Bilkent University

Department of Computer Engineering

Senior Design Project – CS 491

T2416
Edux

Project Analysis and Requirements Report

Bilginer Oral	22103163	bilginer.oral@ug.bilkent.edu.tr
Murat Çağrı Kara	22102505	cagri.kara@ug.bilkent.edu.tr
Cahit Ediz Civan	22003206	ediz.civan@ug.bilkent.edu.tr
Efe Kaan Fidancı	22102589	kaan.fidanci@ug.bilkent.edu.tr
Görkem Kadir Solun	22003214	kadir.solun@ug.bilkent.edu.tr

Supervisor: Doruk Öner

Course Instructors: Atakan Erdem, Mert Bıçakçı

December 13th, 2024

1. Introduction

In today's fast-paced world, learning and staying ahead requires more than just access to resources. It demands efficiency, adaptability, and focus. While the internet and digital platforms have made educational content more accessible than ever, they've also created an ocean of information that often overwhelms learners. Juggling multiple resources, managing time, and ensuring progress can be intimidating, but many find it hard to keep their knowledge and reach their goals.

This is where Edux steps in as a game-changer. Edux changes the learning process by utilizing the power of large language models (LLMs) and personalized tools to make learning more effective and interactive. With features like detailed explanations powered by LLMs, customizable study aids, and dynamic progress tracking, Edux transforms scattered study efforts into structured, goal-oriented journeys. By tailoring study schedules to individual needs and generating interactive tools such as flashcards, quizzes, and skill trees from user-uploaded content, Edux simplifies complex learning processes while boosting understanding and retention. Moreover, Edux also bridges the gap between students and instructors, offering collaborative tools like task assignments and automated evaluations that ensure academic integrity and consistency between theoretical and practical work. With plans for future expansion into mobile and tablets, Edux is poised to meet the evolving demands of modern education, though these developments fall outside the current scope of the senior design project.

This report details the structure and components of Edux, starting with an introduction that outlines the project's purpose and scope. The current system is described along with its limitations. The proposed system section provides an overview of the system planned at the end of the senior design project, followed by a breakdown of its functional requirements, nonfunctional requirements, and pseudo requirements. System models are presented to illustrate various aspects of the system, including scenarios, use case models, object and class models, dynamic models, user interface navigational paths and screen mock-ups. Additional analysis covers external factors affecting the system design, such as constraints, relevant standards, risks and alternatives, the project plan, teamwork considerations, ethics and professional responsibilities, and strategies for acquiring new knowledge.

2. Current System

2.1. Current Situation and Limitations of Existing Learning Solutions

In today's educational landscape, the proliferation of digital resources has created a paradox for learners. While access to information is easier than ever, the overwhelming content can hinder effective learning. Students often find themselves juggling multiple platforms and resources, struggling to manage their time, and needing a clear path to reach their educational goals. This fragmentation leads to needing help retaining knowledge and staying focused, ultimately impacting academic performance.

Several AI-powered platforms have emerged to address these challenges by offering personalized study aids and resource management tools. Platforms like StudyFetch, Flint, and Mindgrasp.ai provide features such as AI-generated notes, flashcards, quizzes, and even personalized tutoring. These tools aim to transform study materials into interactive learning aids, enhancing engagement and efficiency. However, these existing solutions have notable limitations:

- **Lack of Integrated Learning Paths**: Many platforms focus on content transformation without providing structured learning paths that guide students through their educational journey.
- **Insufficient Personalization**: While some offer personalized study aids, they often lack customized study schedules based on individual syllabi, assessment performance, and time availability.
- **Limited Collaboration Tools**: More facilitation of meaningful interaction between learners and instructors is needed to align theoretical knowledge with practical application.
- **Overemphasis on Content Generation**: The focus tends to be on generating study aids from existing materials rather than creating unique practice questions that challenge the learner and ensure academic integrity.

For example, StudyFetch transforms individual study materials into interactive tools and offers an AI tutor but does not emphasize collaborative features or structured learning paths. Flint is geared towards educators creating AI-driven activities within institutional settings, offering less autonomy to individual learners. Mindgrasp.ai excels in content transformation and AI tutoring but does not provide personalized study schedules or progress tracking aligned with individual goals.

2.2. Comprehensive and Personalized Learning Solution

While competitors offer valuable features, Edux sets itself apart by integrating these functionalities into a single, cohesive platform:

- **Holistic Approach**: Unlike platforms focusing solely on content generation or AI tutoring, Edux provides a comprehensive learning environment that addresses all aspects of the educational process.
- **Enhanced Personalization**: Edux's ability to tailor study schedules and generate interactive tools based on individual needs ensures that learners receive support that aligns closely with their goals and learning styles.

- **Facilitated Collaboration**: Edux fosters a collaborative learning environment that enhances practical understanding and skill application by bridging the gap between students and instructors.
- **Future-Ready Features**: Integrating advanced technologies like RAG positions Edux at the forefront of educational innovation, offering capabilities that competitors lack.

2.3. Edux vs. Competitors: Addressing the Gaps

Edux enters this landscape as a transformative platform designed to overcome the limitations of current solutions by offering a holistic and personalized learning experience. Leveraging advanced AI technologies and modern development frameworks, Edux provides features that enhance learning efficiency, adaptability, and focus. Below, we detail the current features, ongoing developments, and planned improvements.

Current Features Implemented

1. **Interactive Learning Tools**: Edux generates flashcards and quizzes from user-uploaded content, transforming static study materials into engaging aids that improve understanding and retention. These tools are available via the platform's dashboard.
2. **Dynamic Progress Tracking**: Learners can monitor their progress in real time. The system tracks completed activities and updates performance metrics, helping users identify strengths and weaknesses.
3. **Collaboration Features**: Task assignments and chat functionalities facilitate collaboration between students and instructors, enhancing the bridge between theoretical knowledge and practical application.

Technologies Used:

- **Frontend**: React.js [1] for the user interface, ensuring a responsive and interactive experience.
- **Backend**: FastAPI [2] handles API requests, gRPC [3] is used for efficient communication between services, and PostgreSQL [4] is used for database management.

Features in Progress

1. **Skill Trees**: The development of interactive skill trees is underway. This feature will allow learners to visualize their progress and see how individual topics connect to broader learning objectives.
2. **Course Creation Tools**: We build tools to help instructors and learners create and manage custom courses, allowing for greater flexibility and personalization.

Planned Features

1. **Retrieval-Augmented Generation (RAG)**: This feature will enhance personalized learning by generating unique practice questions. The goal is to improve academic integrity and provide deeper personalization.
2. **Personalized Study Schedules**: We plan to develop automated study plans tailored to individual syllabi, assessment results, and available study time. This will help learners manage their time effectively and stay on track.

Current solutions on the market provide fragmented assistance, often focusing on isolated aspects of learning. Edux addresses these gaps by integrating multiple features into a single platform:

- **Engagement**: Interactive tools such as flashcards and quizzes make learning dynamic.
- **Adaptability**: Progress tracking helps users adjust their learning strategies.
- **Collaboration**: Task assignments and chat features streamline communication between students and instructors.
- **Future Personalization**: Planned features like RAG and personalized schedules will further unify the learning experience.

By combining implemented features, ongoing developments, and future innovations, Edux transforms scattered study efforts into structured, goal-oriented journeys. This comprehensive approach empowers learners to navigate the overwhelming sea of information confidently and efficiently, enhancing academic performance and lifelong learning.

3. Proposed System

3.1. Overview

Edux is a web application designed to improve how people learn and assess knowledge. It supports learners by providing tools and features that enhance understanding, retention, and organization.

Edux allows learners to upload various study resources, including scholarly articles, textbooks, slides, syllabi, and other materials. Using LLMs, Edux provides detailed explanations of these resources, either page-by-page or in the context of the entire content. Supported formats include images, reports, slides, and books, with plans to integrate voice recordings in the future.

Edux includes interactive learning tools like flashcards, skill trees and quizzes to reinforce knowledge and assess progress. Flashcards and quizzes are created based on the user's uploaded content to their individual study. Skill trees can be predefined individual studies with hierarchical structures or custom skill trees created by learners based on the content uploaded to their individual studies.

To help learners manage their time effectively, Edux generates personalized study schedules. These schedules are customized based on a user's syllabus, can be manually entered or uploaded, and include performance in assessment tools and time spent on these assessments. They promote efficient planning by identifying areas that require improvement and helping learners focus their efforts where it matters most.

The platform also will offer user-instructor relationships for engineering students. Instructors can assign tasks that will include reports and coding projects to students in their classes. Edux will include a model that will evaluate whether the submitted report aligns with the assigned code, ensuring consistency between theory and practices and will promote academic integrity.

Edux is designed with plans to introduce paid tiers in the future, offering advanced features to enhance the learning experience. One such feature will be Retrieval-Augmented Generation (RAG), which will enable users to create content such as practice questions similar to publicly available past exams while ensuring the questions are unique. Additionally, RAG will provide learners with highly accurate and contextually relevant responses, helping them better understand complex topics and refine their knowledge effectively.

Edux combines advanced technologies, interactive tools, and personalized features to create a comprehensive learning platform. Its focus on structured learning paths, time management, and collaboration between learners and instructors makes it an effective tool for education.

3.2. Functional Requirements

The functional requirements of Edux define the core capabilities necessary to deliver its intended features and user experience effectively. These requirements span user management, content processing, interactive learning tools, personalized scheduling, collaboration, analytics, and other essential functionalities.

3.2.1. User Management and Authentication

Edux must support multiple user roles, including learners, instructors, and administrators, each with distinct functionalities. Each user will be registered to the system and saved to the database with a unique ID. Also, secure authentication mechanisms are required, enabling users to log in or register using email, passwords, or third-party integrations like Google or Microsoft accounts. Additionally, the platform should provide comprehensive account management, allowing users to update profiles, reset passwords, and configure preferences with their unique user ID.

3.2.2. Resource Upload and Processing

A critical feature of Edux is its ability to process various types of educational materials. Users should be able to upload resources in multiple formats, such as PDFs, images, presentations, and text documents. Advanced content parsing is required to extract and analyze textual data from these uploads. The platform must offer explanations either on a page-by-page basis or across entire documents. Future enhancements include voice integration, allowing learners to upload or record voice notes for transcription and further analysis.

3.2.3. Interactive Learning Tools

Edux uses interactive tools to enhance learning. It automatically generates flashcards from uploaded materials, highlighting key concepts while allowing users to customize or create their own, which is linked to their unique ID. The quiz feature supports multiple formats, including multiple-choice, fill-in-the-blank, and short answers, to reinforce learning. Skill trees, both predefined and customizable, visually guide users through hierarchical learning paths, offering a structured and flexible educational experience.

3.2.4. Personalized Study Schedules

The ability to generate personalized study schedules is a cornerstone of Edux. Schedules should be based on uploaded syllabi or manually entered data, with dynamic adjustments informed by performance in quizzes and time spent on tasks with their unique user ID. The platform should identify areas of weakness, allocate study time accordingly, and send notifications about deadlines, progress milestones, or missed sessions to keep learners on track.

3.2.5. Instructor-Student Collaboration

For students, Edux offers collaborative tools to enhance the learning process. Instructors should be able to assign tasks, including reports and coding projects, and use evaluation tools to ensure alignment between theoretical and practical components with their unique user ID and roles.

3.2.6. Reporting and Analysis

Edux must generate comprehensive reports to help learners and instructors track performance. Learners should receive insights into their progress, strengths, and areas needing improvement, while instructors should have access to dashboards detailing class performance, task completion rates, and individual student metrics. Time analytics will correlate learning activities with performance, enabling data-driven adjustments.

3.2.7. Learning Assistance with LLMs

The platform's integration of large language models is central to its functionality. Edux will provide detailed, context-aware explanations of uploaded content and support natural language queries, allowing learners to ask questions and receive precise answers. This enhances the understanding and retention of complex material.

3.2.8. Accessibility and User Interface

Ensuring accessibility is crucial for a diverse user base. Edux must feature a responsive design compatible with desktops, tablets, and smartphones. Accessibility features, such as screen reader support, adjustable font sizes, and high-contrast settings, will improve usability for individuals with disabilities. Multilingual support is also essential to serve users across different linguistic backgrounds. If the correct unique identifier is provided to the system, the corresponding materials will be supplied to the user (i.e., the files associated with that username and password will be delivered).

3.2.9. Data Security and Privacy

Robust security measures are essential to protect sensitive data. Encryption should safeguard user-uploaded files and personal information during transmission and storage. Role-based access control will ensure that only authorized users can view or manage specific data. Compliance with data protection regulations, such as GDPR [5], is mandatory to maintain user trust and legal conformity. Requests for individual study material that do not correspond to the unique ID will be declined.

3.2.10. Scalability and Performance

To ensure a seamless experience, Edux must maintain high availability during peak periods, such as exam seasons. Scalable infrastructure will accommodate growth in user numbers and uploaded content without performance degradation. Optimization techniques, including caching and efficient query handling, are critical for maintaining responsiveness and reliability.

3.3. Nonfunctional Requirements

3.3.1. Usability

Edux is designed for learners requiring a user-friendly interface that is intuitive, accessible, and easy to navigate. The platform must cater to the diverse needs and preferences of its users. Key usability features should include clearly labelled buttons accompanied by text or universally recognized icons to ensure clarity. The user interface (UI) should be compact and responsive, allowing students to efficiently access tools and resources, especially during busy study periods. The system shall ensure that 95% of users can complete common tasks, such as uploading content or generating flashcards, within 3 clicks or fewer.

3.3.2. Reliability

Ensuring no loss of data integrity in case of system failures is crucial, as these losses can be real-time data entries that have not been committed to the database and could be at risk during system failures. The system should maintain a data consistency rate of at least 99.9%, ensuring that data remains consistent and accurate even during system failures. Scheduled maintenance should be conducted during periods of minimal user activity to minimize disruption. Ensuring that user data is protected during interactions with API gateways and safeguarded against cyber-attacks is crucial. The system should achieve an uptime of at least 99.9% annually, which translates to no more than 8.76 hours of downtime per year.

3.3.3. Performance

Database queries, such as loading individual study materials in the chat screen and generating skill trees, flashcards, and quizzes, as well as reloading skill trees, should return results within milliseconds to a few seconds, depending on the complexity and size of the data, to ensure an optimal user experience. For instance, a query handling a few hundred records should ideally be completed in under 100 milliseconds, while more complex queries involving millions of records might take a few seconds. This order of magnitude ensures that users experience minimal delay, maintaining a responsive and efficient system. Moreover, Edux will utilize visual indicators to notify users during longer loading times and manage expectations. The average response time should not exceed 2 seconds under normal conditions and should not exceed 3 seconds during peak loads.

3.3.4. Supportability

The application must be accessible and function seamlessly across various devices and operating systems, including desktops, mobile phones, and tablets integration in the future scope of the project. The application should be compatible with at least 95% of the most commonly used devices. Support processes must align with industry standards and regulatory requirements, ensuring data security and user trust.

3.3.5. Scalability

Edux should seamlessly scale to accommodate increasing user traffic and a growing database of study materials. The system should be designed to handle higher loads without compromising performance or usability. An example metric to track Edux's ability to handle higher loads without compromising performance or usability is system throughput, measured in transactions per second (TPS). This metric captures the volume of transactions the system can process within a given timeframe, reflecting performance and scalability under varying loads. This approach can provide early detection of issues and proactive scaling. Moreover, the system shall handle at least 1000 simultaneous users without performance degradation.

3.4. Pseudo Requirements

3.4.1 Implementation Requirements

- Next.js will be used for the frontend and FastAPI will be used for the backend.
- MySQL database will be used to store user data and other business logic related data.
- Gemini API will be used as the Large Language Model API.
- Backend will consist of microservices that operate in Docker containers and orchestrated by Kubernetes.
- Git will be used as the version control tool and GitHub will be used to store the project
- Jira will be used as the project management tool.
- The project should run on any web browser.
- User files will be stored in Amazon S3 buckets.
- The system will be deployed on AWS EC2.

3.4.2 Economic Requirements

- The free subscription plan will use Gemini API as it is free. In the future, other LLM's that require payment might be available for paid plans.
- Next.js, FastAPI, MySQL, Docker, Kubernetes, the subscription plans we use for GitHub and Jira and Gemini API are all free.
- AWS costs should be limited to $50 until the project starts generating revenue. Once revenue is achieved, AWS expenses should not exceed 10% of the total revenue.

3.4.3 Ethical Requirements

- User data, files and metrics will not be shared with any third party or other users in the system.
- Files uploaded by a user will not be shared with any third party and will not be used in response generation for other users.
- Within the project, databases and interservice communication will be secured to prevent privacy breach.
- The system will comply with GDPR regulations for data storage and access.
- Any payment that goes through the system will be secure.

3.5. System Models

We utilized Visual Paradigm software [6] and adhered to UML [7] rules to create the following models.

3.5.1. Scenarios

1.

Use case

Create Individual Study

Participating actors

Learner

Entry conditions

User clicks the "Create Individual Study" button in the main page.

Exit conditions

Individual study is successfully created / terminated with an error.

Flow of Events

1. User gets to the home page through logging in or routing in from elsewhere and clicks the "Create Individual Study" button.
2. The user uploads material about the study such as syllabus.
3. Study is successfully created.

2.

Use case

Update Individual Study

Participating actors

Learner

Entry conditions

User clicks the "Update Individual Study" button through the hamburger button on the individual study's UI card.

Exit conditions

Update operation is completed.

Flow of Events

1. User clicks the "Update Individual Study" button through the hamburger button on the individual study's UI card.
2. User fills in the relevant fields, such as new individual study name or syllabus and clicks "Update".
3. Update operation is complete.

3.

Use case

Create Study Schedule

Participating actors

Flow of Events

1. User creates an individual study.

Learner

Entry conditions

As soon as an individual study is created, system starts generating a study schedule.

Exit conditions

Generation of weekly study plan is completed.

2. User uploads the syllabus, and other individual study materials.
3. Based on the provided content, the system generates a weekly study plan.
4. If user reuploads a syllabus, or if s/he didn't upload one during creating an individual study and uploading at a later time, the system regenerates an updated schedule instantly.

4.

Use case

Create Study Schedule

Participating actors

Learner

Entry conditions

As soon as an individual study is created, system starts generating a study schedule.

Exit conditions

Generation of weekly study plan is completed.

Flow of Events

1. User creates an individual study.
2. User uploads the syllabus, and other individual study materials.
3. Based on the provided content, the system generates a weekly study plan.
4. If the user reuploads a syllabus, or if s/he didn't upload one during creating an individual study and uploading at a later time, the system regenerates an updated schedule instantly.

5.

Use case

Create Chat

Participating actors

Learner

Entry conditions

An individual study must have already been created. User then clicks the "Create New

Flow of Events

1. User goes to the individual study's home page and clicks the "Create New Chat" button.
2. User enters a title for chat, and uploads slides if s/he wants.
3. Chat is created and slides are successfully uploaded to the system.

Chat" button in the relevant screen.

Exit conditions

New chat is successfully created.

6.

Use case

Create Chat

Participating actors

Learner

Entry conditions

An individual study must have already been created. User then clicks the "Create New Chat" button in the relevant screen.

Exit conditions

New chat is successfully created.

Flow of Events

1. User goes to the individual study's home page and clicks the "Create New Chat" button.
2. User enters a title for chat, and uploads slides if s/he wants.
3. Chat is created and slides are successfully uploaded to the system.

7.

Use case

Update Chat

Participating actors

Learner

Entry conditions

An individual study and a chat must have already been created. User then clicks the "Update Chat" button in the relevant UI component.

Exit conditions

Chat is successfully updated with the new data.

Flow of Events

1. User clicks the "Update Chat Button" from the ellipsis button on the chat.
2. User enters the fields to be updated, such as the chat's title.
3. S/he also uploads new slides if desired.
4. Or, the user may choose to delete the chat.
5. New chat data is saved on the system, or chat is deleted; depending on the action chosen by the user.

8.

Use case

Toggle Mode

Participating actors

User

Entry conditions

The user must be inside the homepage.

Exit conditions

The UI theme is changed.

Flow of Events

1. Settings icon is clicked on the main page.
2. Either one of "Toggle Dark Mode", "Toggle Light Mode" or "Toggle System Mode" is clicked.
3. The UI theme is set up accordingly.

9.

Use case

Edit Profile

Participating actors

User

Entry conditions

The user must be inside the "My Account" page and have clicked the relevant UI button.

Exit conditions

Necessary action is taken, such as "Update profile" or "Delete Account".

Flow of Events

1. User clicks on the profile icon in his/her account.
2. The user may edit the fields in his/her profile, such as name or email.
3. The user may also choose to delete their account as well.
4. Relevant action is then taken on the system.

10.

Use case

Generate Learning Content

Participating actors

Learner

Entry conditions

The user must have logged in and navigated

Flow of Events

1. User navigates to the chat screen.
2. The user may choose "Generate Skill Tree", "Generate Quiz", or "Generate Flashcards".
3. Based on the content of the chat history, (i.e. slide explanations,

to a chat screen in an individual study.

Exit conditions

Learning content is successfully generated, or an error occurred due to external APIs.

questions from user, etc.) the system generates a skill tree, or personalized quizzes or flashcards.

4. The generated contents are then saved to the system for later access.

11.

Use case

Sign In

Participating actors

User, Admin

Entry conditions

Must be at the login screen.

Exit conditions

Users or admins are successfully authenticated and navigated to the main page.

Flow of Events

1. Users or admins go to the login page and enter their credentials.
2. If they forgot their passwords, they click on the "Forgot password" button and renew their passwords.
3. If credentials match with what's saved on the system, they are authenticated and obtain a unique authentication key.
4. Based on the credentials, they are navigated to the relevant screens (e.g. admin panel, learner screen, instructor screen).

12.

Use case

Sign Up

Participating actors

User

Entry conditions

Must be at the sign up screen.

Exit conditions

Account is created and user is navigated to the relevant page.

Flow of Events

1. Users enter their account information on the sign up screen (i.e. name, email, password, etc.).
2. After verifying their emails, they are navigated to the role selection page, in which they can specify whether it's a learner account or an instructor account.
3. After role selection, they are navigated to the relevant homepages.

13.

Use case

Search

Participating actors

User

Entry conditions

The user must be logged in to the system, and the search bar can be accessed from the global navigation bar.

Exit conditions

Search query is made, and results are returned from the system.

Flow of Events

1. User logs in to the system.
2. From the global navigation bar, the user enters query keywords (e.g. individual study, chats, etc.)
3. Results are shown on-the-fly, as the user types in each character.
4. User then clicks on the search results and forwarded in to the relevant page.

