Task 4.3: Frontend-Backend Integration:*** *Integrate the React frontend with backend APIs to ensure smooth data flow and real-time updates. Focus on API calls, data rendering, and error handling to ensure reliable functionality.*
Deliverables ***D4.1:*** *Completed UI Design Mockups and Prototypes.* ***D4.2:*** *Fully implemented React-based screens for core functionalities.* ***D4.3:*** *Integrated and functional frontend communicating with backend services.*

Table 7: Work Package 5 - Backend Development

WP 5: *Backend Development*			
Start date: *15.12.2024*. **End date:** *Early-May 2025*			
Leader:	*Dilara Mandıracı*	**Members involved:**	*<write the names of all team members working in this work package, excluding the work package leader >*
Objectives: *This work package includes developing backend services, excluding the gateway service.*			
Tasks: ***Task 5.1 Authentication Service Development***: *Implement authentication token structure and register request logic. Implement also the authenticating entities storage logic on the database. Connect all other services to the authentication service to verify all requests received by the system.*			

Task 5.2 Question Service Development: *Implement CRUD operations of questions. Includes the validation of LaTeX scripts.* ***Task 5.3 Exam Builder Service Development***: *Implement CRUD operations of exams. Includes the validation of LaTeX scripts.* ***Task 5.4 Portfolio Builder Service Development***: *Implement the CRUD operations of the portfolio. Includes portfolio generation.* ***Task 5.5 Objection Service Development***: *Implement the CRUD operations of objection sessions. Implement the real-time and on-at-a-time access properties of slot selection.* ***Task 5.6 Analytics Service Development***: *Implement report generation logic. Various reports can be created according to the metadata gathered for the exams and questions from respective services.* ***Task 5.7 Logging Service Development***: *Implement the system's logging mechanism.*
Deliverables ***D5.1:*** *Source Code*

Table 8: Work Package 6 - Testing & QA

WP 6: *Testing & QA*			
Start date: *Early-Jan 2025*. **End date:** *Early-May 2025*			
Leader:	*Ahmet Reşat Demir*	**Members involved:**	*Dilara Mandıracı, Eren Arım, Burak Demirel, Yusuf Toraman*
Objectives: *The objective is to ensure the system meets the required functionality, performance, and security standards. Comprehensive testing will be conducted to identify and resolve bugs, verify system behavior, and validate that the application operates correctly under different conditions.*			
Tasks: ***Task 6.1 Unit and Integration Testing***: *Perform unit testing for individual components and integration testing to verify that microservices (Question Service, Exam Builder) interact ideally.*			

Task 6.2 Performance and Security Testing: *Evaluate system performance under load and ensure security features such as role-based access control and data protection function as intended.*
Deliverables ***D6.1:*** *Unit and Integration Test Reports.* ***D6.2:*** *Performance and Security Testing Reports.*

3.4. Ensuring Proper Teamwork

Effective teamwork is essential for the success of any project, and our approach combines structured processes, collaborative tools, and strong personal dynamics to achieve strong collaboration. Below are the key practices and strategies we employ to ensure proper teamwork throughout the development of Evalio:

3.4.1. Task Management and Sprint Planning

We utilize JIRA [9] as our project management tool, allowing us to maintain clear task organization and efficient progress tracking. Our team follows two-week sprints, with tasks categorized under distinct epics representing key project areas, such as system modules and features. This structure enables us to break down complex objectives into manageable parts and ensures clarity in task distribution.

To estimate task complexity and workload, we adopt the Fibonacci sequence for weight estimation. This approach ensures tasks are distributed among team members based on complexity, promoting balanced workloads and equal contributions. Additionally, we use a structured task workflow in JIRA, with five columns for tracking progress: To-Do, Ready For Dev, In Dev, In Test, and Done. This clear status tracking enhances transparency and ensures everyone knows a task's progress or dependencies.

3.4.2. Independent Workflows with Microservices Architecture

We have adopted a microservices architecture to minimize interdependencies and ensure that errors or delays do not hinder individual team members in others' tasks. This approach enables each team member to focus on independent components of the project, reducing bottlenecks and improving overall efficiency.

3.4.3. Shared Knowledge and Documentation

We maintain a team logbook documenting our progress, decisions, and completed tasks. This logbook is an essential reference for tracking past work and ensuring continuity. Additionally, we use a shared wiki to store information and resources critical for all team members, such as system specifications, design guidelines, and best practices. This shared knowledge base improves accessibility and reduces the need for repeated discussions or clarifications [10].

3.4.4. Communication and Collaboration Tools

Our team uses multiple tools to facilitate communication and collaboration:

- **Zoom:** Used for meetings and pair programming sessions.
- **Slack:** Used for instant messaging to address essential issues or provide quick updates.

These tools ensure that all team members remain connected and aligned, regardless of location or time.

