Bilkent University - Computer Science

CS491 - Senior Design Project I

Project Analysis and Requirements Report

T2401

Ömer Fırat Bekiroğlu 22002239
Faruk Uçgun 22003016
Cenk Merih Olcay 22002408
Enes Bektaş 2202401
Akif Erdem Tanyeri 22003537

2024 Fall

1. Introduction

Medical school students face an overwhelming volume of material during their studies. Research conducted by Dattathreya and Shillingford found that medical school students have trouble remembering all of this information [1]. Although the causes of these challenges differ, research shows that most students avoid textbooks and rely mostly on lecture slides [2]. Consequently, individuals are 'learning passively', repeatedly reading the same content, which is an ineffective learning strategy that causes memorization issues [3]. ReMediCard.io seeks to address these issues and facilitate medical school students' learning.

Although there are a number of applications available for medical students currently, none of them truly tackle the difficulties associated with creating flashcards. Current programs like Anki and Quizlet offer limited customization options and tools for creating flashcards by hand, but they demand a large time commitment from students, which can be difficult for medical students with heavy workloads. Two well-known apps with spaced repetition capabilities are Anki and Quizlet. Nevertheless, Anki in particular is unable to generate flashcards automatically from handwritten notes and other legacy teaching materials. Quizlet still relies on the user to manually generate flashcards, even though it features a document scanning tool that allows users to choose particular words from the provided document.

ReMediCard.io is an AI-powered flashcard application intended to help medical students study more efficiently, in recognition of the difficulties that come with medical school. Our product reduces the amount of manual labor needed by automatically creating flashcards from lecture notes, videos, and images by utilizing computer vision, machine learning, and natural language processing. Students can concentrate more on studying and less on preparation thanks to its automation.

Additionally, ReMediCard.io uses adaptive learning algorithms to customize study sessions based on each user's performance and progress. This personalized approach not only enhances learning efficiency but also aligns with the diverse needs of medical students. The integration of these cutting-edge technologies is the innovative part of our project, differentiating us from similar products in the market.

The remainder of this report outlines the analysis and requirements of ReMediCard.io. Section 2 presents the current systems in the market. Section 3 includes a detailed description of our design. Section 4 explains engineering principles taken into account, possible risks, our current plans, and alternative plans.

2. Current System

Most medical students rely on traditional study methods, which typically include reading textbooks, lecture slides, or rewatching lecture videos. According to the poll we conducted among medical students from different universities, 66% are not using a flashcard application. These traditional methods often involve significant time investment and are not the most effective for retention and understanding of complex material.

Current applications like Anki and Quizlet are well-known flashcard applications. However, they heavily rely on manual input from the users to create their content. This approach is time-consuming, and it can become overwhelming for the users. In our case, these users will be medical school students who already have less time for such manual interaction.

While Anki is a popular flashcard application, known for its deep customization and plugin support. It requires quite a lot of effort to create flashcards by hand. ReMediCard.io comes into play here with an auto flashcard generation feature making use of the documents provided by the user, such as lecture slides, personal notes and textbooks. This could not be viable for many medical students who already have enough workload.

Quizlet has a user-friendly interface which is more suitable for comfortable usage and better document features such as document scanning. But still relies on users' manual interaction while creating flashcard content. While their document scanning tool can extract words or phrases, it is not fully automated and creates a time-consuming experience, making the process less efficient.

The aforementioned limitations indicate a huge opening for an application that is capable of automating the process of converting traditional educational materials into modern and more personalized flashcards and decks. ReMediCard.io merges highly personalized flashcards with automation to fill this gap. Through the deployment of computer vision and machine learning techniques, it will extract the relevant information from legacy materials and automatically generate flashcard decks based on the personal experience of each student. This system will lift the burden on the medical students to manually create flashcards, saving time and effort so they can focus more on studying.

3. Proposed System

The subsections below describe our proposed system, outlining the added features to the aforementioned current system. Our discussion covers both functional and

non-functional requirements of the system, along with static and dynamic diagrams illustrating core functionalities, including class, use case, activity, and state diagrams.

3.1. Overview

ReMediCard.io is a mobile application designed to help medical students increase their efficiency and effectiveness during studying. As the medical studies include overwhelming amounts of memorization and repetitions, ReMediCard.io proposes a personalized solution that utilizes the advantages of flashcards, and reinforces them with computer vision, machine learning, and natural language processing technologies.

The features provided by our application represents what other flash card applications lack: they provide complete automation within flashcard creation in a personalized manner. By analyzing the traditional educational materials such as lecture notes, voice recordings, videos or figures, the application will generate highly relevant and authentic flashcard decks tailored for each user. This will fit with the content of their studies focusing on their target areas, meanwhile saving them from the waste of time and the burden of manual flashcard creation.

ReMediCard.io is not only a flashcard application. It creates an interactive studying environment by offering features like study goals tracking and study statistics. Additionally, the project has features such as preparing quiz questions and giving feedback to increase interactivity and provide a more effective studying. Moreover, users will be able to share quizzes and decks they prepared, creating an environment that supports peer-to-peer learning.

3.2. Functional Requirements

- The user can register/login by using their email.
- The user can register/login by using their Google accounts.
- The user can reset their password using their email.
- The user can delete their account.
- The user can create custom flashcard decks.
- The user can create/edit custom flashcards and add them to the decks.
- The user can attach media(link, voice record, image, or video) to each side of a flashcard.
- The user can upload a video lecture record and generate a deck of cards automatically.

- The user can upload a voice record and generate a deck of cards automatically.
- The user can upload their lecture notes and generate a deck of cards automatically.
- The user can upload a figure with labels and generate the corresponding deck automatically.
- The user can upload test questions with their answers to the systems and can generate corresponding flashcards and similar questions.
- The user can automatically generate a test by selecting a set of flashcards or decks.
- The user can be shown flashcards based on the spaced repetition algorithm. For example, if they struggled last time to remember the backside of the flashcard, they can expect to encounter it more often when reviewing the corresponding deck or vice versa.
- The user can get feedback for the solutions of the practice test questions in the application.
- The user can get indirect hints while solving test questions or before flipping the flashcards.
- The user can search flashcards, questions or media(if available) based on keywords.
- The user can monitor their study statistics.
- The user can share their flashcard decks with other users
- The user can see and use other user's flashcard decks if the publishing user consents to share.
- The user can set study goals and receive notifications about these goals.