14.

Use case

Ban User

Participating actors

Admin

Entry conditions

Admin must be logged in and authenticated. Users list menu must be opened.

Exit conditions

Selected user is banned.

Flow of Events

1. Admin logs in to the system.
2. Selects a user from the list, and clicks the "Ban" option.
3. Selected user is banned permanently from the system.

15.

Use case

Delete User Data

Participating actors

Admin

Entry conditions

Admin must be logged in and authenticated.

Flow of Events

1. Admin logs in to the system.
2. Selects a user from the list, and clicks the "Delete User Data" option.
3. Then selects which user data to remove from the system, e.g. files, personal data.

Users list menu must be opened.

Exit conditions

Selected user data is deleted from the system.

4. The selected data is removed from the system.

16.

Use case

Assign Project

Participating actors

Instructor

Entry conditions

Instructor must be at the homepage.

Exit conditions

Project is uploaded to the system for review.

Flow of Events

1. Instructor logs in and navigates to the system.
2. Instructor clicks the "Assign New Project" button and uploads the project documents (i.e. code, report) for review.
3. The system starts generating a comprehensive review report.

17.

Use case

View Accuracy

Participating actors

Instructor

Entry conditions

Instructor must be at the homepage, and have assigned a project already.

Exit conditions

Generated report is shown to the instructor.

Flow of Events

1. Instructor clicks on the project while at the mainpage.
2. If the system is completed generating the accuracy report, it is shown to the instructor in a nicely formatted manner.
3. If not, a warning message is displayed to notice the instructor that the report is still being generated.

3.5.2. Use Case Model

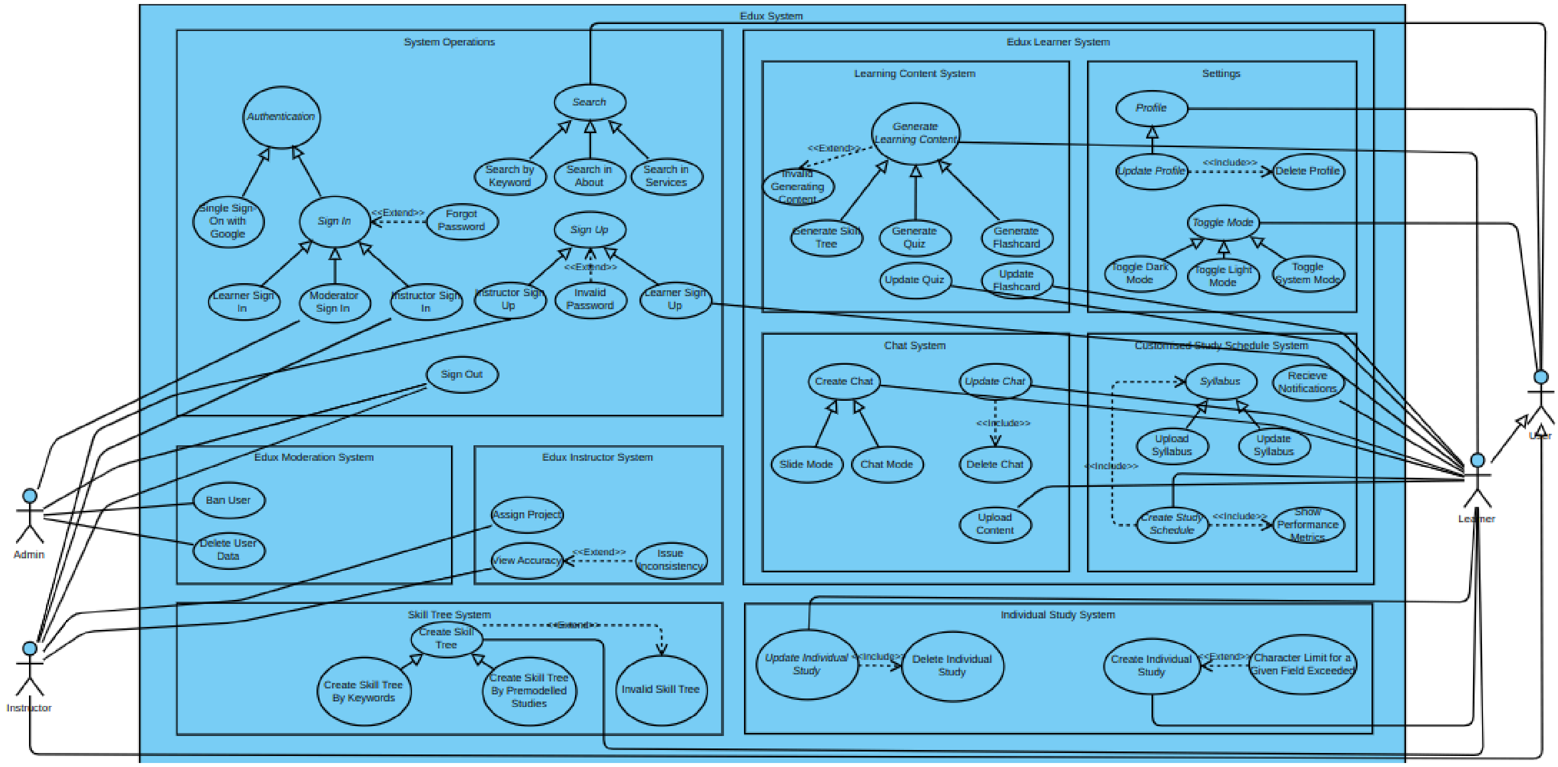


Figure 1: Use Case Diagram of Edux (For better resolution: drive)

3.5.3. Object and Class Model

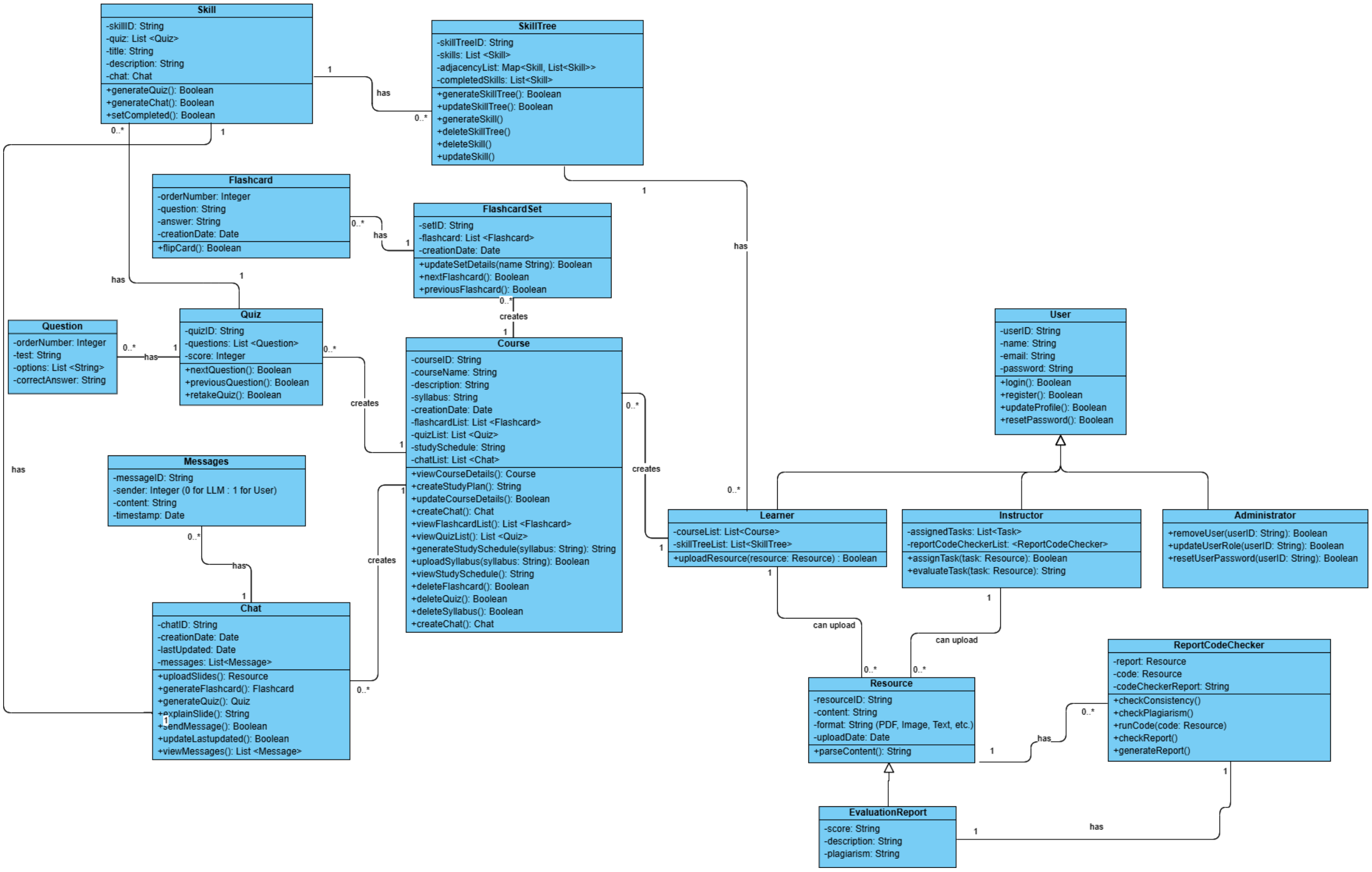


Figure 2: Class Diagram of the Design of Edux (For better resolution: drive)

3.5.3.1. User

Attributes:

userID: String
name: String
email: String
password: String

Methods:

login(): Boolean
register(): Boolean
updateProfile(): Boolean
resetPassword(): Boolean

Relationships:
Inheritance: Learner, Instructor, and Administrator inherit from User

3.5.3.2. Learner

Attributes:
courseList: List<Course>
skillTreeList: List<SkillTree>

Methods:
uploadResource(resource: Resource): Boolean

Relationships:
Associations: Can create and manage multiple Course instances. Can work with SkillTree objects. Can upload Resource files.

3.5.3.3. Instructor

Methods:
assignTask(task: Resource): Boolean
evaluateTask(task: Resource): String

Relationships:
Associations: Works with ReportCodeChecker to ensure consistency between reports and code submissions.

3.5.3.4. Administrator

Methods:
removeUser(userID: String): Boolean
updateUserRole(userID: String): Boolean
resetUserPassword(userID: String): Boolean
Responsibilities:
Manages users and their roles within the system.
Attributes:
assignedTasks: List<Task>
reportCodeCheckerList: List<ReportCodeChecker>

3.5.3.5. Course

Attributes:
courseID: String
courseName: String
description: String

syllabus: String
creationDate: Date
flashcardList: List<Flashcard>
quizList: List<Quiz>
studySchedule: String
chatList: List<Chat>

Methods:
createStudyPlan(): String
updateCourseDetails(): Boolean
createChat(): Chat
viewFlashcardList(): List<Flashcard>
viewQuizList(): List<Quiz>
generateStudySchedule(): String
deleteFlashcard(): Boolean
deleteQuiz(): Boolean

Relationships:
Associations: Contains Flashcard, Quiz, Chat, and StudySchedule.

3.5.3.6. ReportCodeChecker

Attributes:
report: Resource
code: Resource
codeCheckerReport: String

Methods:
checkConsistency()
checkPlagiarism()
findCode()
checkReport()
generateReport()

Relationships:
Associations: Works with Resource to check for consistency and plagiarism between reports and code submissions.

3.5.3.7. Resource

Attributes:
resourceID: String

content: String
format: String (PDF, Image, Text, etc.)
uploadDate: Date

Methods:
parseContent(): String

Responsibilities:
Represents the study materials and code resources uploaded by users.

3.5.3.8. Flashcard

Attributes:
orderNumber: Integer
question: String
answer: String
creationDate: Date

Methods:
flipCard(): Boolean

Relationships:
Belongs to: Course or FlashcardSet.

3.5.3.9. Quiz

Attributes:
quizID: String
questions: List<Question>
score: Integer

Methods:
nextQuestion(): Boolean
previousQuestion(): Boolean
retakeQuiz(): Boolean

3.5.3.10. SkillTree

Attributes:
skillTreeID: String
skills: List<Skill>
adjacencyList: List<Skill, List<Skill>>

Methods:
generateSkillTree(): Boolean
updateSkillTree(): Boolean
addSkill(): Boolean
deleteSkill(): Boolean

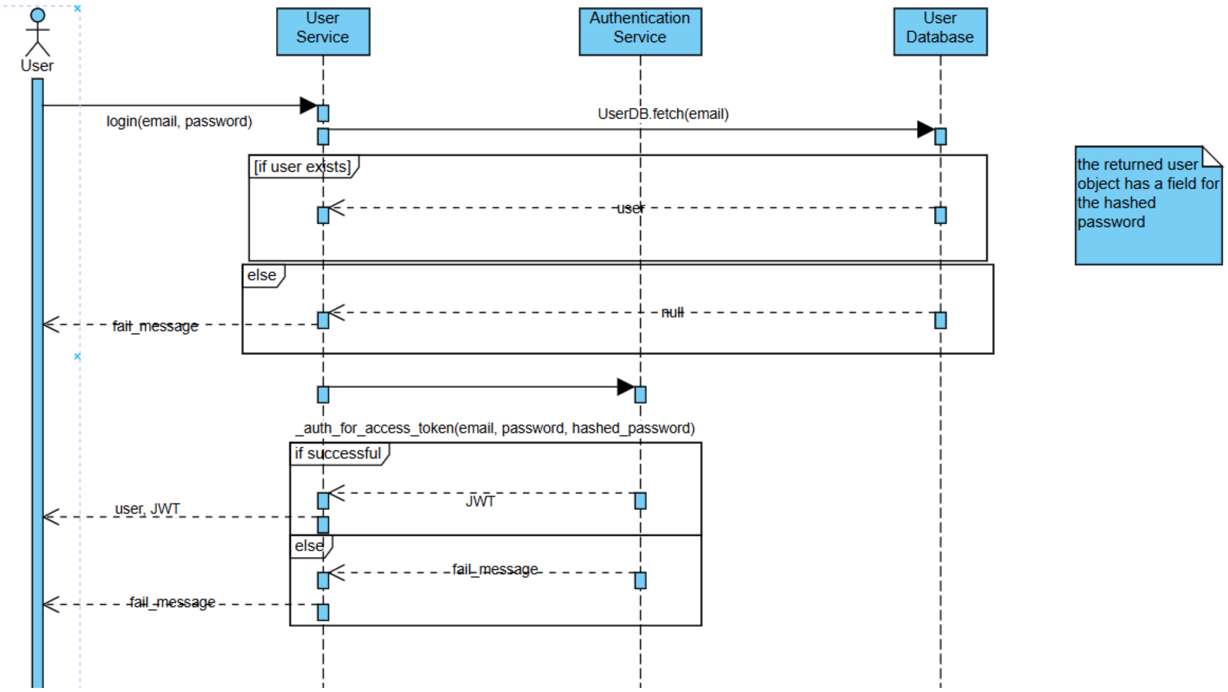
Relationships:
Associations: Contains multiple Skill instances.

3.5.3.11. Relationship Overview

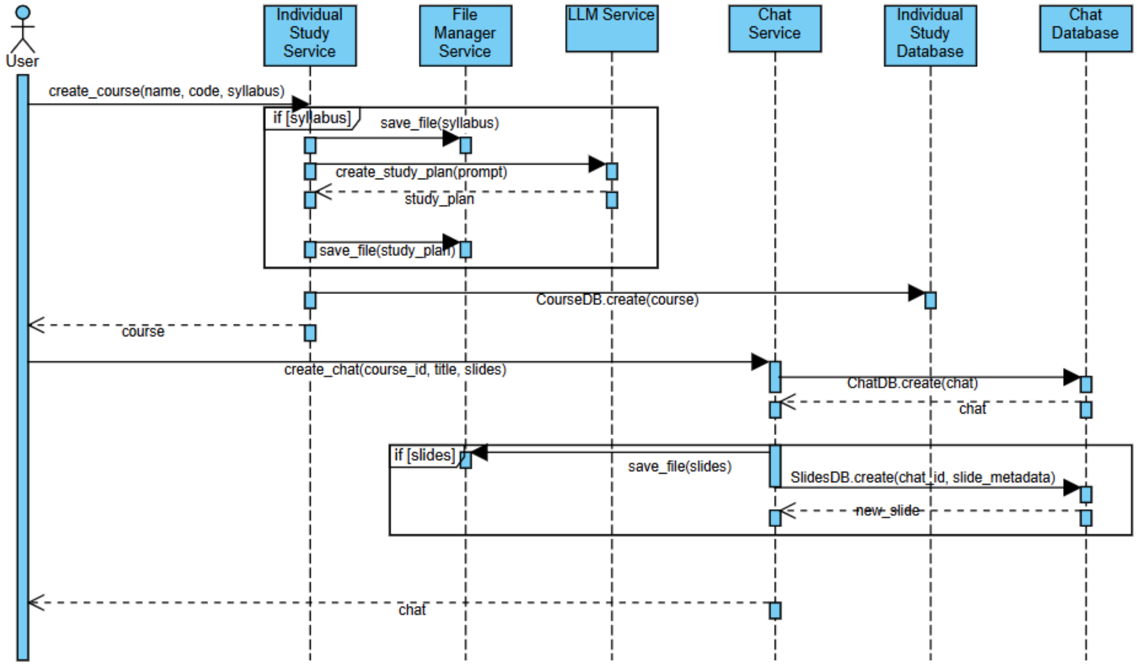
Inheritance: User is a parent class to Learner, Instructor, and Administrator.
Aggregation/Composition: Course contains Flashcard, Quiz, and Chat. SkillTree contains Skill.
Associations: Learner and Instructor interact with ReportCodeChecker. Instructor assigns tasks and evaluates submissions using ReportCodeChecker.

3.5.4. Dynamic Models

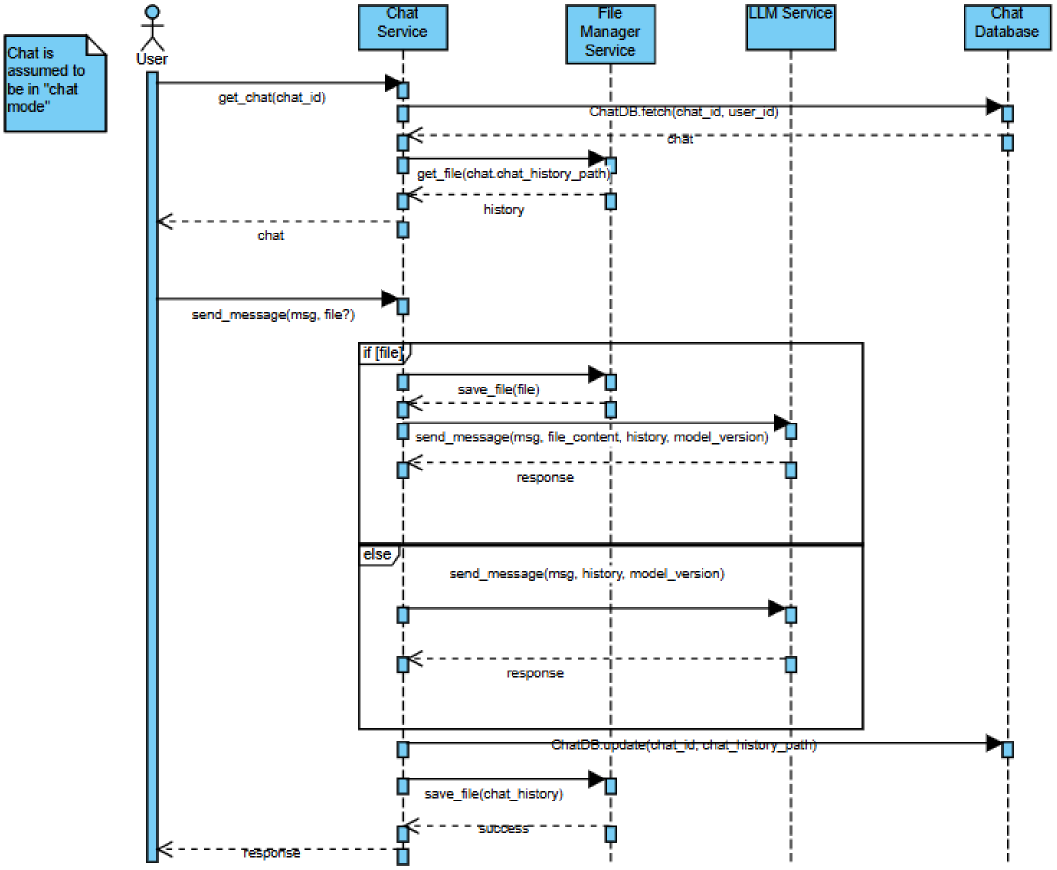
3.5.4.1. Sequence Diagrams



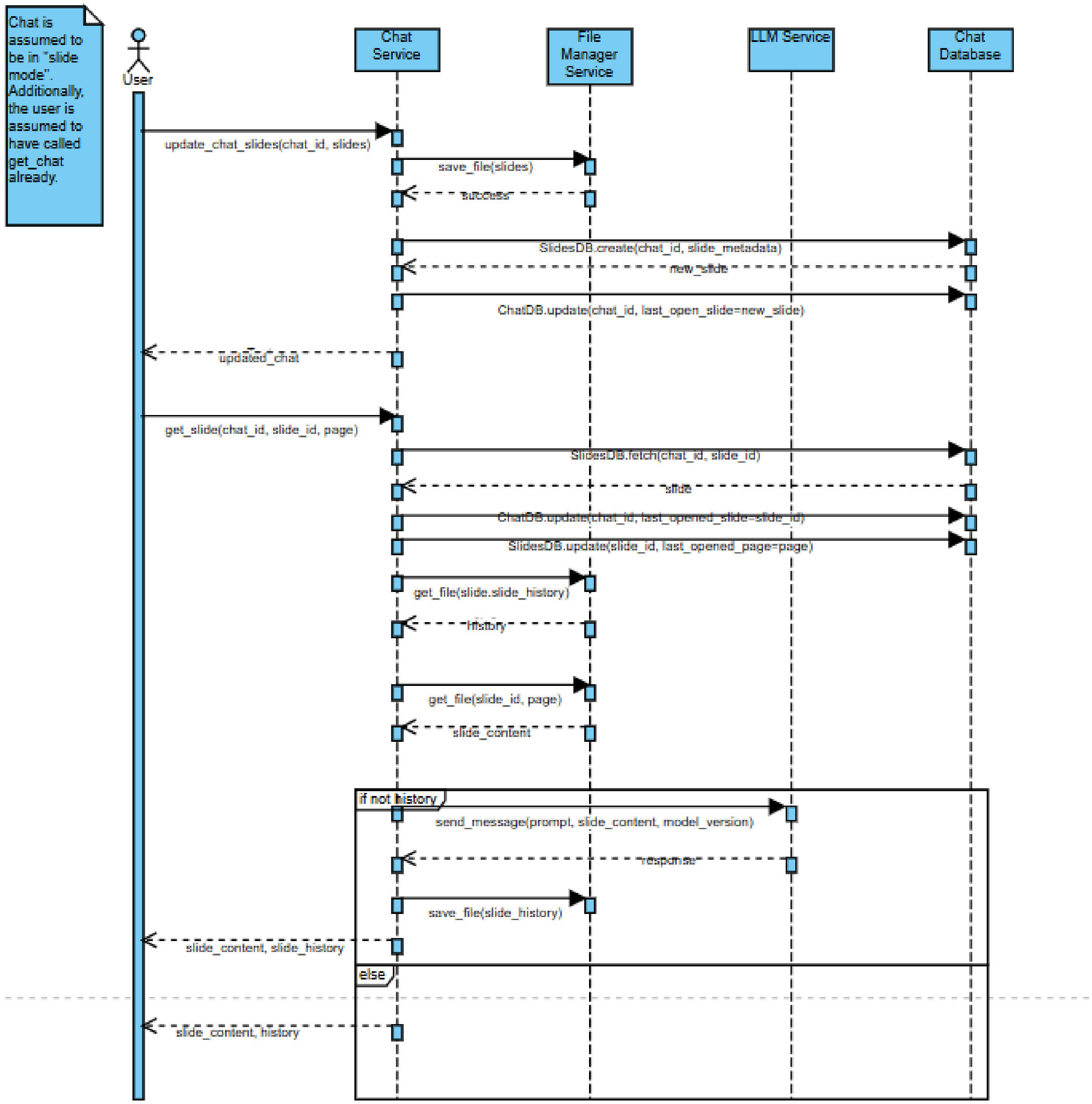
This diagram depicts the logging in process for users. The user provides their email and password and the system checks if a user exists with the given email from the database. If there is no such user the system returns a fail message, else, the system checks if the provided password is correct. If authentication is successful, the user receives a JSON Web Token which is required for authentication in every backend endpoint.