3.4.5. Team Synergy and Experience

Having worked together on projects for the past two years, our team has developed a deep understanding of each other's strengths, weaknesses, and work styles. This familiarity allows us to assign tasks strategically and leverage individual expertise. As a result, our teamwork has become more fluid and productive, enabling us to handle challenges effectively.

3.5. Ethics and Professional Responsibilities

Ethical values and professional responsibility guide every aspect of our project. Evalio is being developed with a focus on fairness, security, and integrity to support educators and institutions effectively.

3.5.1. Ease of Use and Efficiency

Evalio is designed to be an accessible and user-friendly platform for instructors and TAs. Automating exam preparation tasks and offering clear flows reduces complexity and saves educators time.

3.5.2. Protection of Sensitive Information

The platform handles critical academic data, such as exam questions and student performance metrics, which requires strict protection. Role-based access ensures only verified users can interact with the system, safeguarding sensitive information from unauthorized access.

3.5.3. Ensuring Academic Fairness

As a system closely tied to exam preparation, Evalio upholds academic fairness by maintaining the confidentiality of exams and ensuring they are not misused. This responsibility is fundamental to supporting the credibility of assessments.

3.5.4. Accountability in Development

The development process is guided by clear documentation and open communication within the team. This ensures that decisions are made responsibly and reflect end-users needs, resulting in a system educators can trust.

3.5.5. Prioritizing Instructor Needs

Evalio’s primary focus is to support educators by reducing administrative issues and improving efficiency. By addressing real-world challenges instructors and TAs face, the system is built to be a dependable tool that aligns with their needs.

3.6. Planning for New Knowledge and Learning Strategies

As we develop Evalio, we understand the need to continually expand our knowledge and learn new strategies to address our challenges. Below are the key areas we aim to improve and how we plan to learn the necessary skills.

3.6.1. Secure Software Development

Since Evalio involves sensitive data, learning advanced security practices is a priority. This includes encryption techniques, role-based access control, and secure database management. We plan to enhance our knowledge in these areas through video-based courses like Udemy, focusing on hands-on implementation.

3.6.2. Microservices Architecture

We must strengthen our understanding of this approach since Evalio is built using a microservices architecture to ensure modularity and independence. We plan to combine team knowledge (know-how) with external resources to achieve this. Team members with prior experience will guide the project through implementation while we also explore Medium articles and video courses to cover any knowledge gaps.

3.6.3. Team Collaboration and Project Management

We use JIRA for task management, and we plan to learn practical JIRA usage by sharing knowledge and learning from each other’s experiences.

3.6.4. Feedback-Driven Learning

We believe that feedback is essential to growth. Testing our system, gathering comments, and analyzing challenges together will help us improve Evalio. By combining online courses, expert resources, peer collaboration, and real-world feedback, we plan to get new knowledge and learn the necessary skills to develop Evalio.

4. References

[1] Purdue University, "Creating Exams," [Online]. Available: https://www.purdue.edu/innovativelearning/teaching/module/creating-exams/. [Accessed: Nov. 21, 2024].

[2] Ginani, Ahmed. “Budget Overruns in Software Development: 8 Factors to Know.” *Medium*, 9 Sept. 2024, medium.com/@elijah_williams_agc/budget-overruns-in-software-development-8-factors-to-know-60d824a8378c. Accessed 15 Dec. 2024.

[3] Shah, Eshika. “Exploring Role-Based Access Control (RBAC) - Eshika Shah - Medium.” *Medium*, Medium, 30 July 2024, medium.com/@eshikashah2001/exploring-role-based-access-control-rbac-3284 3370e604.

[4] “About the Unified Modeling Language Specification Version 2.5.1.” *Www.omg.org*, www.omg.org/spec/UML/2.5.1/About-UML.

[5] “IEEE Standards Association.” *IEEE Standards Association*, standards.ieee.org/ieee/29148/6937/.

[6] “ISO/IEC 27001 Standard – Information Security Management Systems.” *ISO*, 2022, www.iso.org/standard/27001.

[7] GeeksforGeeks. “RealTime Data Processing: Challenges and Solutions for Streaming Data.” GeeksforGeeks, GeeksforGeeks, 25 Aug. 2024, www.geeksforgeeks.org/real-time-data-processing-challenges-and-solutions-for-streaming-data/.

[8] CFI Team. “Data Loss.” Corporate Finance Institute, 25 Dec. 2022, corporatefinanceinstitute.com/resources/data-science/data-loss/.

[9] Atlassian. “Jira.” *Atlassian*, 2024, www.atlassian.com/software/jira.

[10] Senoro, Prince. “The Power of Documentation in Software Development: Elevating Team Collaboration and Business Success – Liberty Fox

Technologies." Liberty Fox Technologies, 4 Sept. 2024, www.libertyfoxtech.com/the-power-of-documentation-in-software-development-elevating-team-collaboration-and-business-success/. Accessed 15 Dec. 2024.