3.3. Nonfunctional Requirements

3.3.1. Usability

User Friendly UI:

- A clean, simple, and visually appealing design will be applied
- The application will have easily understandable content where users can manage learning materials and create flashcards.
- The application will be designed in a way that users can easily learn and use different features of the application.

Easy Registration and Login:

- Users will be able to log in to the system more practically by registering, logging in via email, and integrating with their Google accounts

Responsive Design:

- The system should be designed to adapt to different screen sizes and devices.

Feedback:

- Users will be able to learn whether their actions were successful or not with visual or text-based feedback when using the application.
- Explanatory warnings will be presented to the user when incorrect or incomplete actions are taken.

3.3.2. Reliability

Uninterrupted Service:

- The system will be accessible to users at all times.
- In case of possible interruptions, the application will be put back into service as soon as possible.

Data Consistency and Management:

- User data and system logs should always be stored accurately and up to date.
- Systematic backups will be made to prevent data loss.
- In case of data loss, the system will be able to be quickly restored from the latest backup.

Version Control and Updates:

- Updates will be performed without harming existing user data and the ongoing system.
- A rollback procedure will be prepared, and in case of possible version update errors, this procedure can be followed to revert to the last working version.

Monitoring and Testing:

- The system will undergo extensive testing before deployment.
- New updates will undergo necessary tests before being released to the user.
- Once the system is deployed, it will be continuously monitored to identify performance and reliability issues.

3.3.3. Performance

User Interaction:

- The system should respond instantly to user actions, providing smooth interaction.
- The system should handle simultaneous active users without a reduction in performance.

Hosting:

- The project will be hosted in an environment strong enough to meet the required user demands.
- The hosting environment will be scalable to adapt to different levels of traffic and provide uninterrupted service during peak usage times.

Optimization:

- The application will be optimized to utilize system resources (CPU, memory, network) efficiently, ensuring no waste of resources.
- The system will be designed to meet user demands without compromising performance.

3.3.4. Supportability

Documentation:

- The reports and documentation we prepare will contain the necessary information about how the system is built and works.
- The structure and working procedure of new updates will be added to the documentation.

User Support:

- Detailed user manuals and FAQs will be provided for ease of use.
- Users will be able to report problems they encounter, and they can seek assistance through support channels such as email.

Compatibility:

- The mobile application will be compatible with commonly used mobile operating systems.
- The website will be compatible with commonly used browsers.
- The system will be tested regularly to ensure compatibility with newer versions of browsers and operating systems.

3.3.5. Extensibility

Architecture:

- The system will be built in a way that allows for the easy addition of new features or components without breaking the existing ones.
- New functions can be integrated into the core system with minimal changes, providing flexibility for future updates.
- The data will be scalable as the user needs and traffic increases.
- Additional data can be added without requiring significant changes to the database.

Third-party Integrations:

- The system will support easy integration with third-party tools and services, such as cloud storage platforms and Google accounts.

3.3.6. Scalability

The system will allocate computation resources elastically to satisfy the dynamic workload stimulated by the user activity to avoid potential latencies and unpredicted downtime. It will adjust to user demand, and the use of computation resources will be cost-effective.

3.3.7. Security & Privacy

The system will comply with modern secure authentication/authorization design practices such as role-based authorization and Oauth2 to provide a secure, personalized experience for the end users. User data transactions will be done atomically to avoid partial updates and data losses. Sensitive user data will be encrypted in transit to secure user privacy.

3.3.8. Maintainability

The system design will consist of modular structures (services) to allow updates to be added with ease, and each modular service will be independently modifiable. The project will be maintained with Git to keep track of changes in the source code.

3.3.9. Availability

The system will have a fault-tolerant infrastructure with load balancers and disaster recovery scenarios providing high uptime. With the redundancy of the compute resources and a scalable architecture, the system will fit into the variable user demand.

3.3.10. Localization

The system will support multi-regional use through multiple languages (Turkish and English) and date formats. This will allow the students with different languages of study to use the application without any problems.

3.4. Pseudo Requirements

- For project management and task tracking, Jira is used
- GitHub is used for version control and maintaining project code
- Expo will be used as the platform to develop the React-Native front-end app on, it will provide the default native cross-platform configurations and abstract the platform-dependent details.
- Expo Document Picker library will be handy for all the input media for the mobile application because, through its interface, all the document-based features will be implemented.
- Similar to the Expo Document Picker, Expo Image Picker will provide the interface for the development of the frontend for the image-based (labeled figures and lecture notes as inputs) functionalities.
- Expo Notifications will provide push notifications for periodic user feedback or progress-tracking reminders.
- Axios will be used in the front end for handling the communications with restful endpoints and ensuring reliable communication with the Spring-Boot backend.
- Spring Boot framework will be used during backend development
- Java Spring Data and Spring Data MongoDB will provide the interfaces to interact with the relational and document-based databases.
- React Native will be used for designing UI and achieving responsiveness for both web and mobile.
- MySQL will be the choice of database for the structured system data, user profiles, flashcard decks, statistics and so on.
- MongoDB will store text-indexed documents like video/voice record transcripts and lecture notes text, providing quick access and search capabilities.
- Amazon EC2 instances will host our restful backend, AI model, or Image Processing services and queues in our infrastructure in virtual machines.
- Amazon S3 buckets will store large image, video, and voice record files.
- Docker containers will be used to host the services abstracting the underlying host platform and version-dependent details.
- Fine-tuning models like GPT, Claude, and Llama for enhancing AI-based features such as generating flashcards, extracting questions, etc., from text, visual and audio data

3.5. System Models

3.5.1. Scenarios

Scenario 1: User Registration via Email

Actors: User
Entry Conditions: The user is neither logged in nor registered. The user accesses the registration/login page.
Exit Conditions: User account is created, and the user is redirected to the dashboard or login page.
Flow of Events:
1. The user navigates to the main page.
2. The user fills out the fields for username, email, password
3. The system verifies the input provided and checks if the email has been previously registered.
4. In case of validity, it proceeds with the creation of a user account.
5. An e-mail will be sent for confirmation.
6. Confirmation through a link click from the user.
7. The system forwards him/her to the login/dashboard page.