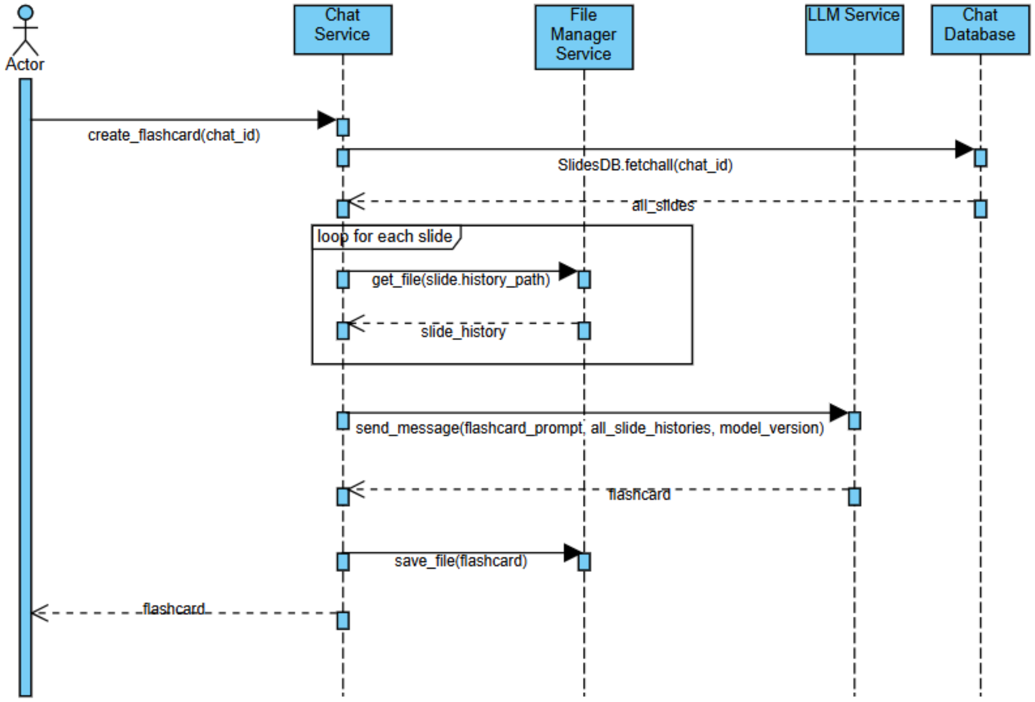
This diagram depicts how the user can create an individual study and create a chat within that individual study. If the user provides a syllabus during individual study creation, it is uploaded to the system and a study plan is generated automatically. After the individual study (course) is created, the user can create a chat within it. If the user provides slides during the creation, those slides are uploaded to the system.


Chat is assumed to be in "chat mode"
User
Chat Service
File Manager Service
LLM Service
Chat Database
get_chat(chat_id)
ChatDB.fetch(chat_id, user_id)
chat
get_file(chat.chat_history_path)
history
chat
send_message(msg, file?)
if [file]
save_file(file)
send_message(msg, file_content, history, model_version)
response
else
send_message(msg, history, model_version)
response
ChatDB.update(chat_id, chat_history_path)
save_file(chat_history)
success
response

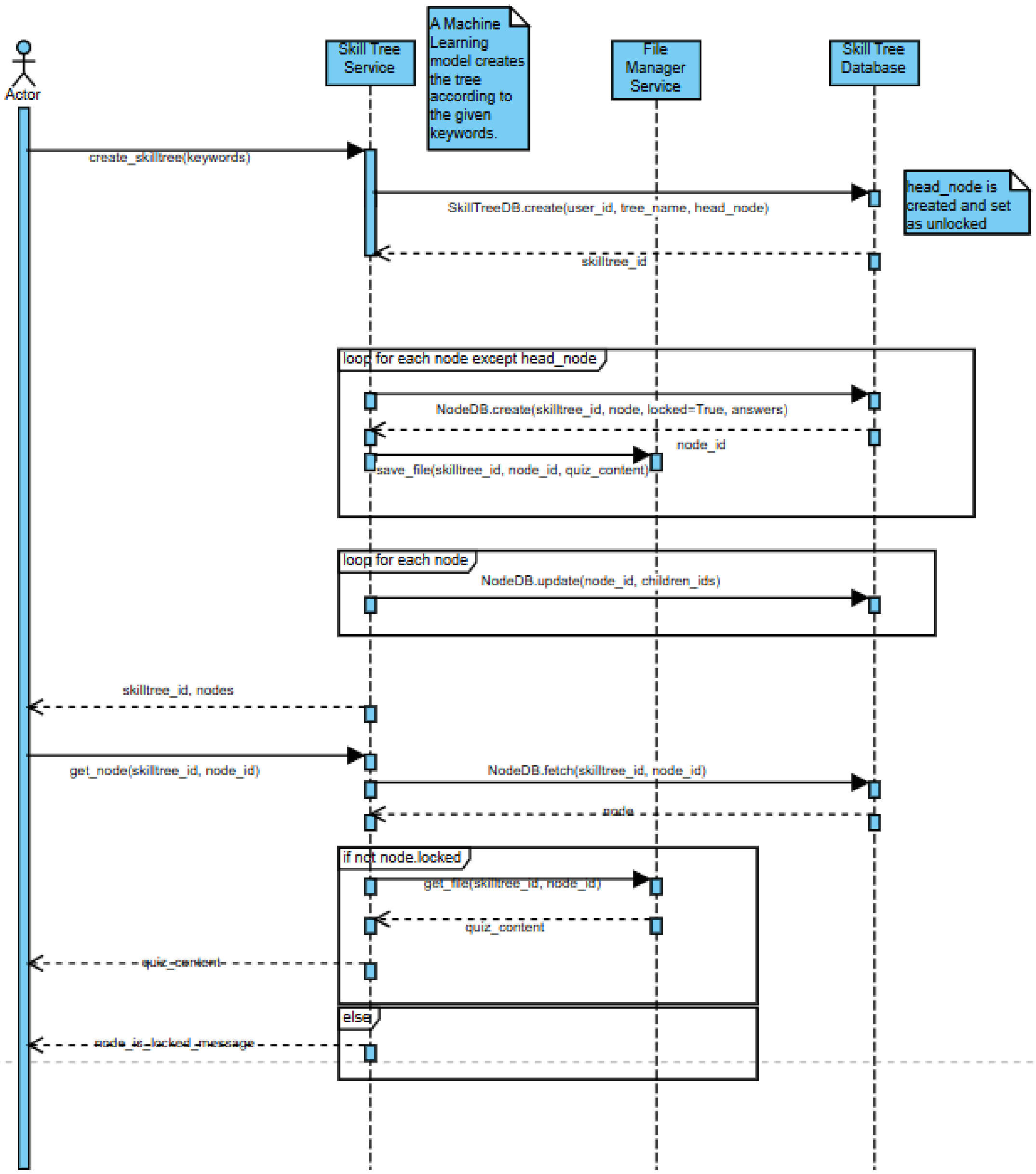
This diagram depicts how the user can interact with a chat in "chat mode". In "chat mode" the user can upload files or chat with the LLM, there is no slide involved in this process. After the user clicks at a chat that belongs to a particular individual study, they can send a message. If they provide a file, that file is uploaded to the system and the file content is sent to the LLM alongside with the chat history and the message from the user. The LLM returns a response accordingly. If a file is not provided, the same parameters are provided except the file content. Finally the chat history is saved and the response is returned to the user


Chat is assumed to be in "slide mode". Additionally, the user is assumed to have called get_chat already.
User
Chat Service
File Manager Service
LLM Service
Chat Database
update_chat_slides(chat_id, slides)
save_file(slides)
success
SlidesDB.create(chat_id, slide_metadata)
new_slide
ChatDB.update(chat_id, last_open_slide=new_slide)
updated_chat
get_slide(chat_id, slide_id, page)
SlidesDB.fetch(chat_id, slide_id)
slide
ChatDB.update(chat_id, last_opened_slide=slide_id)
SlidesDB.update(slide_id, last_opened_page=page)
get_file(slide.slide_history)
history
get_file(slide_id, page)
slide_content
if not history
send_message(prompt, slide_content, model_version)
response
save_file(slide_history)
slide_content, slide_history
else
slide_content, history

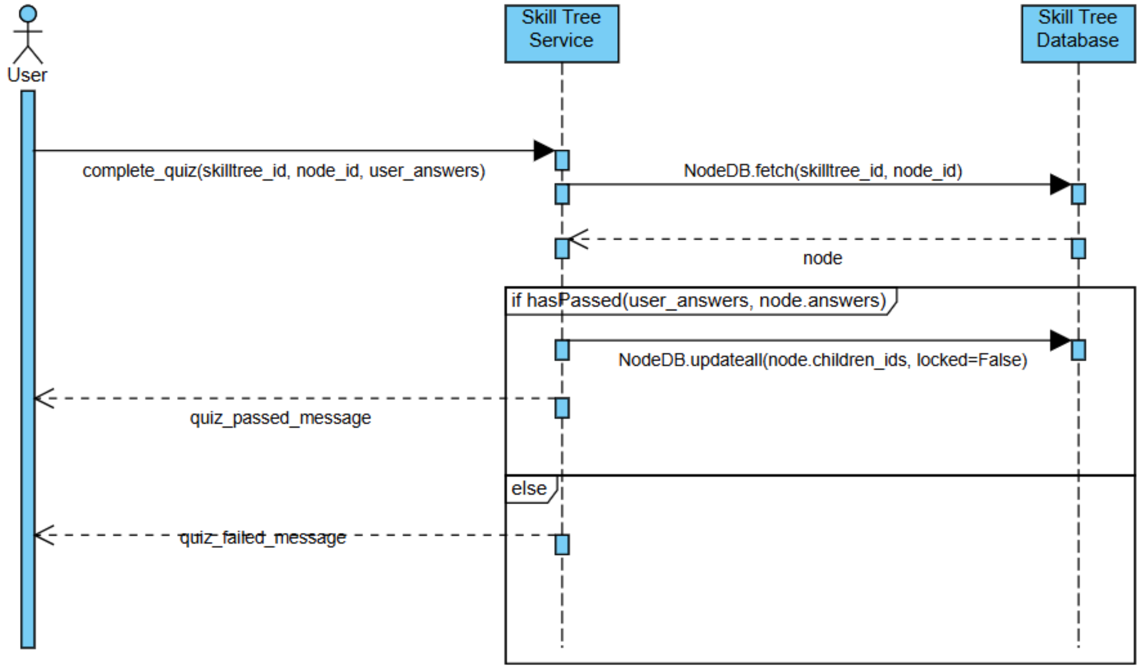
This diagram shows how a user can upload slides and receive their explanations. First, the user provides the slides and they are uploaded to the system and a database entry is created for that slide within the chat. When the user clicks a button inside the chat, a slide and its explanation appears. If there already is a saved explanation for the given slide page (if the user has already viewed that page), that explanation is returned. Else, the slide content is sent to the LLM with our special prompt that requires an explanation. Then the generated explanation is saved for that page and it is returned.


Actor
Chat Service
File Manager Service
LLM Service
Chat Database
create_flashcard(chat_id)
SlidesDB.fetchall(chat_id)
all_slides
loop for each slide
get_file(slide.history_path)
slide_history
send_message(flashcard_prompt, all_slide_histories, model_version)
flashcard
save_file(flashcard)
flashcard

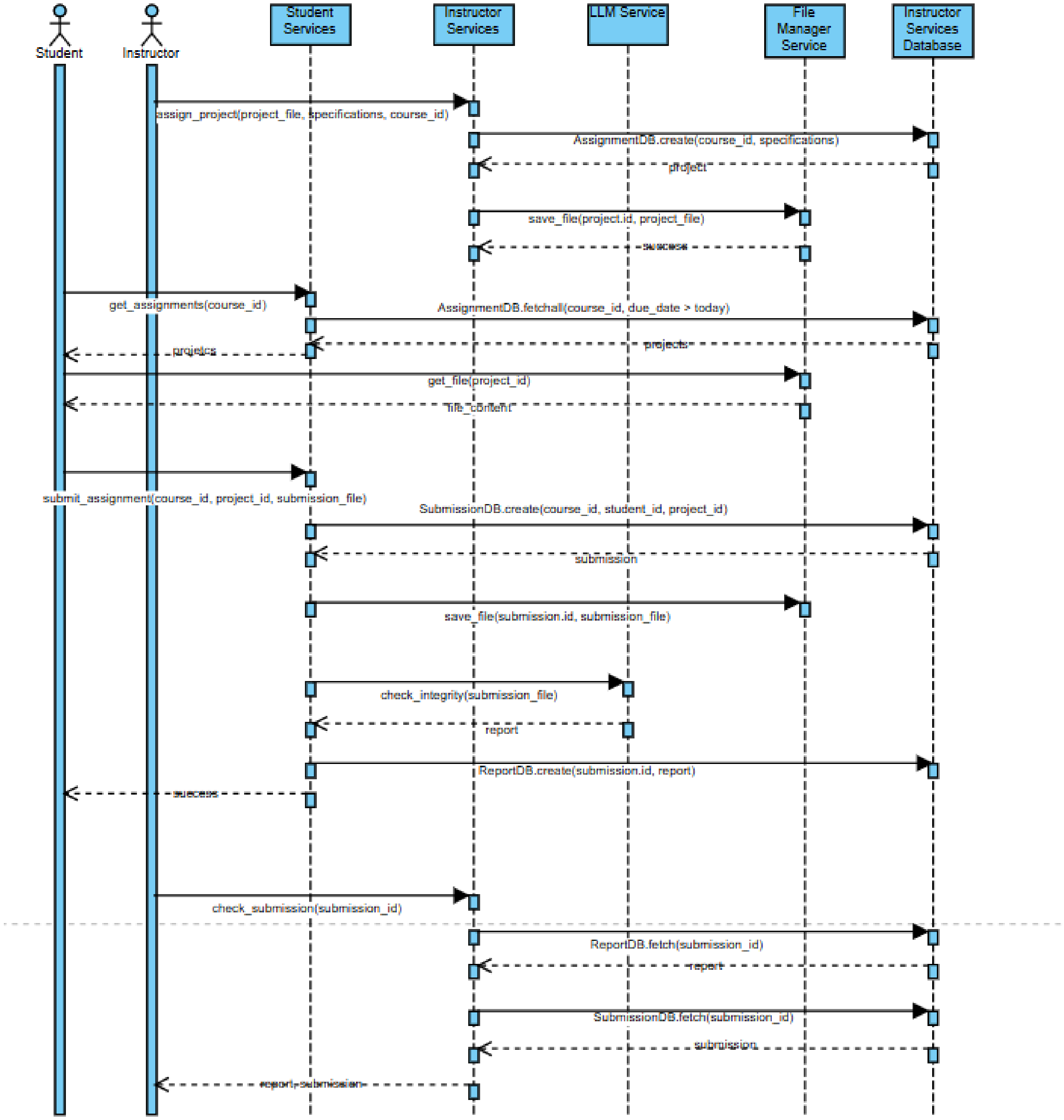
This diagram shows how a user can create a flashcard within a chat. When the user clicks the "create flashcards" button, the system loops over all of the slides and their histories (explanations) and sends them to the LLM alongside with our special prompt requiring flashcards in a predefined format. Finally the response from the LLM is saved and the flashcards are returned to the user. If there are no slides in the chat, they are generated using only the chat history. Quizzes are created the exact same way, except for the special "quiz prompt" provided to the LLM.


Actor
Skill Tree Service
A Machine Learning model creates the tree according to the given keywords.
File Manager Service
Skill Tree Database
create_skilltree(keywords)
SkillTreeDB.create(user_id, tree_name, head_node)
head_node is created and set as unlocked
skilltree_id
loop for each node except head_node
NodeDB.create(skilltree_id, node, locked=True, answers)
node_id
save_file(skilltree_id, node_id, quiz_content)
loop for each node
NodeDB.update(node_id, children_ids)
skilltree_id, nodes
get_node(skilltree_id, node_id)
NodeDB.fetch(skilltree_id, node_id)
node
if not node.locked
get_file(skilltree_id, node_id)
quiz_content
quiz_content
else
node_is_locked_message

This diagram shows how a user can create a skill tree (a tree whose nodes are quizzes and requires the user to solve a node to access the children nodes). The user provides keywords (e.g. support vector machines, neural networks etc.), and a skill tree is created according to those keywords. Then the tree is saved to the database. The root of the tree is unlocked. When the user clicks on a node, the system checks if that node is unlocked, if the quiz is returned, else the user gets a fail message.

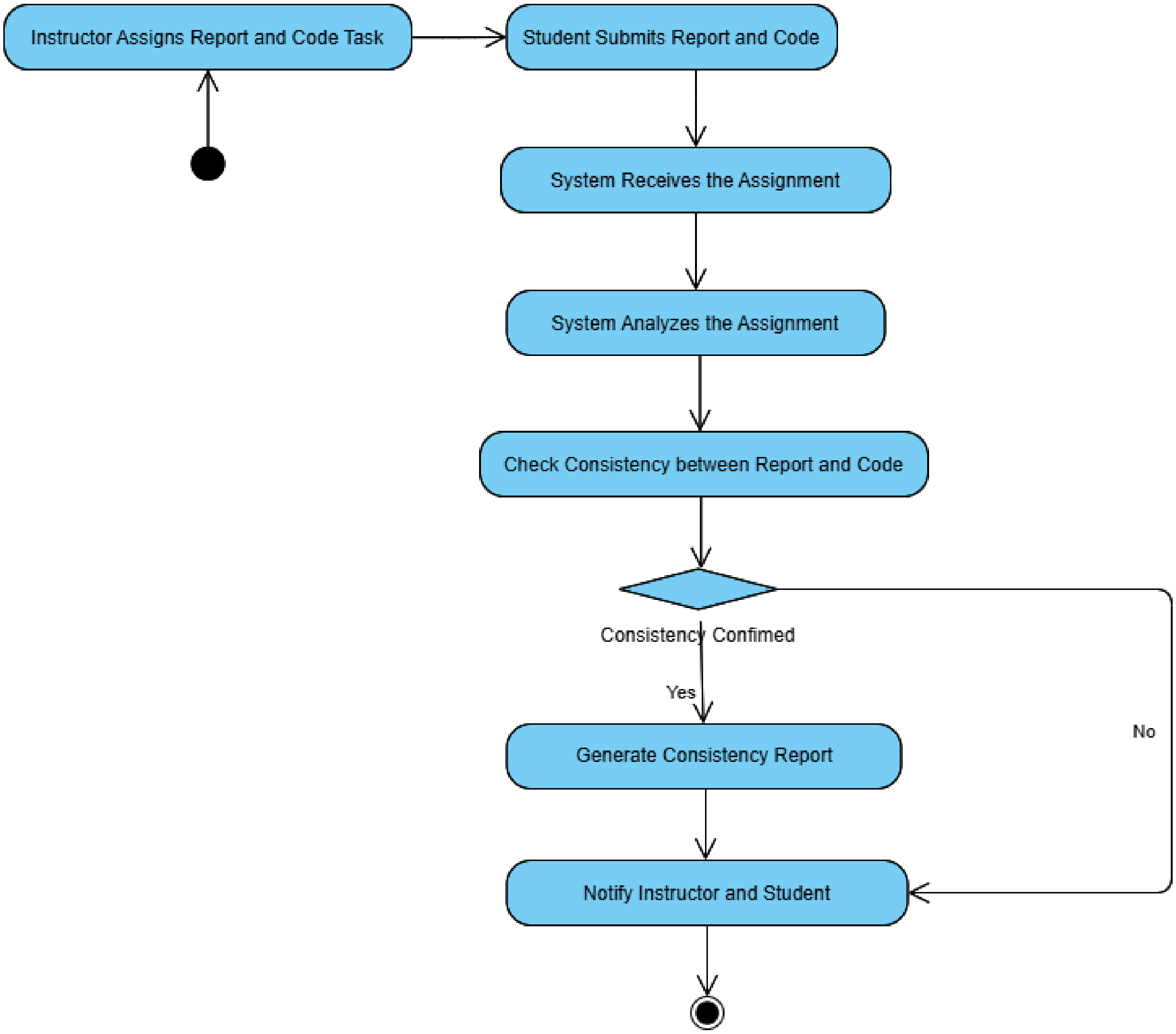

User
Skill Tree Service
Skill Tree Database
complete_quiz(skilltree_id, node_id, user_answers)
NodeDB.fetch(skilltree_id, node_id)
node
if hasPassed(user_answers, node.answers)
NodeDB.updateall(node.children_ids, locked=False)
quiz_passed_message
else
quiz_failed_message

This diagram shows how a user can unlock new nodes in a skill tree. When the user completes the quiz of a node, the system checks if the user has passed the quiz. If they pass, the system updates the children nodes of the completed node as "unlocked". If they fail, the system returns a "quiz failed" message and the children nodes remain locked.


Student
Instructor
Student Services
Instructor Services
LLM Service
File Manager Service
Instructor Services Database
assign_project(project_file, specifications, course_id)
AssignmentDB.create(course_id, specifications)
project
save_file(project.id, project_file)
success
get_assignments(course_id)
AssignmentDB.fetchall(course_id, due_date > today)
projects
projetcs
get_file(project_id)
file_content
submit_assignment(course_id, project_id, submission_file)
SubmissionDB.create(course_id, student_id, project_id)
submission
save_file(submission.id, submission_file)
check_integrity(submission_file)
report
ReportDB.create(submission.id, report)
success
check_submission(submission_id)
ReportDB.fetch(submission_id)
report
SubmissionDB.fetch(submission_id)
submission
report, submission

This diagram shows how an instructor can assign a coding project and how the students can submit that assignment. The instructor provides a project specification file and additional files which are uploaded to the system and the assignment which belongs to a course is saved to a database with that course's id. The students then can see that assignment by inside their course. They can download the necessary files for that assignment from the system. After completion, they can submit their assignment. During submission, the uploaded code is sent to the LLM service with a prompt requiring it to check for potential plagiarism or cheating. Then an integrity report is created and saved with the submission. Then the instructor can check the submission and the academic integrity report affiliated with that submission.

3.5.4.2. Activity Diagrams

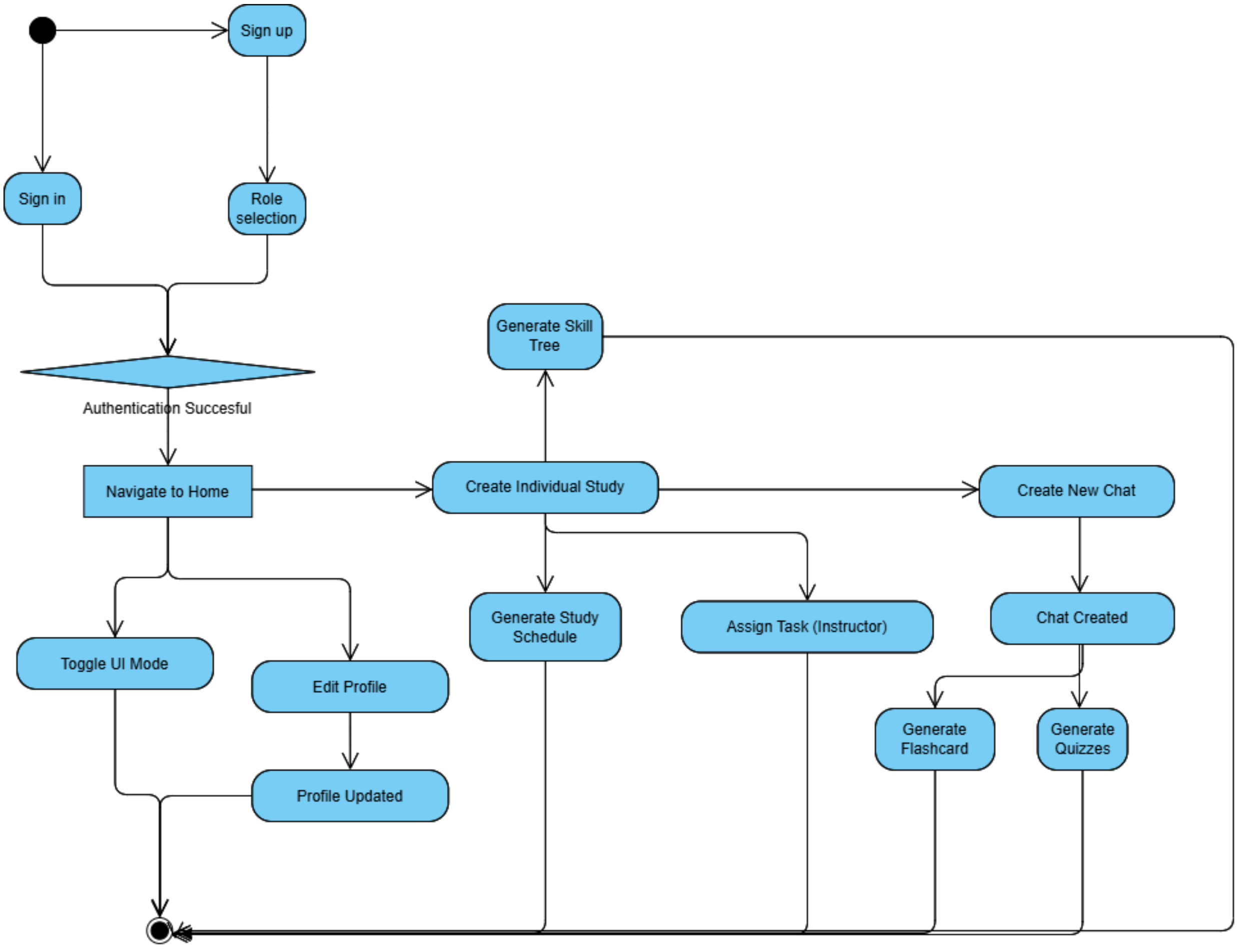

Instructor Assigns Report and Code Task
Student Submits Report and Code
System Receives the Assignment
System Analyzes the Assignment
Check Consistency between Report and Code
Consistency Confirmed
Yes
No
Generate Consistency Report
Notify Instructor and Student

The activity diagram represents a structured process for verifying the consistency between a student's report and their submitted code, ensuring that both elements align. The process begins when an instructor assigns a task requiring the student to submit a written report and the corresponding code. This task could be a project or an assignment where the theoretical explanation (report) must match the practical implementation (code). Once the task is assigned, the student completes it and submits the report and code to the system.

Upon receiving the submission, the system automatically processes the assignment by acknowledging receipt of the materials. It then performs an analysis to prepare for a detailed consistency check. This step involves scanning the content of the report and the code to identify key elements, such as objectives, methodology, and outcomes, to ensure they match the code's functionality and logic. The system subsequently runs a consistency check to determine if the report accurately describes the code's implementation.

At this point, a decision is made based on the results of consistency. If the system confirms that the report and code are consistent, it proceeds to generate a consistency report summarizing the alignment between the two components. This report ensures transparency and provides evidence that the task has been completed. However, if the consistency check fails—meaning the report does not accurately reflect the code—the system bypasses the report generation and moves to the notification step.

Finally, the system notifies the instructor and the student of the outcome. Whether the consistency is confirmed, the feedback helps students understand their performance and informs the instructor about potential issues. This automated process streamlines the evaluation, reducing the time and effort required for manual consistency checks and ensuring accuracy in verifying the relationship between a student's theoretical explanation and their practical coding output.

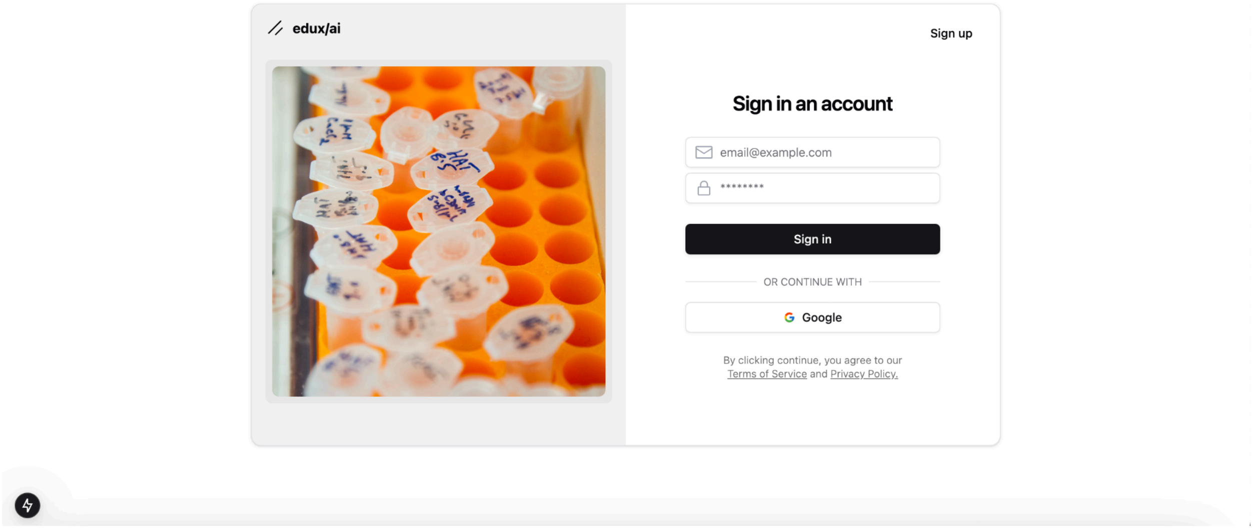


This activity diagram represents a user authentication and role-based functionality system, likely for an educational platform. The process begins with a user either signing up or signing in. If the user selects the Sign Up path, they proceed to Role Selection, where they choose a role, such as a student or an instructor. Once sign-in or sign-up is completed, the system verifies whether authentication is successful. The user is granted access to the home interface if authentication is validated. This marks the end of the authentication phase and the start of navigation within the platform.

After successful authentication, the user is directed to the home page, where they can perform several tasks. They can Toggle UI Mode to change the interface settings or Edit Profile, followed by a confirmation step, Profile Updated, to reflect any changes. In parallel, users can Create an Individual Study, which branches out to actions such as Generate Study Schedule for personalized planning or, if the user is an instructor, Assign Task (Instructor). The platform supports personalized study workflows and task assignments, indicating flexibility for different user roles.

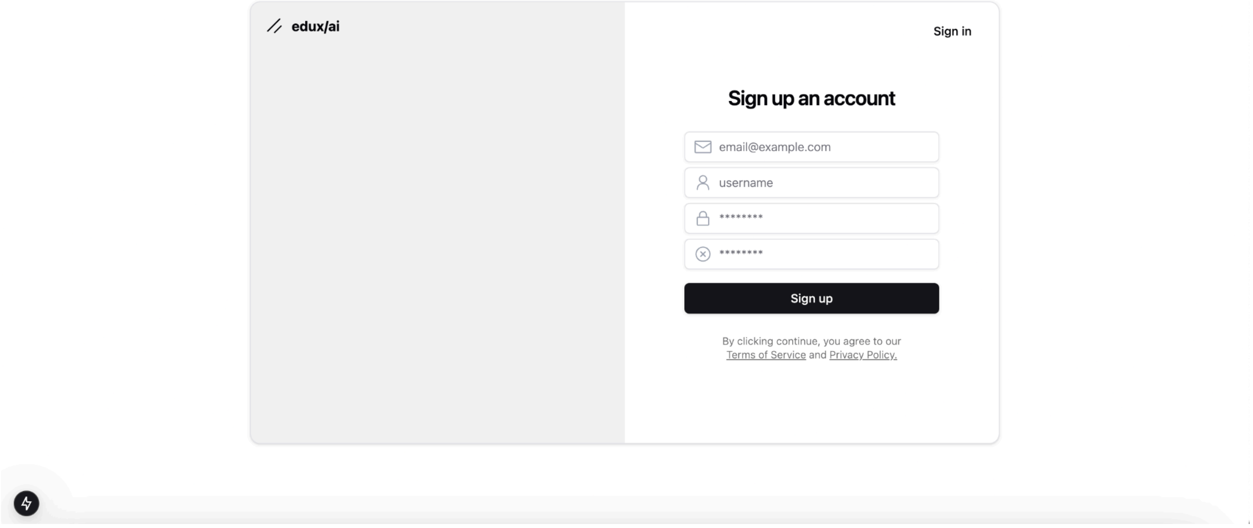
Additionally, the diagram includes a feature for initiating communication through the Create New Chat option, which can be accessed directly from the home interface or after creating individual study plans. This allows users to collaborate, discuss, or seek real-time clarification. The activity diagram's flow returns to the final state, depicted as a black circle, indicating the completion of the user's session. The diagram demonstrates an interconnected set of functionalities tailored to support profile management, study creation, and role-specific activities, providing a streamlined and user-friendly experience for both students and instructors.

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



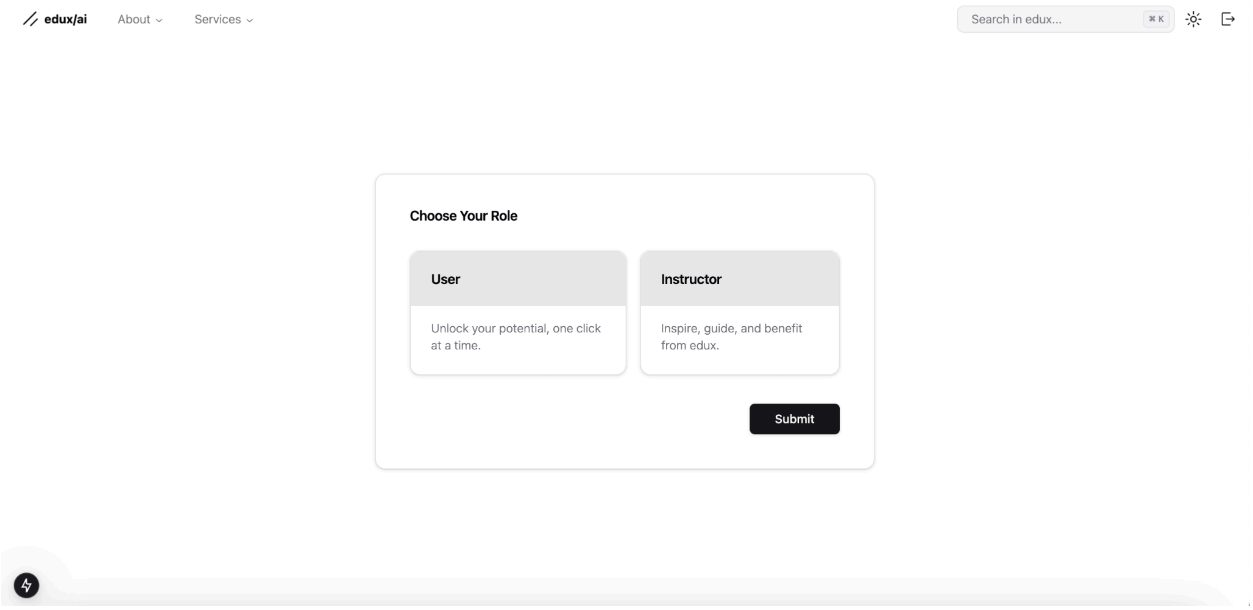
By clicking continue, you agree to our Terms of Service and Privacy Policy.

From the login screen, by typing in their emails and passwords users can log in to the system. Also, users can use their Google accounts to login as well. Sign Up option navigates users to another page to create a new account.



By clicking continue, you agree to our Terms of Service and Privacy Policy.

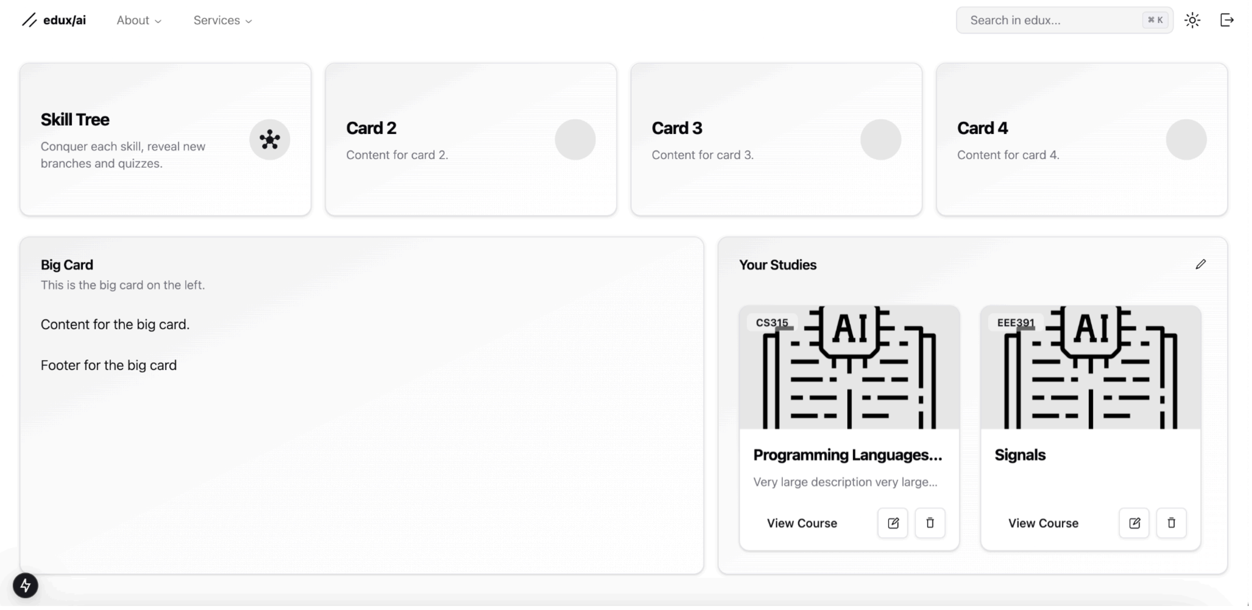
On the sign-up screen, users can create a new account by entering their email, username, and password (with a confirmation field). A "Sign up" button allows them to complete the process. Returning users can choose the "Sign in" option at the top right to access their accounts.



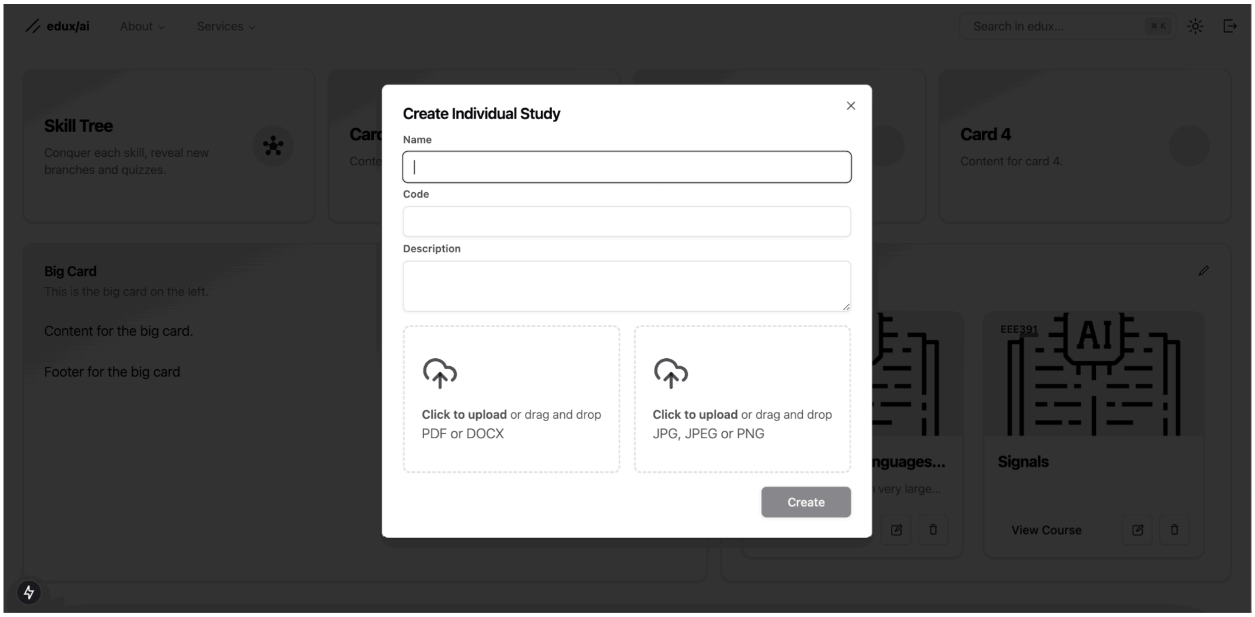
Unlock your potential, one click at a time.

Inspire, guide, and benefit from edux.

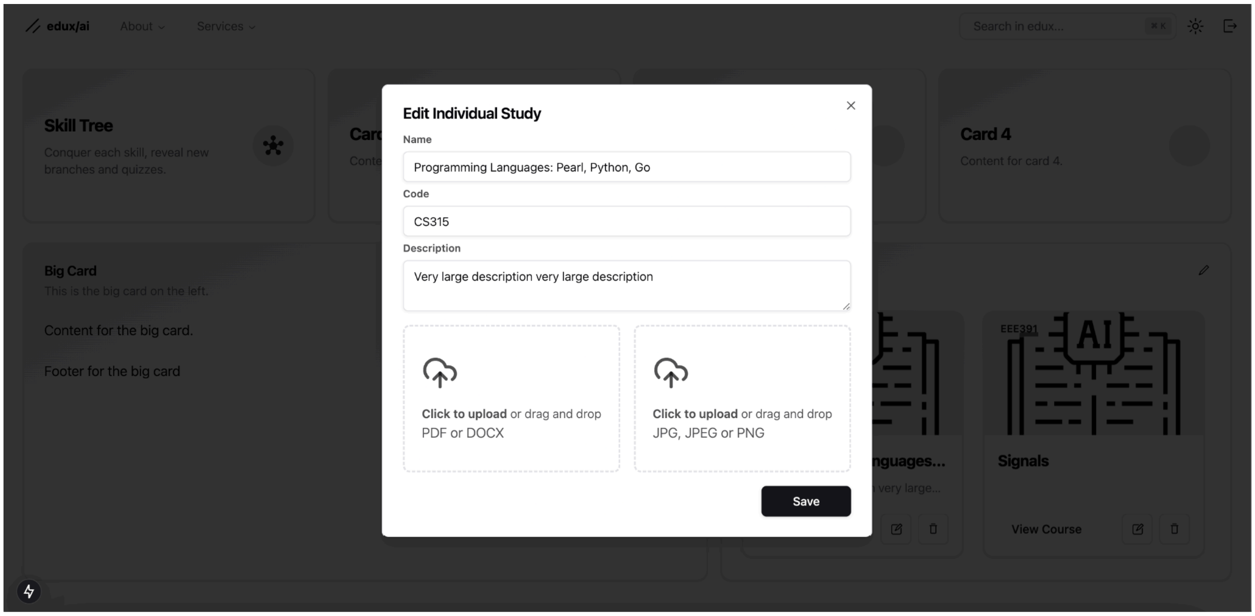
On the role selection screen, users can pick one of two roles: "User" for regular use or "Instructor" to help and guide others. There is a "Submit" button to confirm the choice. At the top, there are options for "About," "Services," and a search bar.