Scenario 2: Log-in through Google Account

Actors: User
Entry Conditions: The user is on the login page and has a valid Google account.
Exit Conditions: The user is logged in and redirected to their dashboard.
Flow of Events:
1. The user selects "Continue with Google."
2. The system redirects the user to Google's authentication service.
3. The user provides Google account credentials and authorizes the application.
4. Google returns the authentication token to the system.
5. The system checks the token and either gets or creates the user account.
6. The user is then taken to their dashboard.

Scenario 3: Password Reset via Email

Actors: User
Entry Conditions: The user is on the login page and clicks on the "Forgot Password" link.
Exit Conditions: The user is able to reset the password and gets transferred back to the login page.
Flow of Events:
1. The user clicks "Forgot Password" on the login page.
2. The user enters their registered email address.

3. The system sends a password reset link to the provided email.
4. The user clicks the reset link and enters a new password.
5. The system validates the new password and updates the user record.
6. The user is informed that the password was reset successfully and is taken to the login page.

Scenario 4: Delete Account

Actors: User
Entry Conditions: The user is logged in and opens the “Account” section for the settings..
Exit Conditions: The user account is deleted, and the user is logged out of the system.
Flow of Events:
1. The user opens account settings.
2. The user chooses "Delete Account."
3. The system asks for confirmation from the user.
4. The user confirms the deletion.
5. The system deletes the user account and associated data.
6. The system logs the user out and redirects them to the homepage.

Scenario 5: Create a Custom Flashcard Deck

Actors: User
Entry Conditions: The user is logged in and accesses the "Create Deck" feature.
Exit Conditions: A new flashcard deck is created and saved.
Flow of Events:
1. The user clicks "Create Deck" in the dashboard.
2. The user provides a name and optional description for the deck.
3. The system validates the input and creates the deck.
4. The user is redirected to the deck editing page to add flashcards.

Scenario 6: Add/Edit Flashcards to a Deck

Actors: User
Entry Conditions: The user wants to edit an existing deck or add a new flashcard by choosing a specific deck on the main page.
Exit Conditions: New flashcards are added to or updated in the deck.
Flow of Events:
1. The user selects a deck to edit or creates a new one.
2. The user enters flashcard details (e.g., front and back text).
3. The user optionally attaches media(image, video, document or link) to the flashcard.
4. The user saves the flashcard.

5. The system updates the deck with the new or modified flashcard.

Scenario 7: Create a Flashcard Deck from a Video Lecture or Voice Record

Actors: User
Entry Conditions: The user is logged in and accesses to the section “Auto Generate a Deck”
Exit Conditions: A new deck is created with flashcards based on the uploaded video/voice record.
Flow of Events:
1. The user uploads a video file or the voice record of a lecture or study session.
2. The system processes the audio of the given record to generate a transcript.
3. The system extracts the key points from the transcript summary and generates flashcards.
4. The system creates a new deck from the created flashcards.
5. Notify User: creation of deck completed. Provide the user with a way to view or edit the flashcards.

Scenario 8: Solve Flashcards Using Spaced Repetition

Actors: User
Entry Conditions: The User selects any deck to be reviewed.
Exit Conditions: Review session completion by the User and save the session statistics for future study sessions. Updated review schedule by system.
Flow of Events:
1. The user selects a deck and starts a review session.
2. The system presents flashcards based on the spaced repetition algorithm.
3. The user attempts to recall the answer or flips the card for a hint.
4. User marks the flashcard such as "Easily Remembered", "Hardly Remembered" or "Forgot."
5. The system updates the flashcard's review schedule based on user input.
6. The session ends, and the system gives a summary of performance.

Scenario 9: Share a Flashcard Deck

Actors: User
Entry Conditions: The user has created a deck and wants to share it.
Exit Conditions: The deck is shared with selected users and deep copies of the deck are generatable for those users.
Flow of Events:
1. The user selects a deck to share.
2. User set sharing preferences, such as selecting specific users or generating a public link.

3. The system verifies the preference list and creates a share link or gives permission.
4. User shares the link or informs the users selected.

Scenario 10: Set Study Goals and Receive Reminder Notifications

Actors: User
Entry Conditions: The user logs in and navigates to the section "Study Goals"
Exit Conditions: The study goals have been set, and notifications are turned on.
Flow of Events:
1. The user navigates to the goals section in the dashboard.
2. Users set study goals.
3. The user turns on notifications for reminders after setting the goals.
4. The system validates and saves the goals.
5. The system schedules notifications, tracks progress compared to goals, and creates push notifications.