The dashboard has different sections. At the top, there are cards like "Skill Tree" for exploring skills and quizzes. The main page will include information like user statistics, or success scores from quizzes. By clicking at the courses at the right bottom panel, users can navigate to the courses' home pages.



From the pop-up form titled "Create Individual Study," users can enter the name, code, and description of the study. They can also upload PDF or DOCX files and image files like JPG, JPEG, or PNG. The "Create" button at the bottom completes the process.



Edit Individual Study

Name

Programming Languages: Pearl, Python, Go

Code

CS315

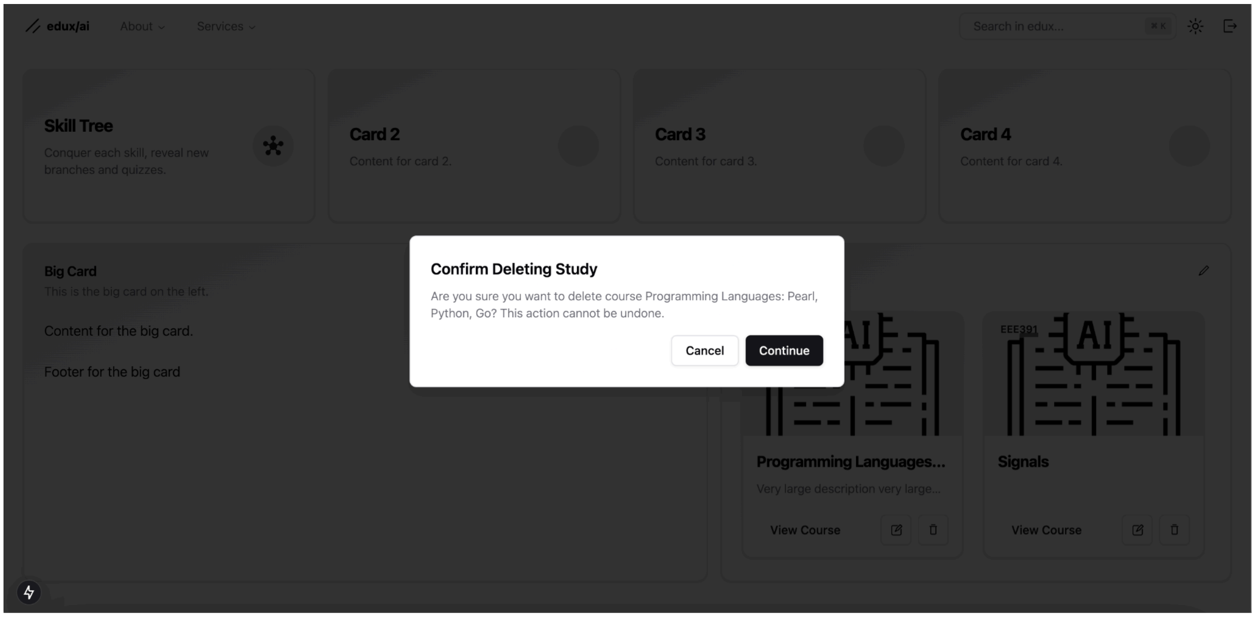
Description

Very large description very large description

Click to upload or drag and drop
PDF or DOCX

Click to upload or drag and drop
JPG, JPEG or PNG

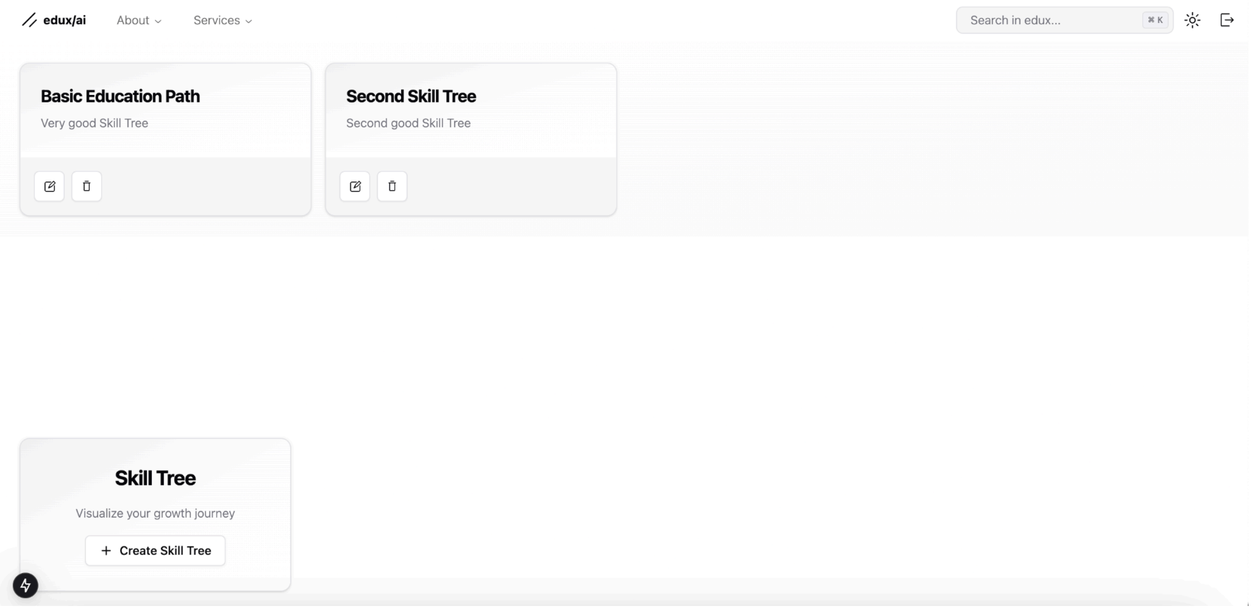
From the pop-up form for editing an individual study, users edit course name, code and description, can upload or replace PDF/DOCX files (syllabus file) and image files like JPG, JPEG, or PNG (course icon). The "Save" button at the bottom allows users to confirm their changes.



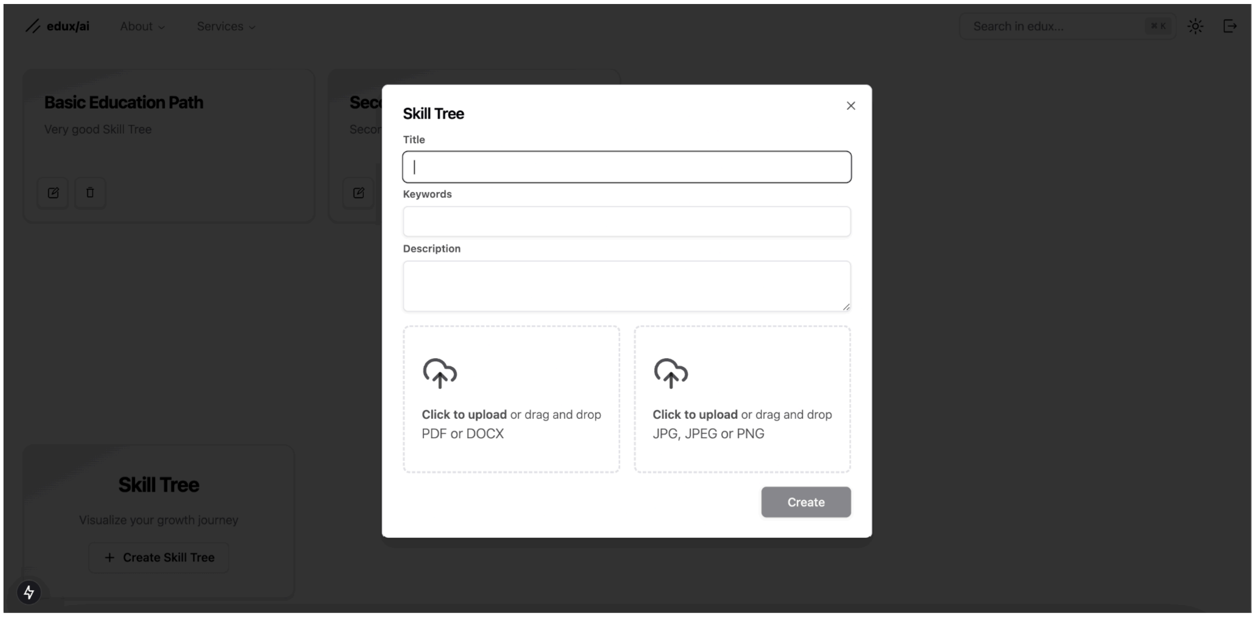
Confirm Deleting Study

Are you sure you want to delete course Programming Languages: Pearl, Python, Go? This action cannot be undone.

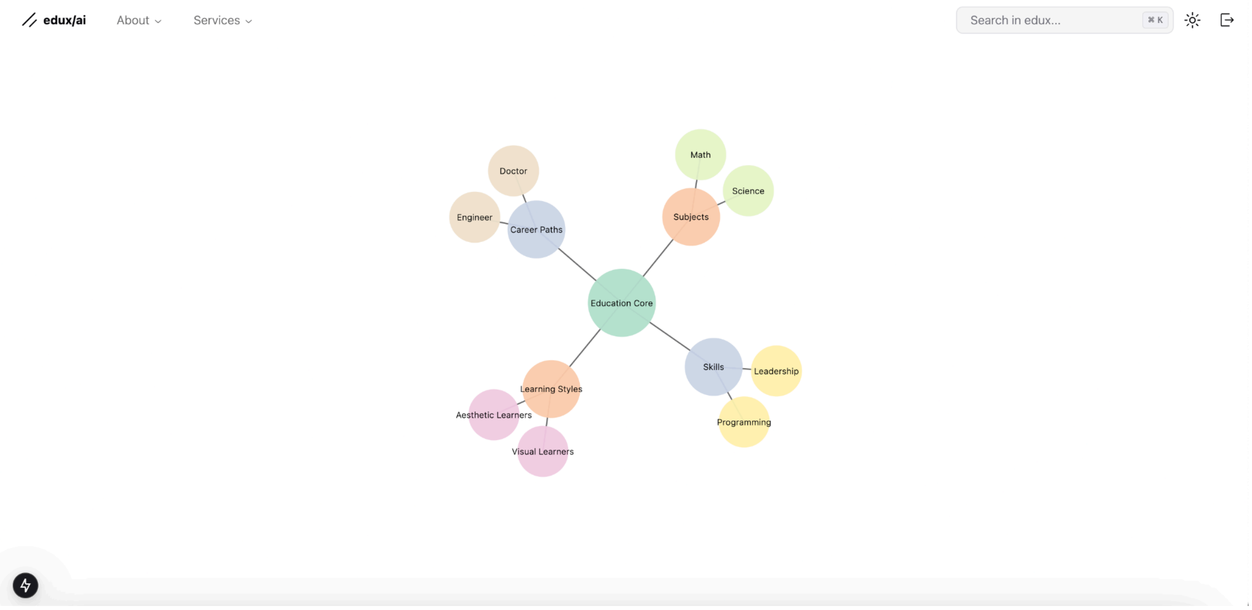
This image shows a confirmation pop-up for deleting a study. If continue is clicked, course is deleted forever.



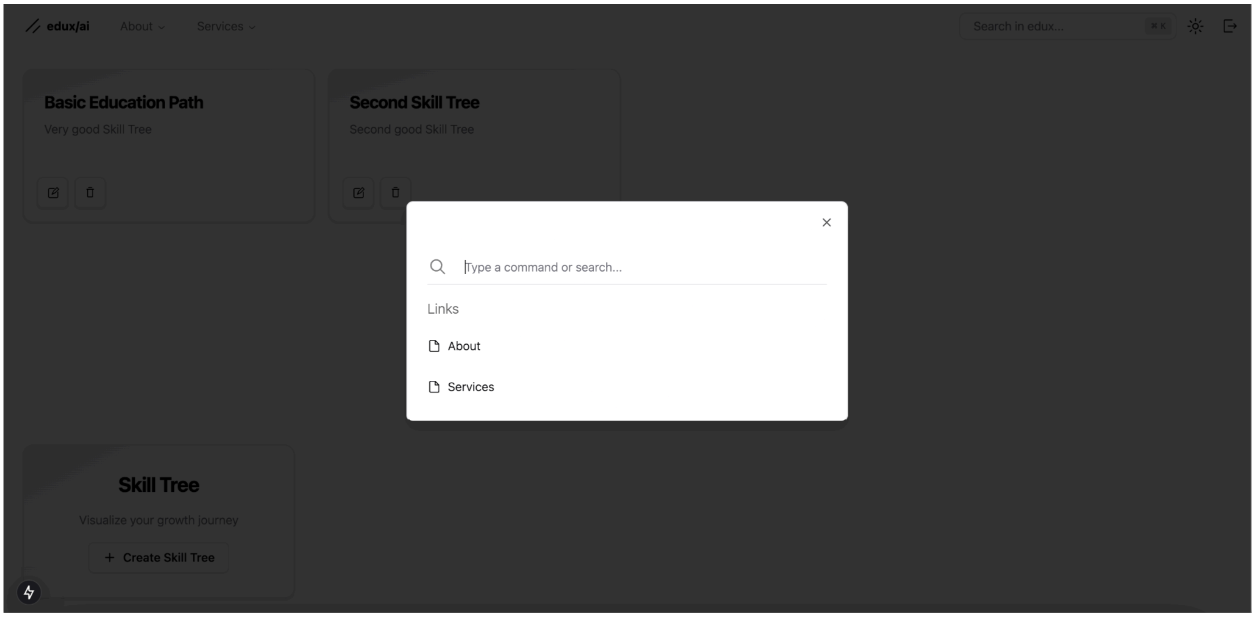
From the skill tree management interface, users can see two cards at the top: "Basic Education Path" and "Second Skill Tree," each with descriptions and icons to edit or delete them. At the bottom, there is a "Skill Tree" section with a button to create a new skill tree.



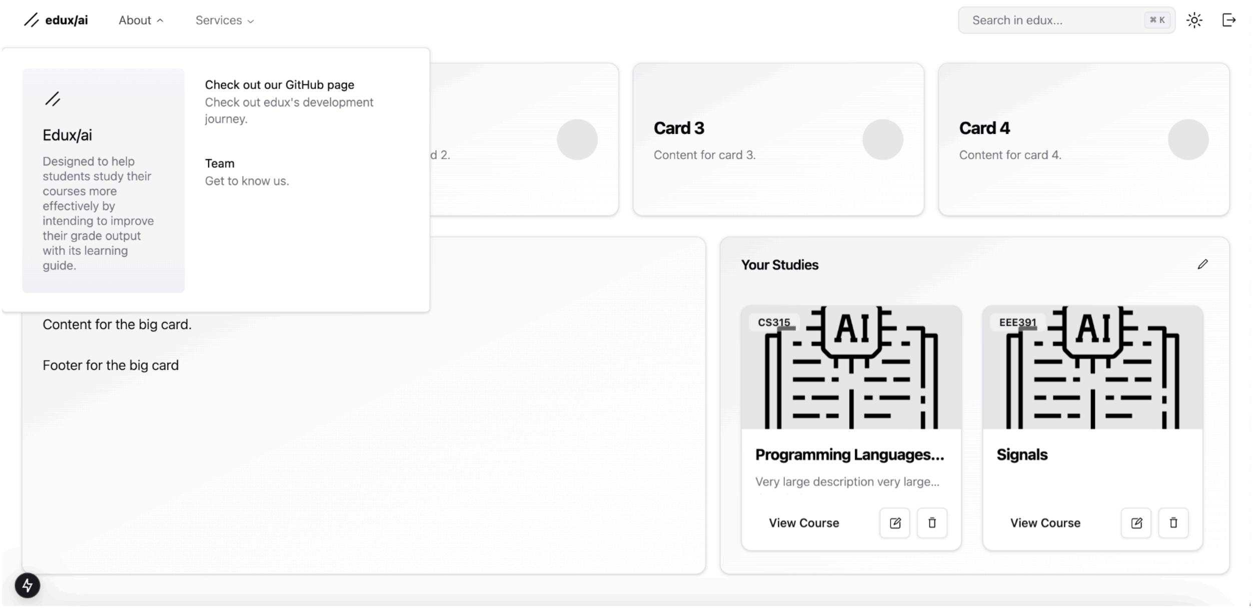
From the pop-up form for creating a new Skill Tree, users can enter a title, keywords, and description. The "Create" button at the bottom lets users complete the process.



This is the skill-tree UI. Each skill (nodes) are connected to each other as learner progresses in their studies.



From the search bar, users can type in any search queries and relevant results will be shown.



edux/ai About Services

Search in edux... ⌘ K

Edux/ai

Designed to help students study their courses more effectively by intending to improve their grade output with its learning guide.

Check out our GitHub page

Check out edux's development journey.

Team

Get to know us.

d 2.

Card 3

Content for card 3.

Card 4

Content for card 4.

Content for the big card.

Footer for the big card

Your Studies

CS315

Programming Languages...

Very large description very large...

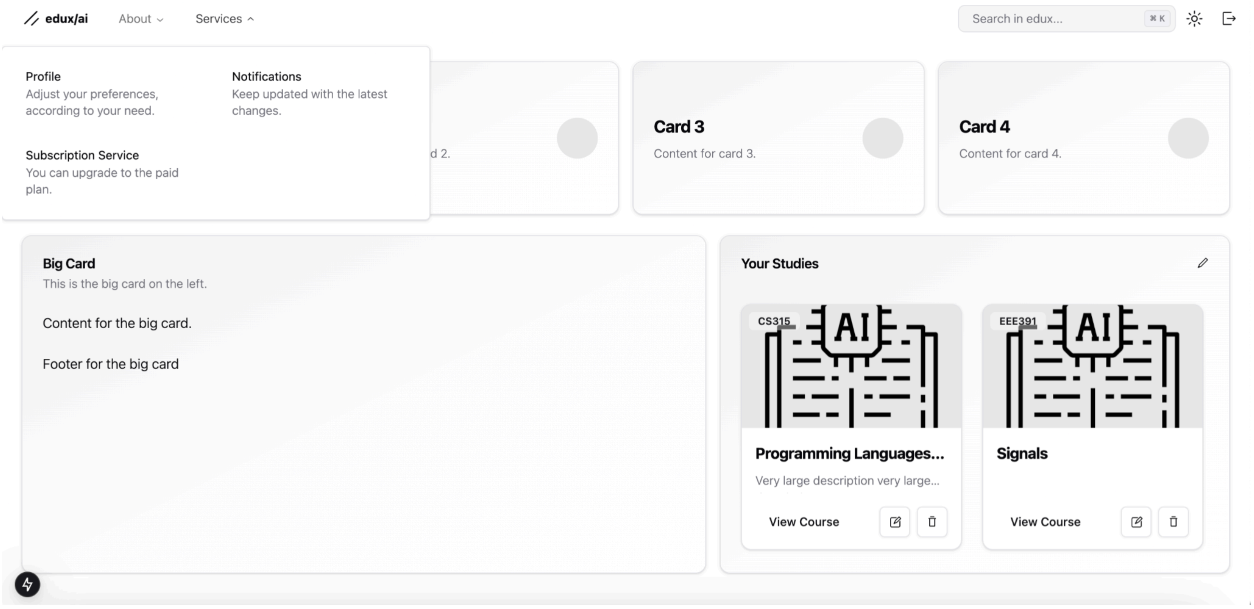
View Course

EEE391

Signals

View Course

The global navigation bar makes it easier for users to access certain buttons, such as About, Services, etc.



edux/ai About Services

Search in edux... ⌘ K

Profile

Adjust your preferences, according to your need.

Notifications

Keep updated with the latest changes.

Subscription Service

You can upgrade to the paid plan.

d 2.

Card 3

Content for card 3.

Card 4

Content for card 4.

Big Card

This is the big card on the left.

Content for the big card.

Footer for the big card

Your Studies

CS315

Programming Languages...

Very large description very large...

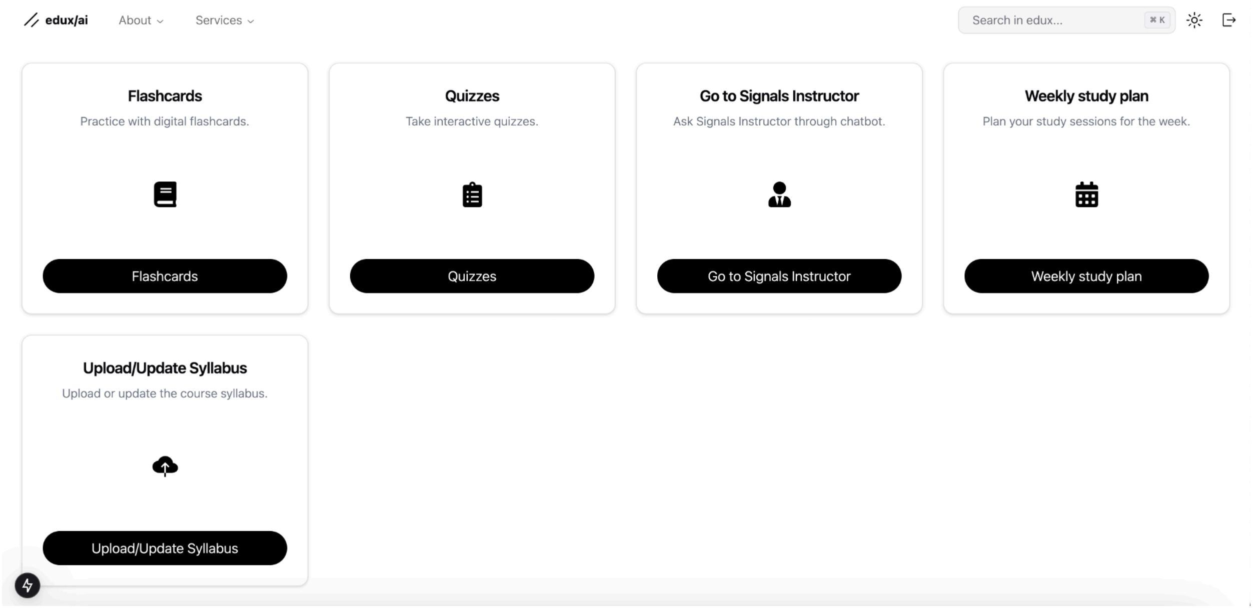
View Course

EEE391

Signals

View Course

From the global navigation bar, users can access directly to certain services such as Profile, Notifications, and their Subscription plans.



Flashcards

Practice with digital flashcards.

Quizzes

Take interactive quizzes.

Go to Signals Instructor

Ask Signals Instructor through chatbot.

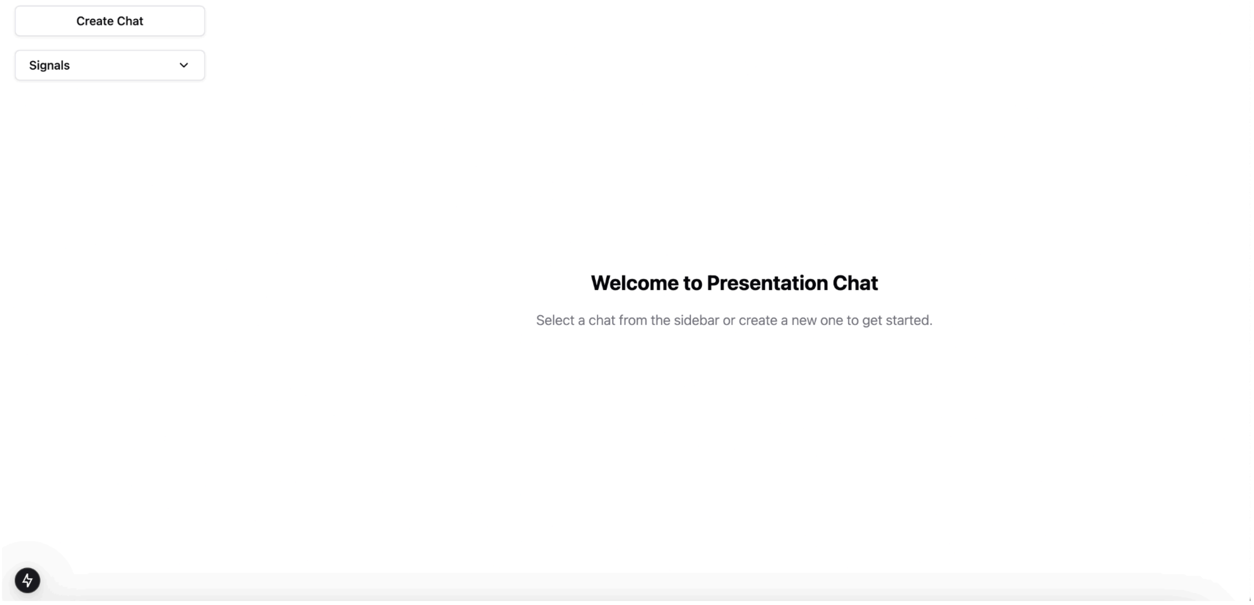
Weekly study plan

Plan your study sessions for the week.

Upload/Update Syllabus

Upload or update the course syllabus.

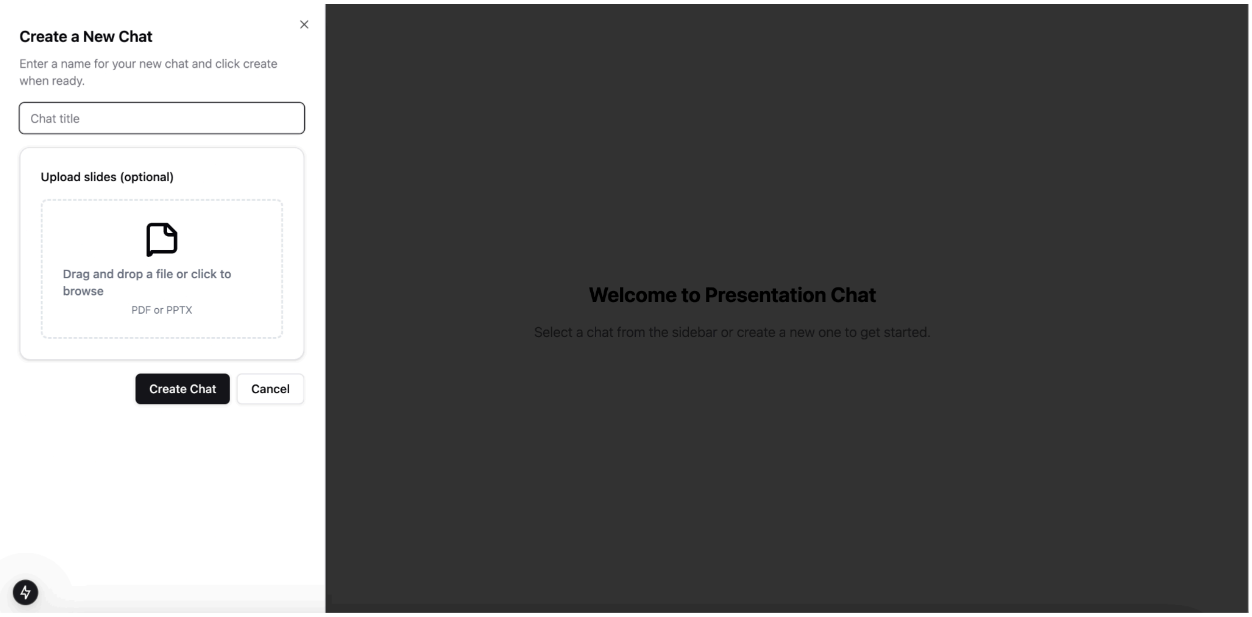
From the course main page, users can access different features through cards. The first card, "Flashcards," allows practice with digital flashcards. The second card, "Quizzes," provides interactive quizzes. The third card, "Go to Instructor," lets users talk to an AI Instructor using a chatbot. The fourth card, "Weekly Study Plan," helps users plan their weekly study sessions. The fifth card, "Upload/Update Syllabus," allows users to upload or update the syllabus for a study or course. Each card has a button for quick access to its features.



Welcome to Presentation Chat

Select a chat from the sidebar or create a new one to get started.

This image shows a chat interface for a "Presentation Chat" feature. There is a "Create Chat" button at the top left, allowing users to start a new chat. Below it, there is a dropdown menu labeled "Signals" for selecting different chat categories or channels. The main area on the right displays a welcome message: "Welcome to Presentation Chat" with instructions to select a chat from the sidebar or create a new one to get started.



Create a New Chat

Enter a name for your new chat and click create when ready.

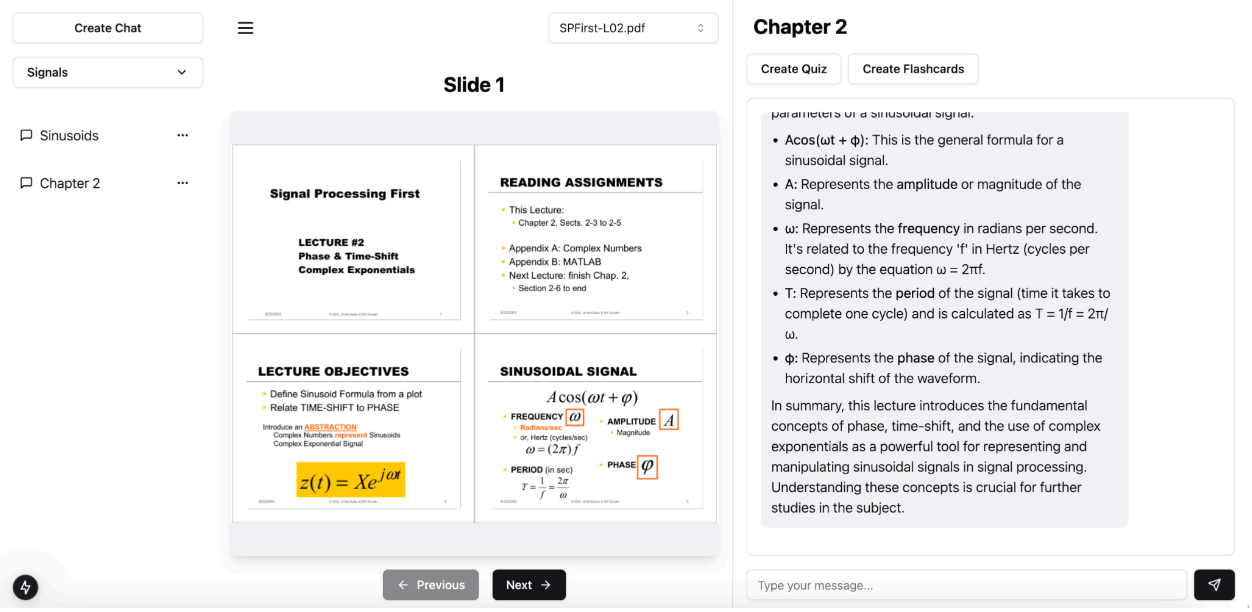
Drag and drop a file or click to browse

PDF or PPTX

Welcome to Presentation Chat

Select a chat from the sidebar or create a new one to get started.

From the pop-up window for creating a new chat in the "Presentation Chat" interface, users can enter a title for the chat in the "Chat title" field on the left panel. Below, they can upload slides in PDF or PPTX format by dragging and dropping the file or browsing manually. At the bottom, there are two buttons: "Create Chat" to confirm or "Cancel" to exit.



Slide 1

Signal Processing First

LECTURE #2
Phase & Time-Shift
Complex Exponentials

READING ASSIGNMENTS

- This Lecture:
  - Chapter 2, Sects. 2-3 to 2-5
- Appendix A: Complex Numbers
- Appendix B: MATLAB
- Next Lecture: finish Chap. 2,
  - Section 2-6 to end

LECTURE OBJECTIVES

- Define Sinusoid Formula from a plot
- Relate TIME-SHIFT to PHASE

Introduce an ABSTRACTION:
Complex Numbers represent Sinusoids
Complex Exponential Signal

$z(t) = Xe^{j\omega t}$

SINUSOIDAL SIGNAL

$A\cos(\omega t + \varphi)$

- FREQUENCY $\boxed{\omega}$
  - Radians/sec
  - or, Hertz (cycles/sec)

$\omega = (2\pi) f$

- PERIOD (in sec)

$T = \frac{1}{f} = \frac{2\pi}{\omega}$

- AMPLITUDE $\boxed{A}$
  - Magnitude
- PHASE $\boxed{\varphi}$

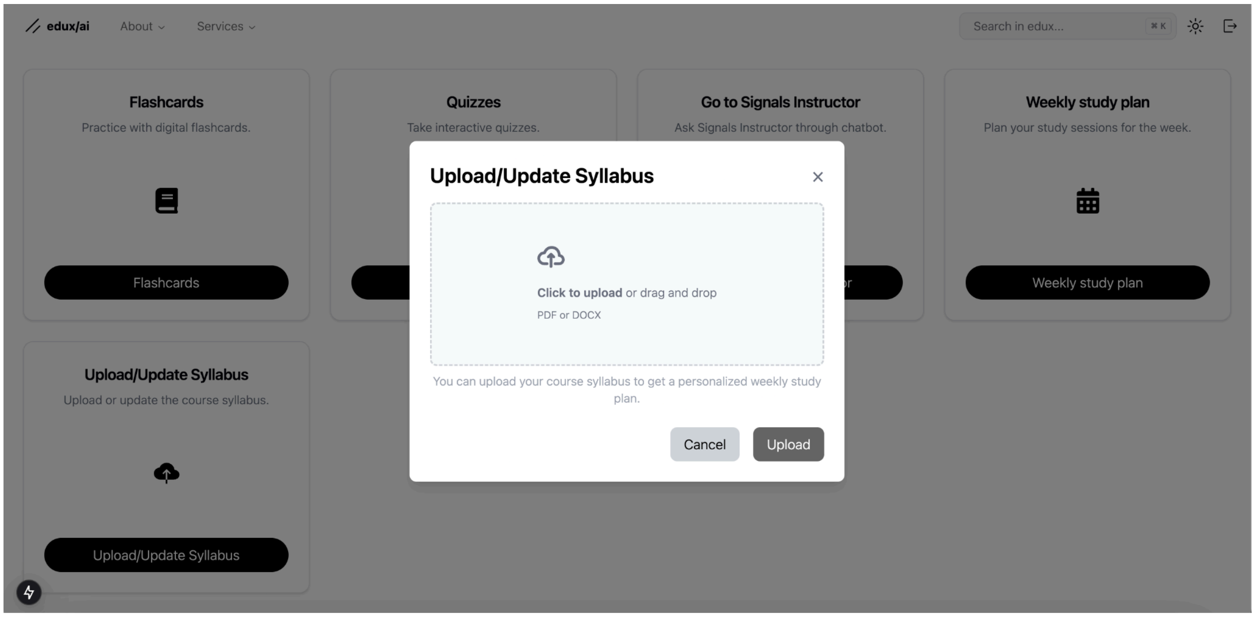
Chapter 2

parameters of a sinusoidal signal.

- **Acos(ωt + ϕ):** This is the general formula for a sinusoidal signal.
- **A:** Represents the **amplitude** or magnitude of the signal.
- **ω:** Represents the **frequency** in radians per second. It's related to the frequency 'f' in Hertz (cycles per second) by the equation ω = 2πf.
- **T:** Represents the **period** of the signal (time it takes to complete one cycle) and is calculated as T = 1/f = 2π/ω.
- **ϕ:** Represents the **phase** of the signal, indicating the horizontal shift of the waveform.

In summary, this lecture introduces the fundamental concepts of phase, time-shift, and the use of complex exponentials as a powerful tool for representing and manipulating sinusoidal signals in signal processing. Understanding these concepts is crucial for further studies in the subject.

From the educational chat interface, the center displays a slide with multiple content blocks. The left sidebar allows users to switch between different sections. At the top, there is a dropdown to select other chats and an uploaded file. On the right, users can view a description with options to create quizzes and flashcards. At the bottom, there is a message input field with a send button. If a user sends a message, a response from the AI instructor will be generated and shown back to the user.



Practice with digital flashcards.

Take interactive quizzes.

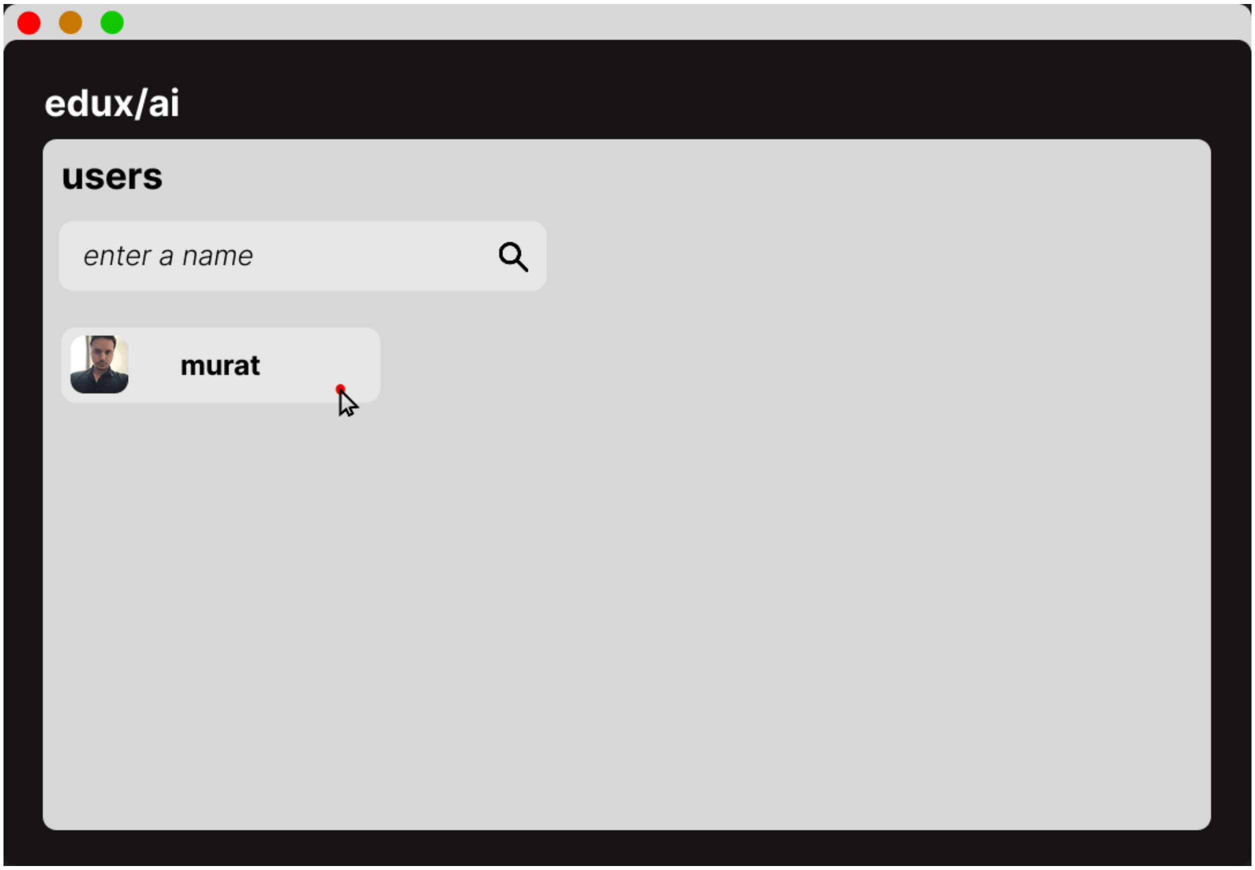
Ask Signals Instructor through chatbot.

Plan your study sessions for the week.

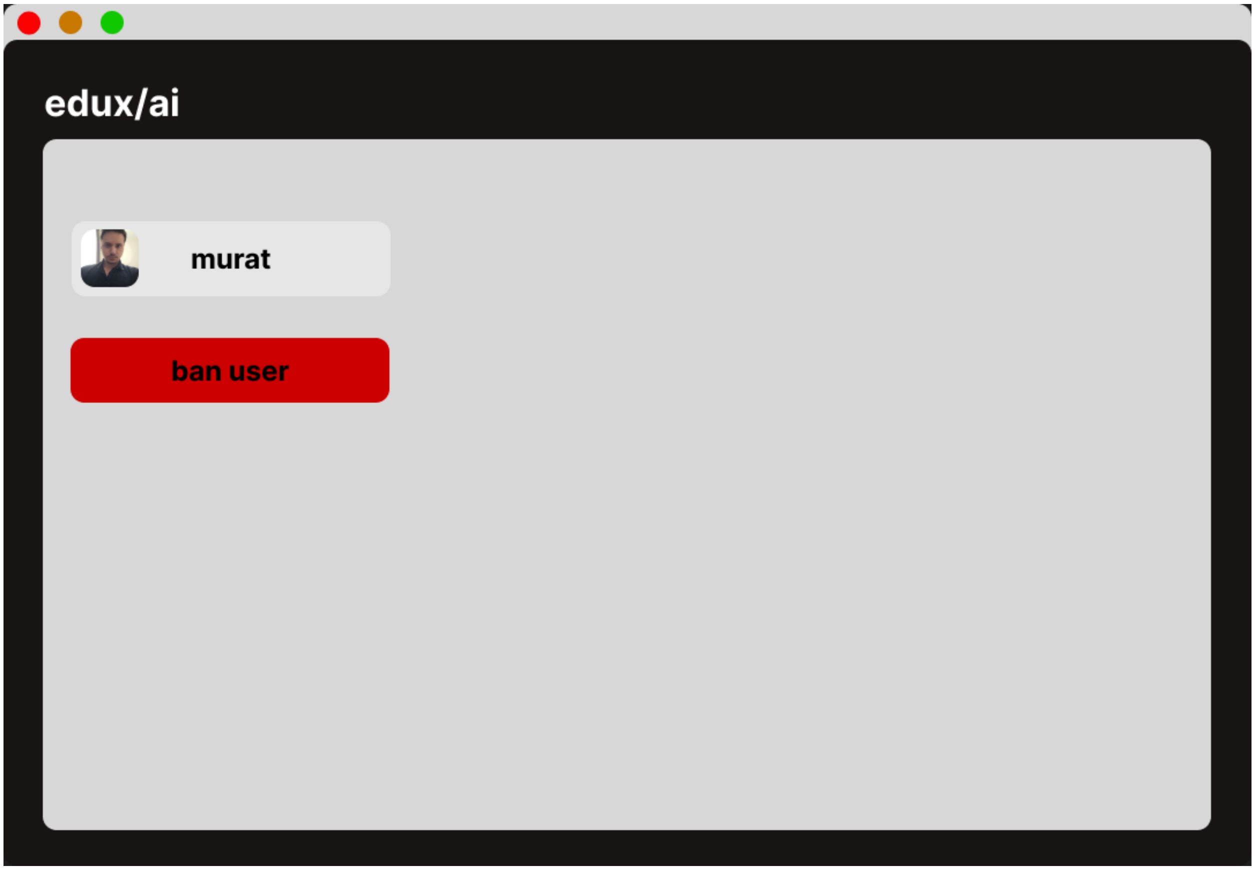
Upload or update the course syllabus.

You can upload your course syllabus to get a personalized weekly study plan.