3.5.2. Use Case Model

3.5.2.1. Performance UML Use Case Diagram

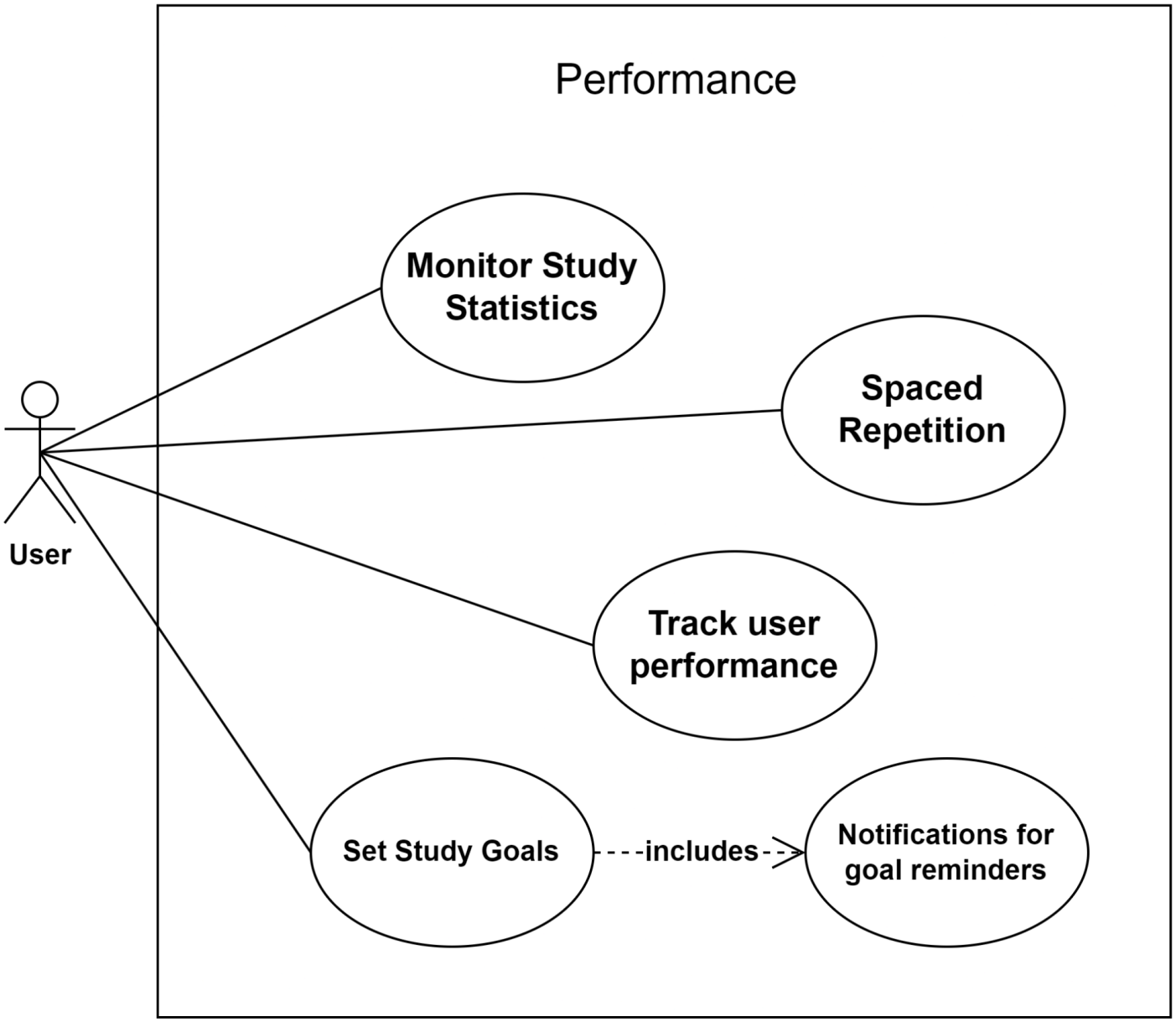


Fig. 1: Performance UML Use Case Diagram

3.5.2.2. Admin UML Use Case Diagram

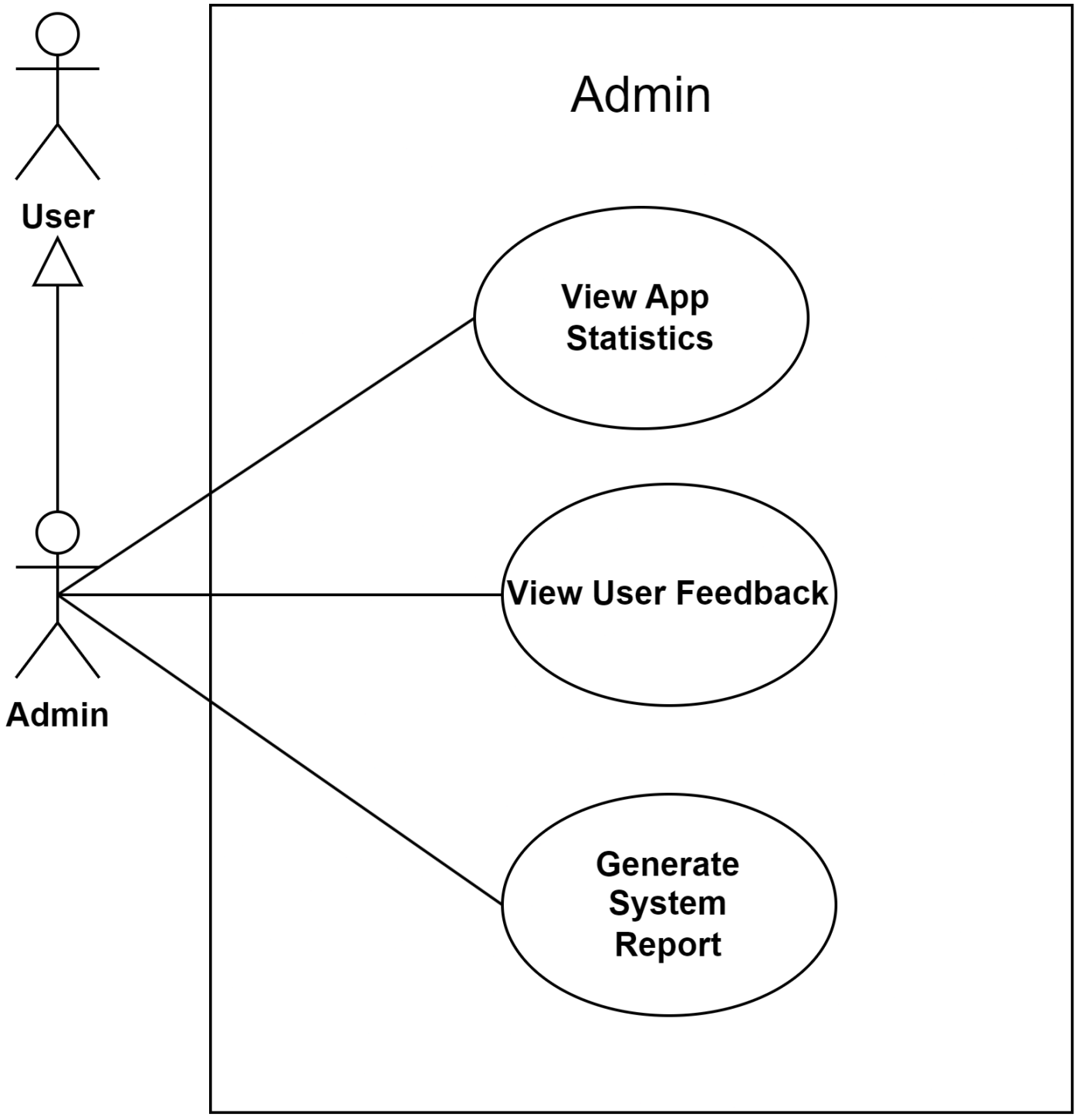


Fig. 2: Admin UML Use Case Diagram

3.5.2.3. Account UML Use Case Diagram

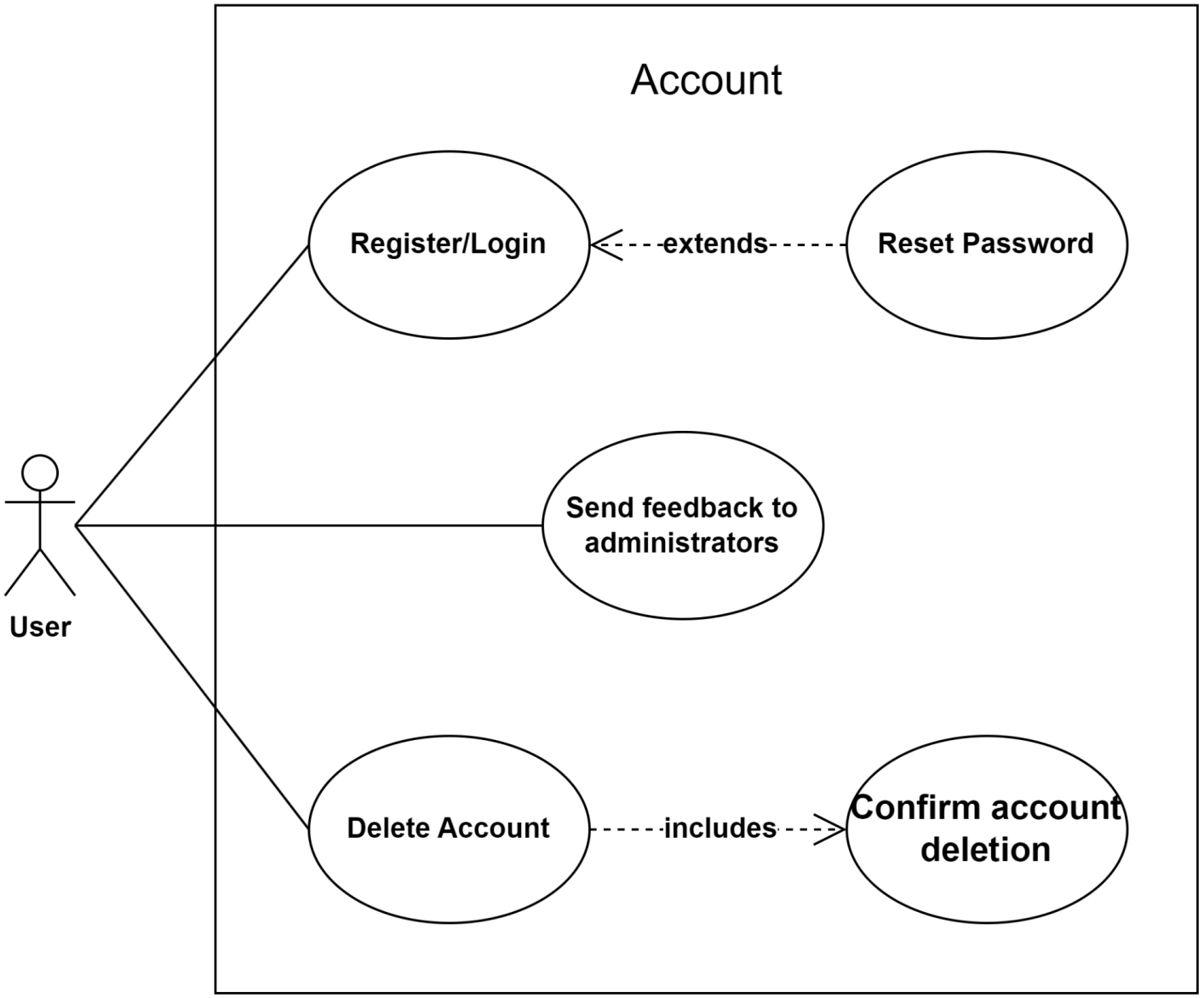


Fig. 3: Account UML Use Case Diagram

3.5.2.4. Deck / Question UML Use Case Diagram

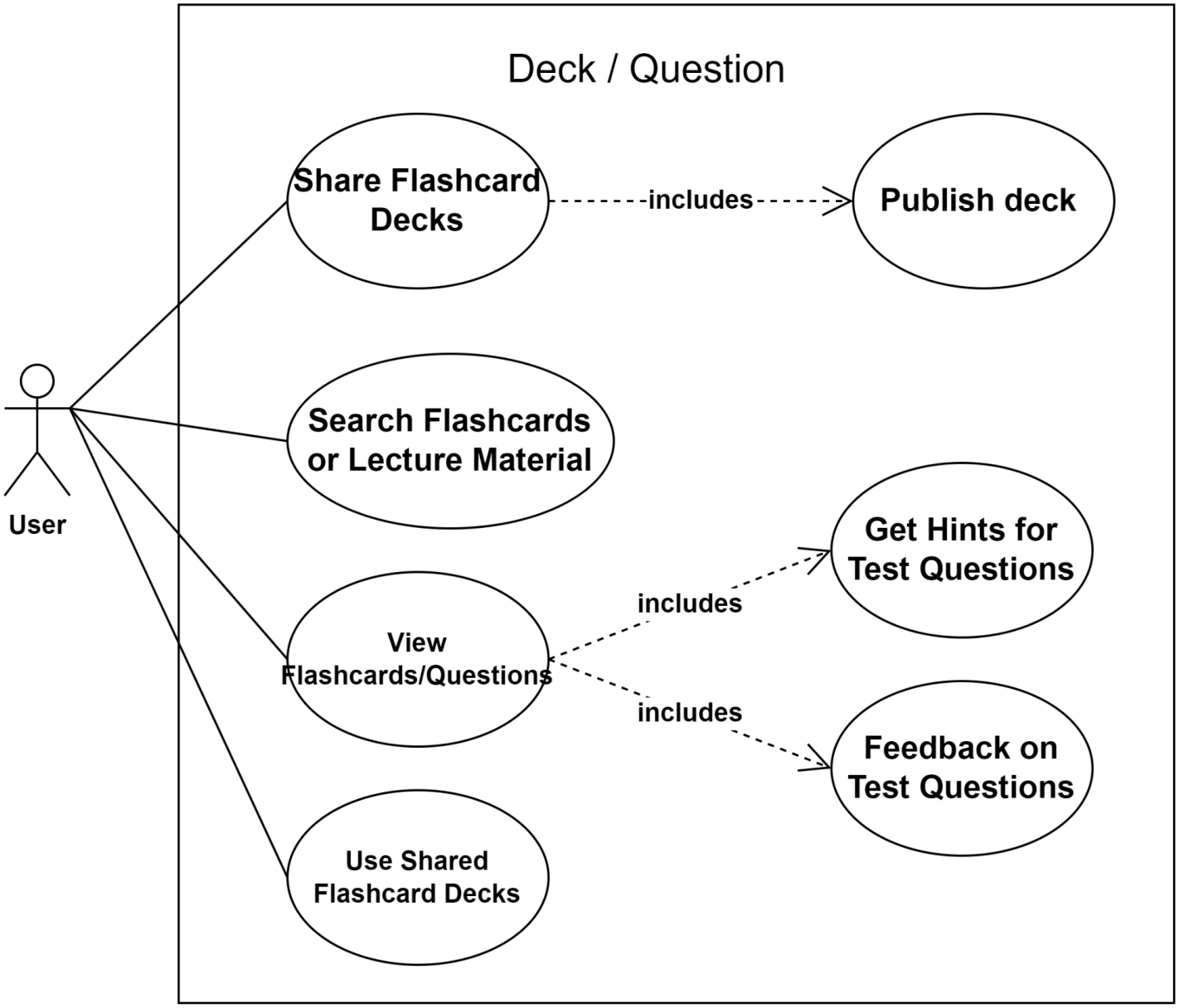


Fig. 4: Deck / Question UML Use Case Diagram