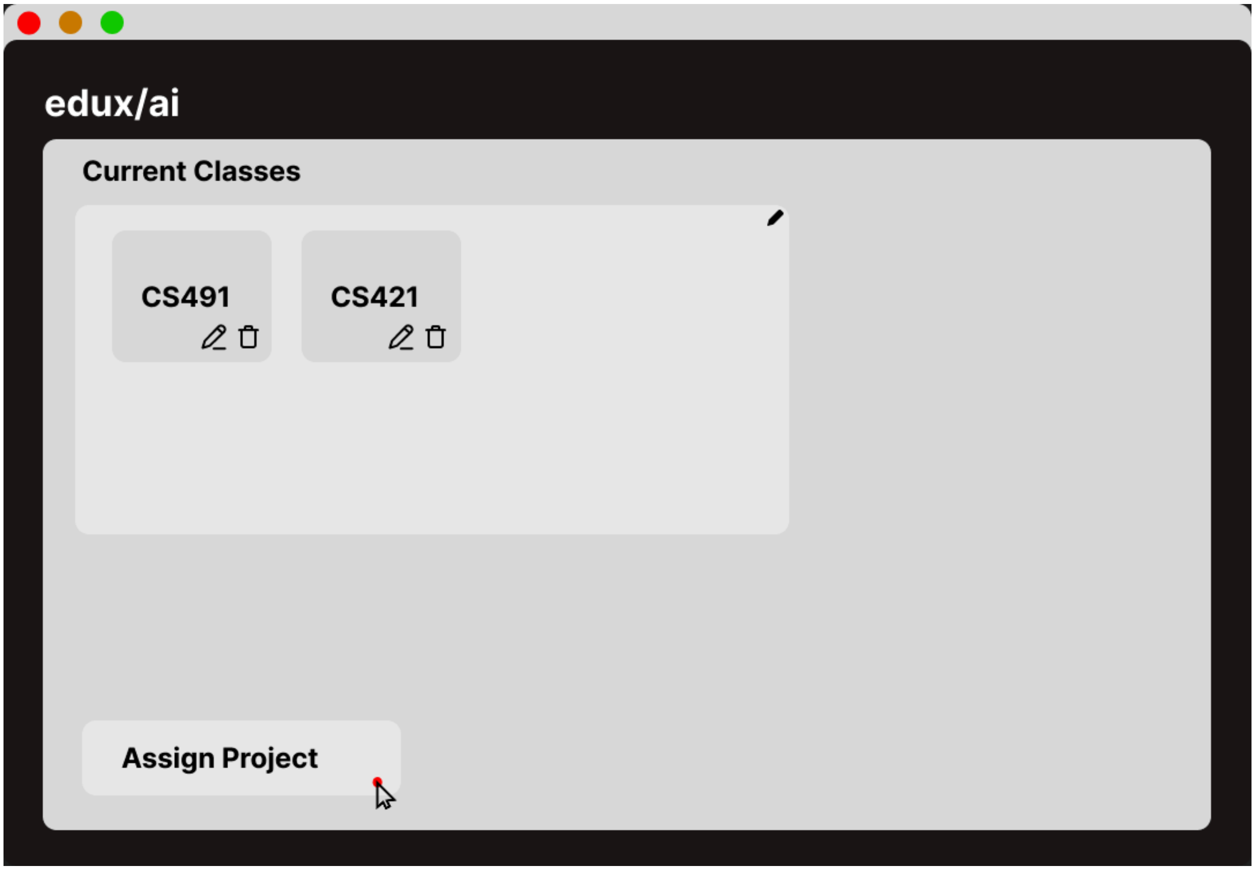
This image shows a pop-up window titled "Upload/Update Syllabus." The window contains an upload section where users can drag and drop a file or click to browse for PDFs or DOCX files. Below the upload area, there's a note explaining that uploading an individual study syllabus helps create a personalized weekly study plan. At the bottom of the pop-up, there are two buttons: "Cancel" to exit and "Upload" to confirm the action.



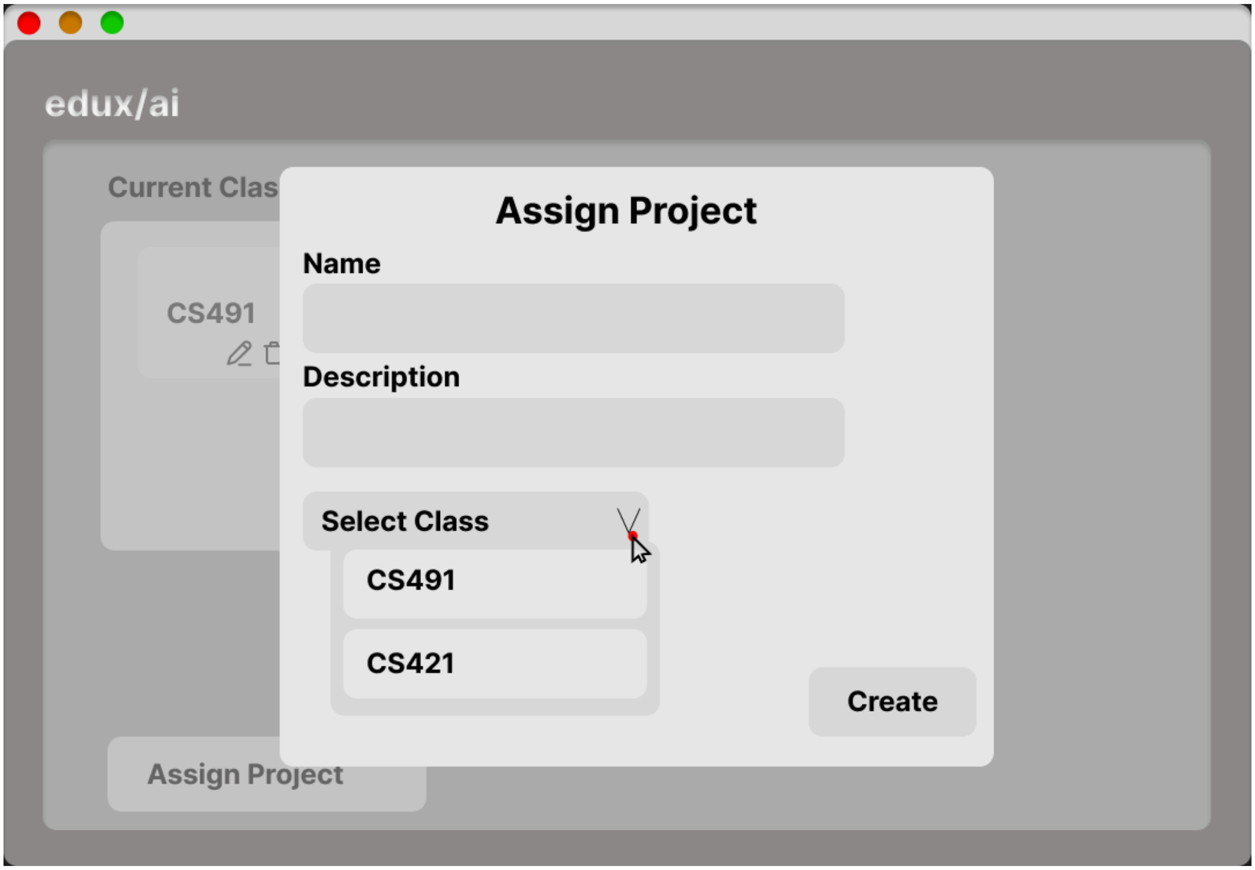
This image displays a user search interface for moderators. At the top, there's a search bar labeled "enter a name". Below the search bar a list of users is shown for moderators for easy access.



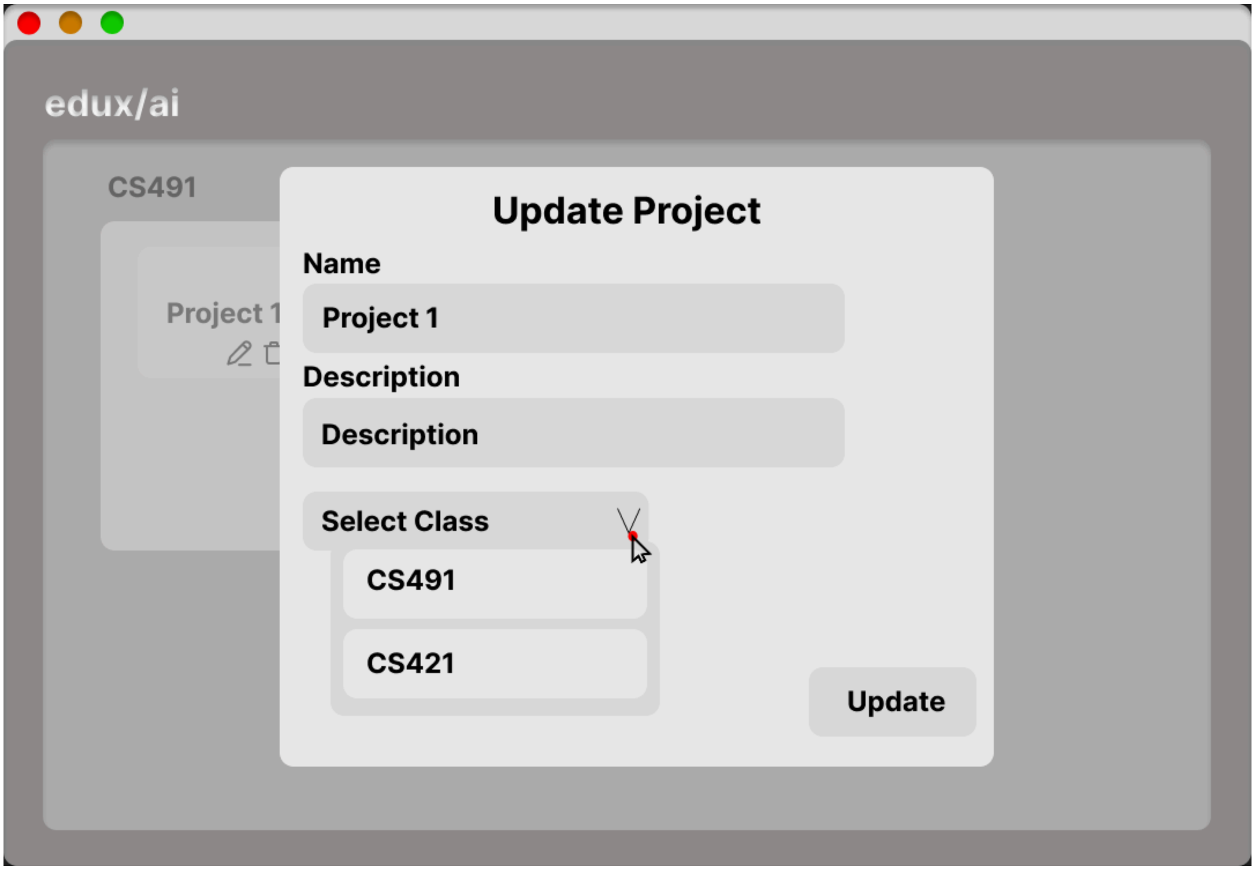
When a user is clicked, an option to ban the user will be shown to the moderator. Clicking and approving the button will ban the user.



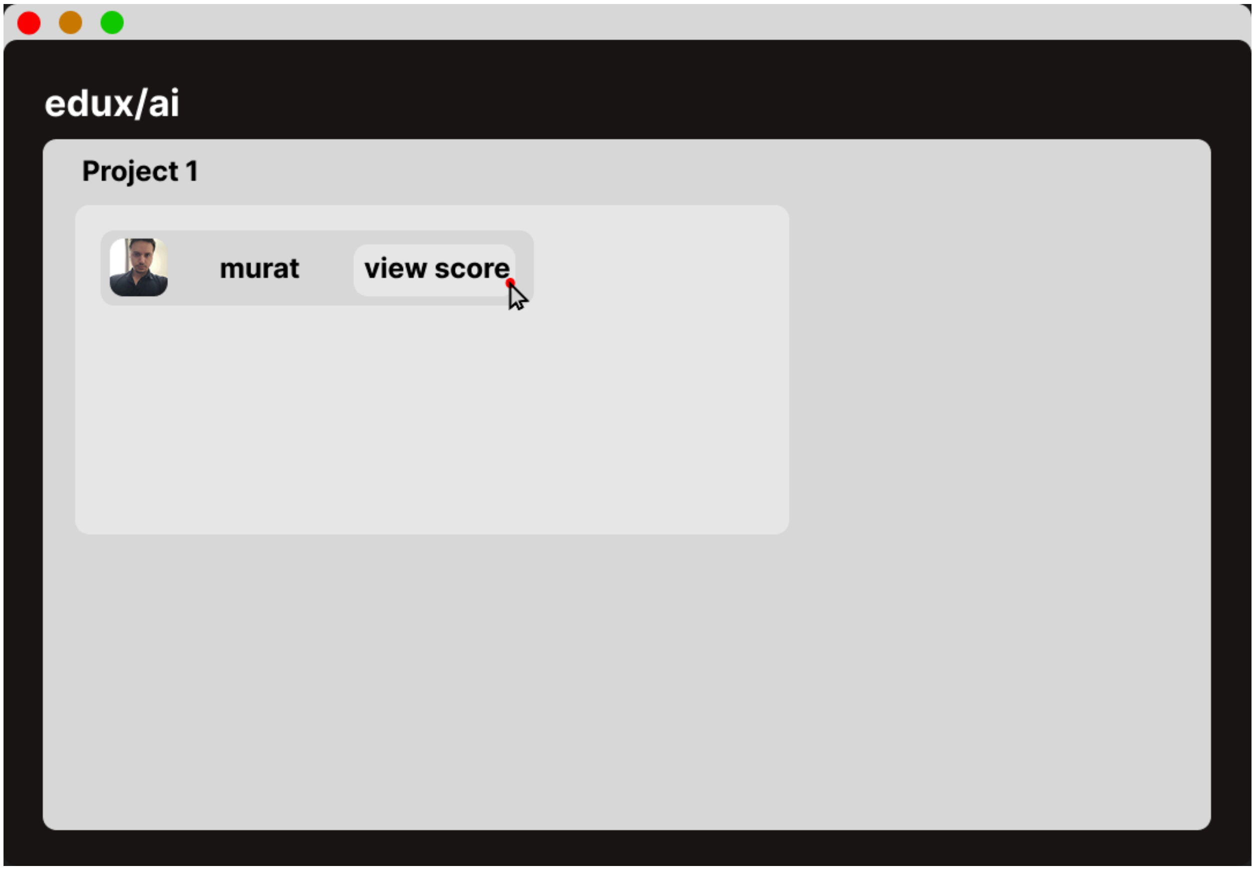
This image shows a class management interface for instructors. The section titled "Current Classes" displays classes the instructor has created. Each class has icons for editing and deleting. At the bottom, there is a button labeled "Assign Project" which allows the instructor to create and assign a new project.



From the "Assign Project" pop-up window, instructors can enter the project name and description. There is a dropdown menu labeled "Select Class" with options like CS491 and CS421, and the cursor is selecting one of them. At the bottom right, the "Create" button allows confirming the assignment.



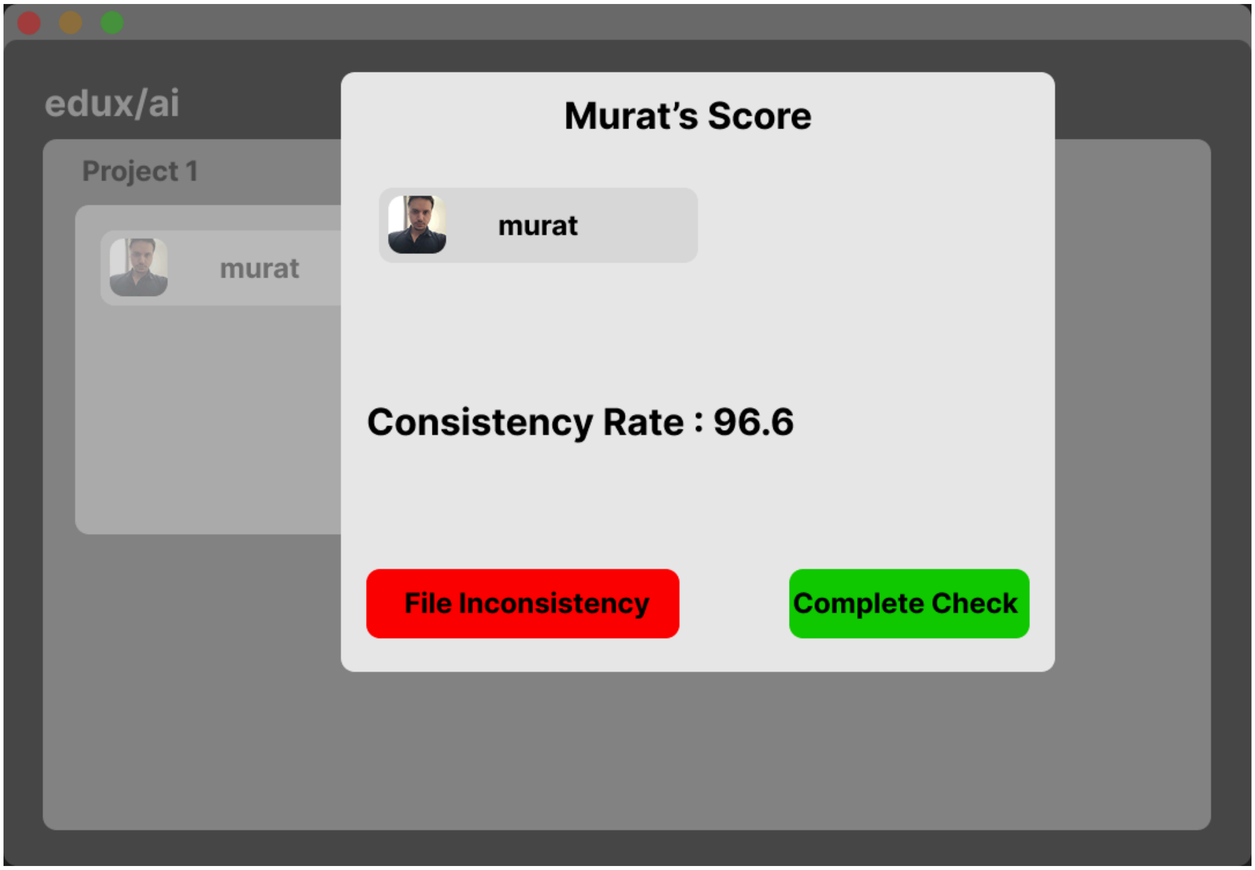
From the "Update Project" pop-up window, users can edit the project details. The "Name" field is pre-filled with "Project 1," and the "Description" field already contains "Description." There is a dropdown menu labeled "Select Class" with options like CS491 and CS421, and the cursor is selecting one of them. At the bottom right, the "Update" button lets users confirm the changes.



edux/ai

Project 1

Instructors can view the details of a project and check individual scores by clicking the "view score" button next to each user's name.



Murat’s Score

murat

Consistency Rate : 96.6

This image shows a score report interface displaying a user's score. The Consistency Rate is displayed as 96.6. Below the score, there are two buttons: a red "File Inconsistency" button and a green "Complete Check" button.

4. Other Analysis Elements

4.1. Consideration of Various Factors in Engineering Design

4.1.1. Public Health Considerations

Edux strives to elevate education through personalized learning tools, ultimately fostering better mental and intellectual well-being. While Edux is not directly involved in health practices, it contributes to mental health by reducing cognitive stress through adaptive study schedules and interactive tools. These features help learners manage their time efficiently and stay focused on their goals.

4.1.2. Public Safety Considerations

Edux prioritizes public safety by implementing robust security measures to protect sensitive user data. The platform will and should ensure that personal information, such as login credentials and uploaded resources, remains encrypted and secure. Unauthorized access is prevented through multi-layered security protocols, while compliance with KVKK and GDPR and other data protection standards ensures that the platform respects user privacy. In addition, Edux prevents the misuse of uploaded educational materials by strictly prohibiting unauthorized sharing or publicizing of data, safeguarding both users and their intellectual property.

4.1.3. Public Welfare Considerations

While Edux does not directly impact public welfare, it indirectly contributes by enhancing access to personalized education, fostering intellectual growth, and preparing individuals for professional success. The platform's collaboration tools also improve the teaching process by bridging the gap between students and instructors, promoting effective communication and skill application. In the long term, Edux can help society by creating a better-educated workforce and reducing inequalities in access to quality learning resources.

4.1.4. Global Considerations

Edux is designed with global accessibility in mind. The platform defaults to English, ensuring broad reach across international audiences. Additionally, it is planned to adhere to international data protection laws such as GDPR, underscoring its commitment to global user safety. Future iterations of the platform aim to include language-specific adaptations and collaborations with global educational organizations to enhance its worldwide relevance.

4.1.5. Cultural Considerations

Edux acknowledges the significance of cultural diversity in education and ensures that its tools and resources are designed to support learners from diverse backgrounds. The platform strives to eliminate biases in its AI-generated content, promoting inclusivity and fairness in its study materials and recommendations. Furthermore, Edux is committed to bridging gaps in access to educational resources and can provide promotional discounts or free versions for underserved communities, thereby ensuring equal opportunities for learning.

4.1.6. Social Considerations

Edux handles sensitive user data, raising social and ethical concerns; therefore, encrypted and secure storage solutions are essential to protect privacy. The stored data in our application will not contain unique user identifiers such as names or identification numbers, ensuring an added layer of anonymity and protection. Furthermore, Edux will not use its existing user data to train machine-learning and LLM models and will not employ hidden data collection mechanisms within the platform. Moreover, the medical data needed for model training and constructing a knowledge base will be collected from reputable institutions with proper permissions, emphasizing transparency and adherence to ethical standards.

4.1.7. Environmental Considerations

Edux places a strong emphasis on environmental considerations, recognizing that any errors can not only jeopardize patient safety but also lead to wasteful consumption of resources. By providing users with comprehensive information promptly, Edux aims to reduce response times and resource usage. In alignment with sustainability goals, Edux will encourage medical centers to adopt more environmentally friendly practices in their operations.

4.1.8. Economic Considerations

Edux offers a wide range of functionalities to its users; however, access to certain features will depend on the user type. Therefore, the system will adopt a tiered monetization structure based on user categories. For example, students will have free access to the platform, while regular users or institutions will be required to pay a predetermined monthly fee. Edux will use a subscription business model, offering different subscription options related to the number of patient cases to be analyzed through the platform. Additionally, some advanced features, such as skill trees, might require an extra payment. The specific subscription amounts are not determined at this stage. Furthermore, varying regional needs and the impact of inflation rates complicate the establishment of a universal subscription model.

4.1.9. Constraints

4.1.9.1. Implementation Constraints

The implementation of Edux faces some challenges related to technology, scalability, content accuracy, and privacy. From a technology perspective, working with large amounts of data, like long textbooks or detailed syllabi, using LLMs requires a lot of computational power. This can slow down response times and affect the user experience, especially when the system has to process complicated tasks. Supporting different types of files, such as images, slides, PDFs, and audio, needs advanced tools to analyze the content accurately, and this can be more difficult with different types of files. Features like creating personalized study schedules or explaining the content in real-time depend on strong server and backend systems, which might struggle during peak usage.

Scalability is an important part of Edux's design because the platform needs to handle many users at the same time, especially in academic peak periods such as exam seasons. The system's setup must be able to handle this increase in users smoothly and allow new features, like voice recording, to be added without causing problems for the current services.

Ensuring the accuracy and relevance of content created by Edux, like flashcards, quizzes, and skill trees, is another big challenge. The platform needs to make sure these learning tools match the materials uploaded by users and keep their educational quality. This is especially important to maintain consistency between what students submit and what instructors assign, as any differences could negatively affect the learning experience.