3.5.2.5. Flashcard UML Use Case Diagram

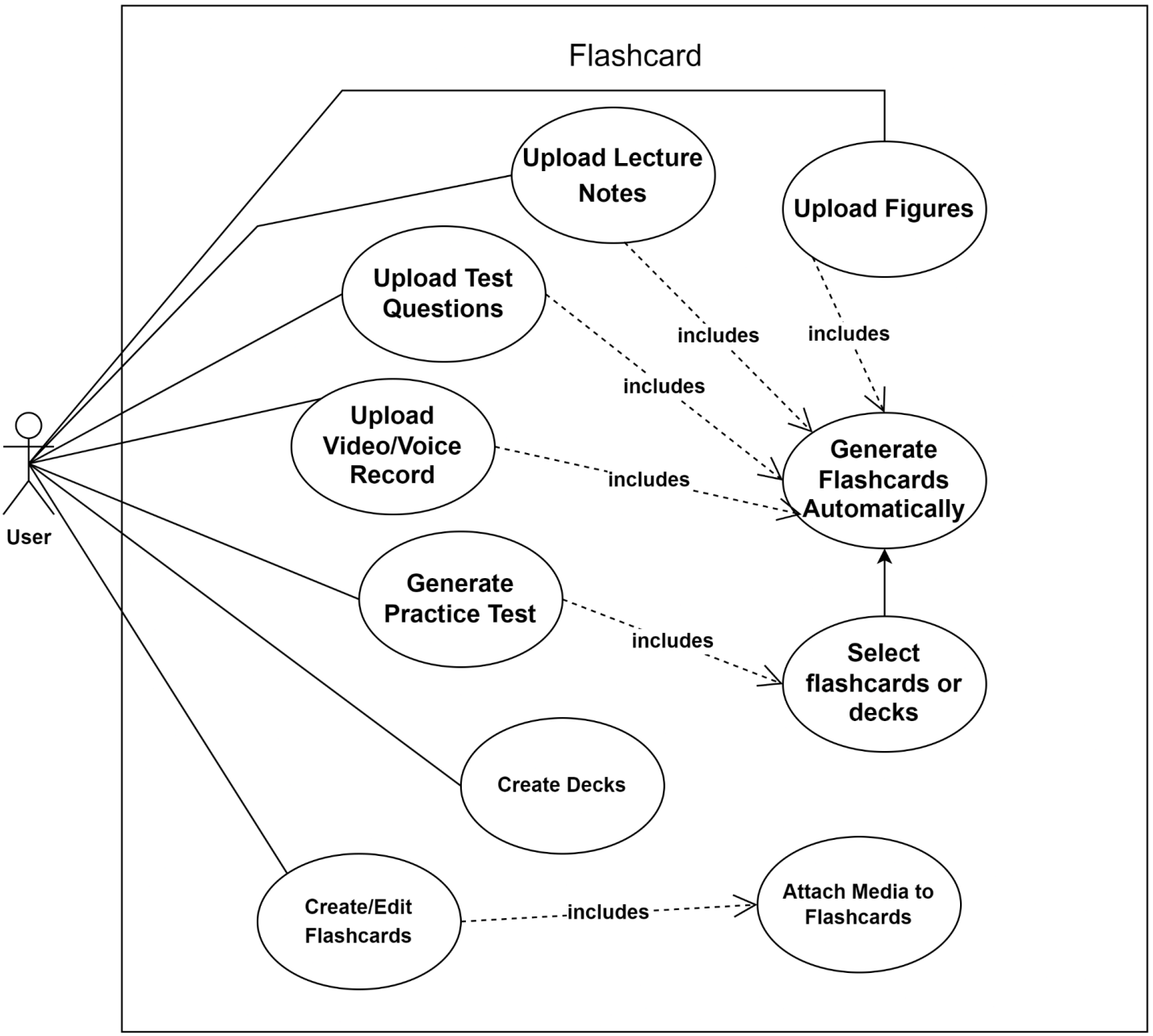


Fig. 5: Flashcard UML Use Case Diagram

3.5.3. Object and Class Model

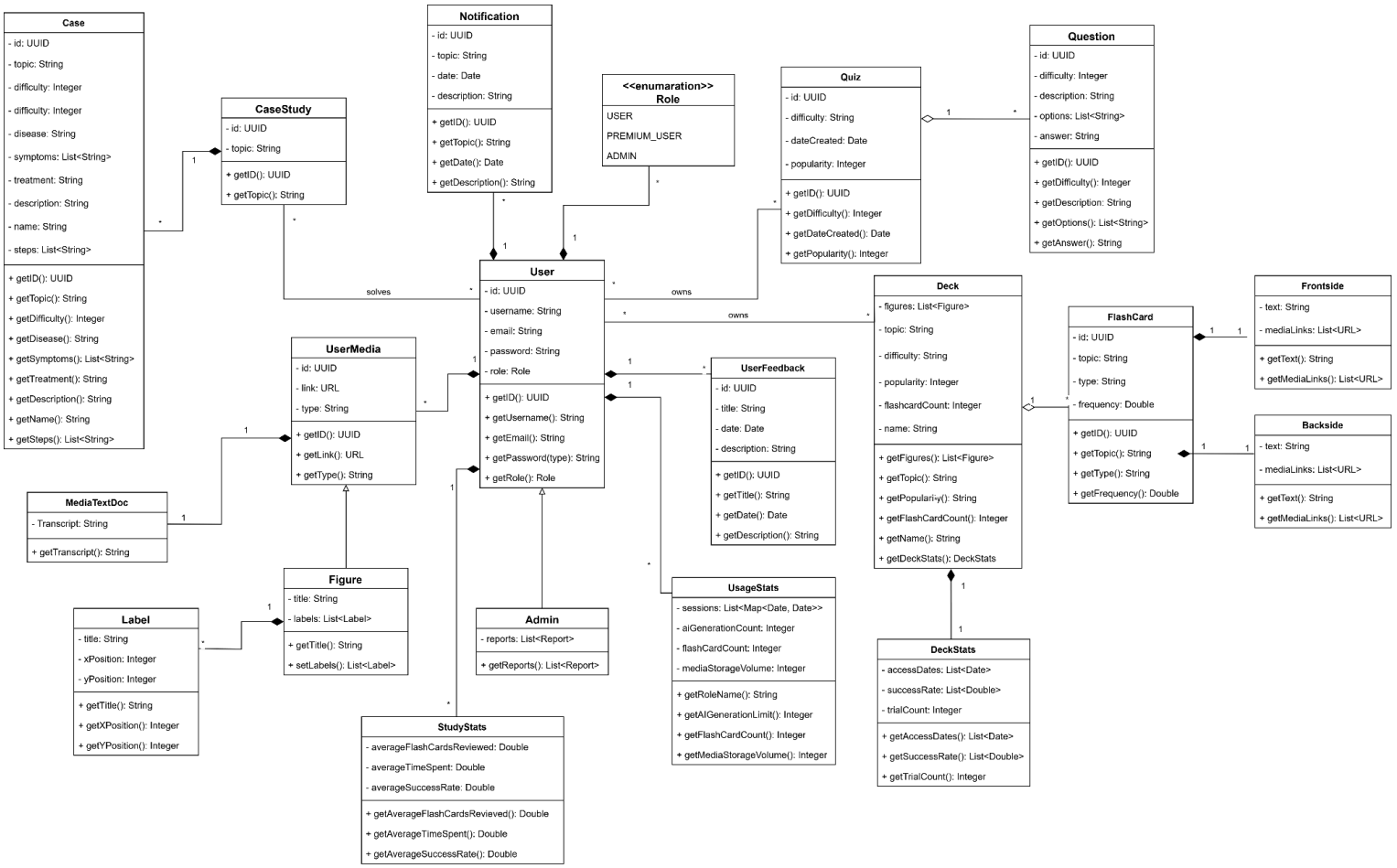


Fig. 6: UML Class Diagram

3.5.4. Dynamic Models

3.5.4.1. Automatic Flashcard Generation UML Activity Diagram

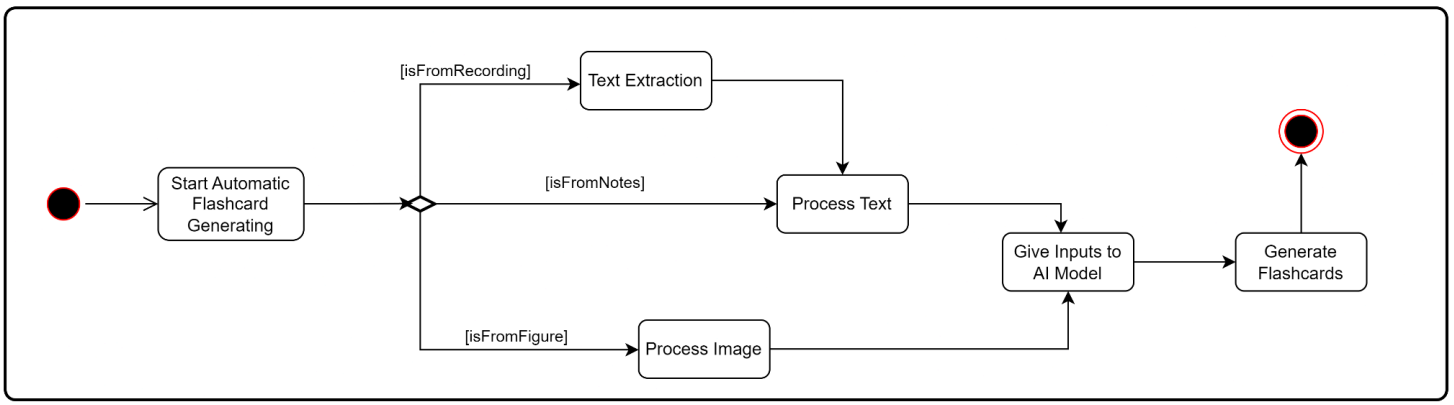


Fig. 7: Automatic Flashcard Generation UML Activity Diagram

3.5.4.2. Add/Edit Flashcards UML Activity Diagram

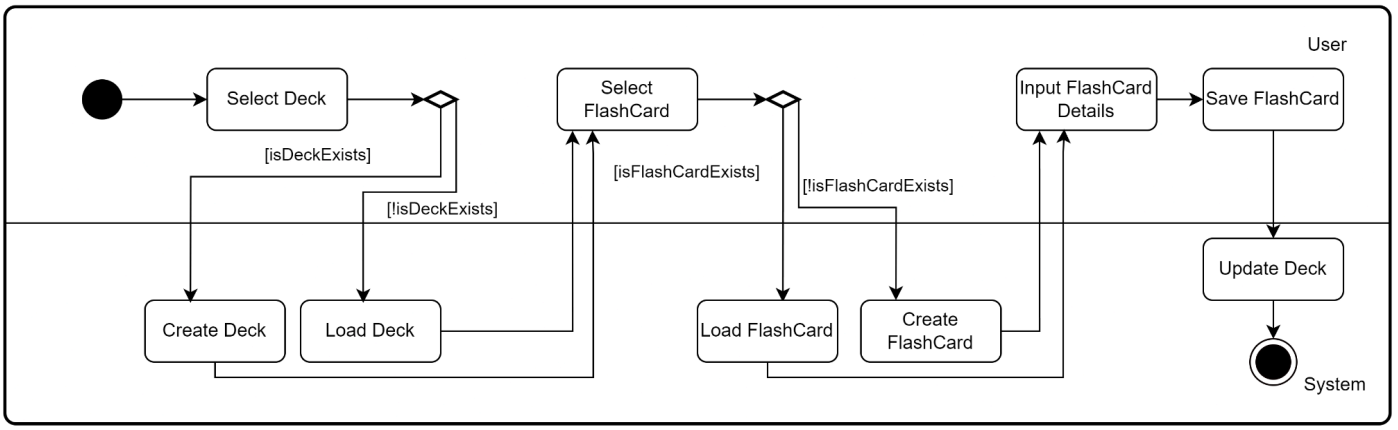