Finally, user privacy and security are also very important when dealing with sensitive educational materials, performance or usage data, and interactions between instructors and students. Edux needs to follow strict data protection laws, like KVKK and GDPR, and use secure login systems to prevent unauthorized access. These constraints set the limits for how Edux can work, aiming to balance new ideas with what is realistic and possible.

4.1.9.2. Economic Constraints

The development and implementation of Edux are shaped by several economic constraints, including development costs, running expenses, earning strategies, managing resources, and getting educational institutions to adopt the platform. The initial development phase requires a lot of funding to build features like LLM-based tools, interactive learning aids, and personalized scheduling systems. A large part of this investment goes toward securing strong cloud infrastructure to support the platform. In addition, licensing or creating custom machine learning models can add high one-time and ongoing expenses, increasing the financial pressure during the early stages.

Handling large amounts of user-uploaded data and providing real-time feedback requires expensive resources. For example, operating machine learning models and frequently using APIs from big LLM providers adds to the ongoing costs for cloud services and computational needs.

Operational costs represent another critical constraint. Hosting and maintaining a platform capable of processing large volumes of user-uploaded content while delivering real-time feedback requires expensive resources. A significant portion of these costs arises from running machine learning models and making frequent API calls to LLM providers to deliver real-time responses.

Monetization is another challenge because Edux needs to be affordable for students and schools while still being financially sustainable. If the platform relies on subscriptions or tiered pricing, it might become less accessible to users with limited financial resources, reducing its impact.

Proper resource management is very important in order to focus on the features that matter the most. For example, adding voice recordings or customizing skill trees might be prioritized to keep resources manageable, especially during the early stages of development.

Convincing schools and teachers to use Edux is another challenge. This might require heavy spending on marketing and providing training support, which increases both financial and time requirements. These financial challenges shape how Edux can grow and scale, making good financial planning and smart resource use key for its long-term success.

Finally, Convincing schools and teachers to use Edux is another challenge. This might require heavy spending on marketing and providing training support, which increases both financial and time requirements. These economic constraints shape the development and scalability of Edux, requiring careful financial planning and resource management to ensure long-term success.

4.1.9.3. Ethical Constraints

The development and deployment of Edux face several ethical challenges, including academic integrity, bias, data privacy, accessibility, and instructor oversight. Maintaining academic integrity is critical, as the platform must ensure its tools are not misused by students, such as relying on LLMs to complete assignments dishonestly. Additionally, the evaluation system designed to identify discrepancies between elements like code and reports must be accurate and fair, ensuring students are not unfairly penalized for legitimate differences in their work.

Another important ethical challenge is addressing bias and ensuring proper representation. The platform must ensure that content generated by LLMs—such as quizzes, flashcards, and study guides—is free from unintentional stereotypes, biases, or errors that might marginalize or misrepresent certain groups of learners. Inclusivity is crucial to delivering equal value to all users, making sure that the study materials are suitable and effective for learners from different backgrounds and perspectives.

Data privacy is a vital ethical responsibility that demands strong measures to protect user-uploaded educational content from unauthorized access, misuse, or commercial exploitation. Clear and transparent communication about data usage policies is essential to establish trust, ensuring that users are fully informed and give their consent before any data is collected or processed.

Accessibility is another key ethical responsibility. The platform must be built to support users with disabilities by including features like screen reader compatibility and alternative input options. Moreover, tools for personalized scheduling and recommendations should be designed to respect different learning speeds and individual circumstances, avoiding generic approaches that could disadvantage certain learners.

Last, Edux must carefully balance instructor oversight with learner independence. The platform's tools should support and complement the role of instructors rather than replace or undermine their guidance in the learning process. By addressing these ethical challenges, Edux can foster a fair, inclusive, and responsible learning environment that respects the needs and rights of its users.

4.1.10. Standards

The development of Edux follows strict standards in usability, accessibility, security, and technological integration to create an effective and reliable educational platform. Usability is achieved through an intuitive design and user-focused features, making it easy for learners with different levels of technical expertise to use the platform effectively. Accessibility is a priority, as is ensuring the platform complies with guidelines to support inclusive learning for individuals with diverse needs and abilities.

The standard we will use for developing the project is IEEE 12207 [8]. It encompasses software life cycle processes, supply, development, operation, and maintenance. It also helps with documentation, configuration management, quality assurance, verification, and validation. The standard mainly helps developers by tailoring processes to fit project needs, promoting a systematic and organized approach to software engineering. We were advised to use this system by our course instructor, Mert Bıçakçı, for a smoother and easier development course.

Security is handled with industry-standard measures, including end-to-end encryption for protecting user data, strict adherence to GDPR regulations for data protection, and strong authentication protocols to maintain privacy and data integrity. Technological standards focus on scalability and reliability, utilizing cloud infrastructure, modular code architecture, and advanced language models for better content analysis and interaction.

The platform's development adheres to best practices in the software development life cycle, including version control, continuous integration and deployment (CI/CD), and thorough testing to ensure functionality, performance, and reliability. All features are built with future growth in mind, enabling the integration of advanced tools like RAG and voice-processing capabilities without disrupting existing functionality. These standards define Edux as an effective tool for both learners and instructors.

4.2. Risks and Alternatives

4.2.1. Data Privacy and Security

Edux handles sensitive academic data, including user-uploaded educational resources and personal information, making data privacy and security paramount. Any data breach could compromise user trust and lead to legal consequences. To reduce this risk, Edux can employ robust encryption protocols, multi-factor authentication, and regular security audits. Unauthorized access is further prevented through strict role-based access controls. These measures ensure compliance with KVKK and GDPR and other international data protection standards, reducing the likelihood of security lapses and bolstering user confidence in the platform.

4.2.2. Accuracy of Content Recommendations

Edux generates personalized learning tools using AI models. Errors in these recommendations could negatively impact learning outcomes, such as misaligned study schedules or inaccurate quiz questions. To address this, Edux will incorporate continuous validation of its algorithms using real-world data and user feedback. Regular updates and refinements to the recommendation models will ensure accuracy and relevance, minimizing the risk of misleading users and maintaining the platform's credibility.

4.2.3. Chatbot Model Selection and Cost Considerations

Edux plans to use a token-based ChatGPT 4.0-mini LLM to generate responses in the chat feature of the platform. However, the cost of using this model is a concern. If Edux is unable to afford the required expenses, the current Gemini API model will continue to be used as a fallback. This ensures continuity of the chat service, albeit at potentially lower accuracy.

4.2.4. User Adoption and Training

The success of Edux relies heavily on how effectively users, including students and instructors, can navigate and utilize its features. Resistance to adopting new technologies or insufficient familiarity with the platform may hinder its effective use. To overcome this, Edux will include a user-friendly interface and detailed documentation to guide users through its functionalities.

4.2.5. Regulatory Compliance

Edux is required to adhere to various educational and data protection regulations that differ by region, such as the GDPR in the European Union. Failure to comply could lead to legal and financial consequences. To address this, Edux must collaborate closely with legal experts to ensure compliance with both local and international regulations. Conducting regular compliance audits and updating the platform's policies and practices are essential to keep Edux in line with changing legal requirements.

4.2.6. Risks and Alternatives Summary

Risk	Likelihood	Effect on Project	Alternative Solution
Inaccurate Content	Medium	Poorly crafted	Continuous validation

Recommendations/Generation		personalized learning tools can hinder user learning experiences and diminish the platform's credibility.	and updates to AI recommendation models based on real-world data and user feedback.
Inability to Afford GPT 4.0-mini	High	If Edux cannot afford GPT 4.0-mini, responses may lack the desired quality, leading to lower user satisfaction and engagement.	If costs are prohibitive, fallback to Gemini API for responses with adjusted accuracy.
Data Privacy Breach	Low	A data breach could damage user trust, result in legal consequences, and negatively impact platform's reputation.	Employ encryption protocols, multi-factor authentication, and role-based access controls to ensure data privacy and security.
Difficulty Navigating the Application	Medium	Resistance to adopting new technologies or insufficient familiarity with the platform could limit user engagement.	Develop a user-friendly interface and provide detailed documentation.
Regulatory Non-Compliance	Low	Non-compliance with regulations could lead to legal and financial consequences, as well as damage to reputation.	Regular compliance audits and updates to platform policies, with legal expert collaboration to ensure adherence to regulations.

4.3. Project Plan

Our next goal is to add RAG to give users more personalized and dynamic content. With this feature, we aim to provide better answers that match users' needs, making sure the content is accurate and fits their specific situation.

We also plan to improve the UI to make it simpler and more attractive. The design will look better and be easier to use, so users can quickly find what they need and access all features without any problems.

Another important feature we will work on is the skill tree functionality. This feature will help users see their progress, set goals, and unlock new features or content as they move forward. It will give users a clear way to follow their learning path and stay motivated.

We will also be developing a Code Checker System. This system will check the code and reports submitted by students to see if they used AI tools to complete their work. By detecting possible cheating, the system will ensure fair evaluations and maintain honesty in assessments.

Finally, we will introduce admin and instructor panels to complete the education system. These panels will provide tools for managing users, content, and assessments efficiently. The details and UI mockups for both panels have already been explained earlier in this report.

WP 1: RAG Integration	
Start Date: 3 March 2024	**End Date:** 30 March 2025
Leader: Görkem Kadir Solun	**Members Involved:** Bilginer Oral, Murat Çağrı Kara
Objectives: RAG integration for specialized content generation	
Tasks: **Task 1.1** Set up RAG environment: Prepare and configure the required tools and environment for RAG integration. **Task 1.2** Develop retrieval pipeline: Design and implement the retrieval component for content generation. **Task 1.3** Integrate RAG with backend: Connect the RAG model to the existing backend infrastructure.	
Deliverables: **D1.1** Configured RAG environment ready for integration. **D1.2** Successful retrieval pipeline implementation for content generation.	

WP 2: UI Improvements	
Start Date: 17 March 2025	**End Date:** 20 April 2025
Leader: Murat Çağrı Kara	**Members Involved:** Efe Kaan Fidancı, Bilginer Oral
Objectives: Improve the user interface to enhance user experience and optimize design consistency across the application.	
Tasks:	

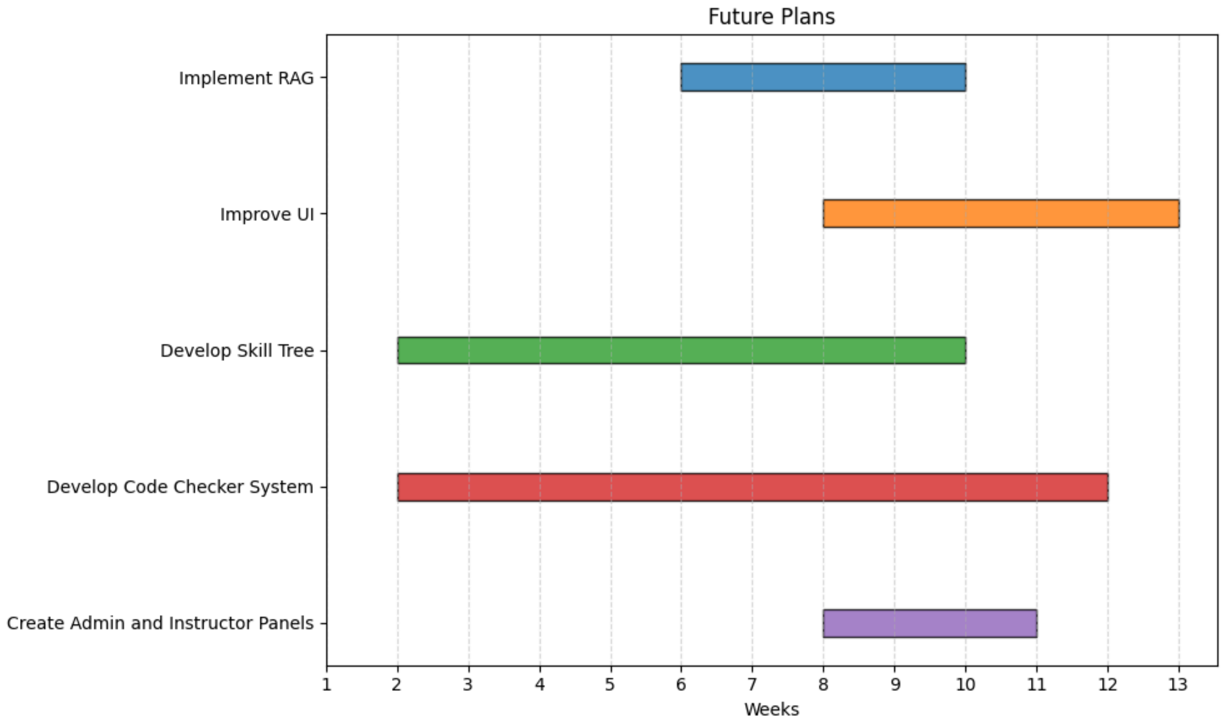
Task 2.1 Conduct UI analysis: Analyze the current UI and gather feedback for improvements. **Task 2.2** Redesign interface components: Implement updated designs for key interface elements. **Task 2.3** Optimize responsiveness: Ensure the application UI works seamlessly across devices and screen sizes. **Task 2.4** Perform usability testing: Test the updated UI with users to identify any issues and gather feedback.
Deliverables: **D2.1** Finalized design updates for key UI components. **D2.2** A responsive and user-tested application interface.

WP 3: Skill Tree	
Start Date: 3 February 2025	**End Date:** 30 March 2025
Leader: Murat Çağrı Kara	**Members Involved:** Cahit Ediz Civan, Görkem Kadir Solun
Objectives: Implement skill tree functionality for dynamic user progression and customization.	
Tasks: **Task 3.1** Design skill tree structure: Develop the layout and logic for the skill tree system. **Task 3.2** Implement backend logic: Integrate the functionality to track and manage user skill progression. **Task 3.3** Develop frontend components: Create visual components to display and interact with the skill tree. **Task 3.4** Test functionality: Perform unit and integration tests to ensure the skill tree works as intended.	
Deliverables: **D3.1** A functional skill tree system integrated with the application. **D3.2** Completed testing and validation of skill tree features.	

WP 4: Code Checker	
Start Date: 3 February 2025	**End Date:** 13 April 2025
Leader: Bilginer Oral	**Members Involved:** Everyone

Objectives: Implement a code checker to evaluate the integrity of code submissions and reports while supporting dynamic user progression and customization.
Tasks: **Task 4.1** Define code evaluation criteria: Establish rules and benchmarks for assessing code quality and integrity. **Task 4.2** Develop code validation logic: Implement backend logic for code integrity and error detection. **Task 4.3** Integrate report analysis: Add functionality to evaluate and verify report submissions. **Task 4.4** Customize user progression: Link code-checker results to dynamic skill progression for users.
Deliverables: **D4.1** Functional code checker tool for evaluating code submissions. **D4.2** Integrated report validation system. **D4.3** User progression system linked to code checker results.

WP 5: Admin and Instructor Dashboards	
Start Date: 17 March 2025	**End Date:** 6 April 2025
Leader: Cahit Ediz Civan	**Members Involved:** Efe Kaan Fidancı
Objectives: Develop dashboards for administrators and instructors to monitor, manage, and analyze user activities and system performance.	
Tasks: **Task 5.1** Design dashboard layout: Plan and design user-friendly interfaces for both admin and instructor dashboards. **Task 5.2** Implement data visualization: Integrate visual components to display key metrics and performance indicators. **Task 5.3** Add user management features: Develop tools for managing users, roles, and permissions. **Task 5.4** Integrate reporting tools: Provide functionality for generating and exporting system activity reports.	
Deliverables: **D5.1** Fully functional admin and instructor dashboards. **D5.2** Visualized metrics and reporting tools integrated into the system. **D5.3** User management interface with role-based access controls.	



4.4. Ensuring Proper Teamwork

To ensure everyone contributes effectively, we divide the project into smaller tasks and aim for a balanced workload. Since this is a large project, managing tasks and keeping everyone informed can be tricky. Since we value and respect each other, it is important to divide the work equally. At the end of CS491/2, each team member aims to evaluate one another objectively by using peer grading deliverable or catme. To avoid problems, we use the following tools:

Google Docs: We utilize Google Docs to record meeting notes and share insights gained during discussions with our supervisor and coordinators. Collaborative note-taking ensures all key points and action items are captured. These notes are subsequently transferred to Jira [9] for ongoing tracking and management. Additionally, we leverage Google Docs to create comprehensive documentation for developers new to specific areas of the codebase, such as backend, and middleware.

Jira: To keep things organized, we use Jira to handle our projects. It helps us break down big tasks into smaller ones and track who's responsible for what. We work in short sprints (usually one or one and a half weeks long) to make sure we stay on top of things, especially during busy academic periods.

GitHub: We use git as our version control system and GitHub [10] to host it, enabling team members to collaborate on the project simultaneously. This approach ensures everyone can work on their tasks independently without interfering with others' progress. Additionally, GitHub plays a key role in tracking individual contributions.

4.5. Ethics and Professional Responsibilities

The development and deployment of Edux involve various ethical and professional considerations that must be carefully addressed. Protecting data privacy and security is a top priority, as users will upload sensitive academic materials, which may include proprietary or copyrighted content. Compliance with copyright laws and intellectual property rights is essential to prevent unauthorized use or redistribution of such materials.

The integration of LLMs presents ethical challenges, particularly in ensuring the accuracy and reliability of the content generated. Errors in explanations could mislead users, so it is important to provide transparency about the limitations of AI-generated content to maintain user trust and prevent misuse. The platform's features for generating personalized schedules and assessments must be designed to avoid biases that could unfairly impact certain user groups.

The feature for evaluating the alignment between submitted reports and assigned code must be carefully implemented to avoid excessive reliance on automated tools. Over-reliance could undermine critical thinking and problem-solving skills if the tools are not thoughtfully integrated into the educational process.

Pay tiers must be structured ethically to ensure that advanced learning tools remain accessible to all users, regardless of their socio-economic status, thereby avoiding creating barriers to equitable education.

Finally, the user-instructor collaboration feature must include safeguards to prevent misuse and ensure that interactions promote fairness and contribute positively to the learning experience for both students and instructors. By addressing these ethical and professional considerations, Edux can uphold its commitment to providing a responsible and inclusive educational platform.

4.6. Planning for New Knowledge and Learning Strategies

To successfully develop Edux and its innovative features, our team recognizes the necessity of acquiring new knowledge and refining our learning strategies. We have identified key areas requiring deeper understanding, such as advanced large language models (LLMs), personalized learning tool integration, and dynamic progress tracking methodologies. To bridge these knowledge gaps, we plan to engage in comprehensive research and training sessions. This includes participating in workshops, online individual studies, and webinars focused on artificial intelligence, machine learning, and educational technology. Collaborating with experts in these fields will provide us with valuable insights and hands-on experience, ensuring that we are well-equipped to implement these advanced features effectively.

In addition to individual learning, we are fostering a collaborative environment to enhance team-wide knowledge acquisition. Regular knowledge-sharing meetings, brainstorming sessions, and peer reviews will be integral to our strategy, allowing team members to share insights and learn from each other's experiences. We also plan to stay abreast of the latest trends and best practices by subscribing to relevant journals, attending industry conferences, and engaging with professional networks. By strategically planning for new knowledge acquisition and continuously refining our learning strategies, we aim to not only overcome current challenges but also to position Edux at the forefront of educational innovation, ultimately delivering a platform that meets the evolving needs of modern learners and educators.

5. Glossary

LLM: Large language model
A Large Language Model (LLM) is an advanced artificial intelligence system trained on extensive text datasets to understand and generate human-like language. These models are pivotal in natural language processing tasks, including text generation, translation, and sentiment analysis [11].

API: Application programming interface
An Application Programming Interface (API) is a set of protocols and tools that allow different software applications to communicate with each other [12]. APIs enable developers to access specific functionalities or data from external services without needing to understand their internal workings.

RAG: Retrieval augmented generation
Retrieval Augmented Generation (RAG) is a technique that combines large language models with external data sources to enhance the accuracy and relevance of generated content [13]. By retrieving pertinent information from databases or documents, RAG systems can produce more informed and contextually appropriate responses.

GDPR: General data protection regulation
The General Data Protection Regulation (GDPR) is a comprehensive data protection law enacted by the European Union to safeguard individuals' personal data and privacy. Implemented in 2018, it imposes strict guidelines on data collection, processing, and storage, with significant penalties for non-compliance.

TPS: Transactions per second

Transactions Per Second (TPS) is a metric used to measure the number of transactions a system can process in one second. It's commonly used to assess the performance and scalability of databases, networks, and other transactional systems [14].

ID: Identifier
An Identifier (ID) is a unique symbol or sequence assigned to an entity to distinguish it from others. IDs are essential in databases, programming, and various systems to reference and manage specific records or objects efficiently.

6. References

[1] Facebook, Inc. (n.d.). React - A JavaScript library for building user interfaces. [Software].

Retrieved from https://reactjs.org

[2] Sebastián Ramírez. (n.d.). FastAPI. [Software]. Retrieved from https://fastapi.tiangolo.com

[3] gRPC Authors. (n.d.). gRPC - A high performance, open-source universal RPC framework. [Software]. Retrieved from https://grpc.io

[4] PostgreSQL Global Development Group. (n.d.). PostgreSQL - The World's Most Advanced Open Source Relational Database. [Software]. Retrieved from https://www.postgresql.org

[5] European Parliament, Council of the European Union. (2016). Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). Official Journal of the European Union.

[6] Visual Paradigm International Ltd. (2023). Visual Paradigm 16.2. [Software]. Retrieved from https://www.visual-paradigm.com

[7] Object Management Group (OMG). (2017). Unified Modeling Language (UML), Version 2.5.1. OMG.

[8] IEEE. (2017). IEEE Std 12207-2017 - IEEE/ISO/IEC International Standard - Systems and software engineering -- Software life cycle processes. IEEE.

[9] Atlassian. (n.d.). Jira Software. [Software]. Retrieved from https://www.atlassian.com/software/jira

[10] GitHub, Inc. (n.d.). GitHub. [Software]. Retrieved from https://github.com

[11] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language Models are Few-Shot Learners. arXiv preprint arXiv:2005.14165.

[12] Fielding, R. T. (2000). Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine.

[13] Lewis, P., Liu, E., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., & Stoyanov, V. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv preprint arXiv:2005.11401.

[14] Gray, J., & Reuter, A. (1993). Transaction Processing: Concepts and Techniques. Morgan Kaufmann Publishers Inc.