Fig. 8: Add/Edit Flashcards UML Activity Diagram

3.5.4.3. Solve Case UML Activity Diagram

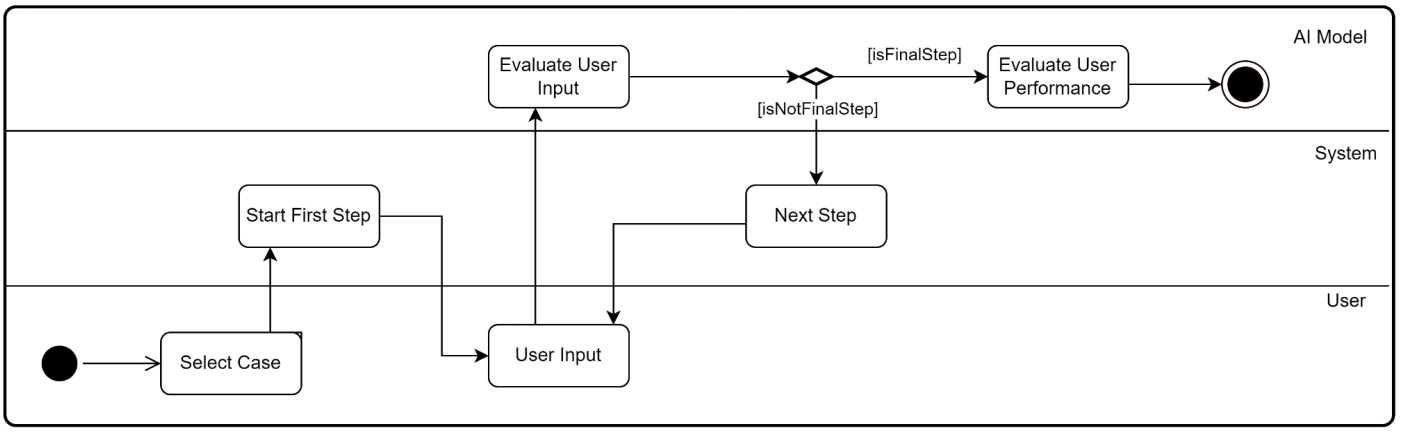


Fig. 9: Solve Case UML Activity Diagram

3.5.4.4. Set Study Goals and Receive Notifications UML Activity Diagram

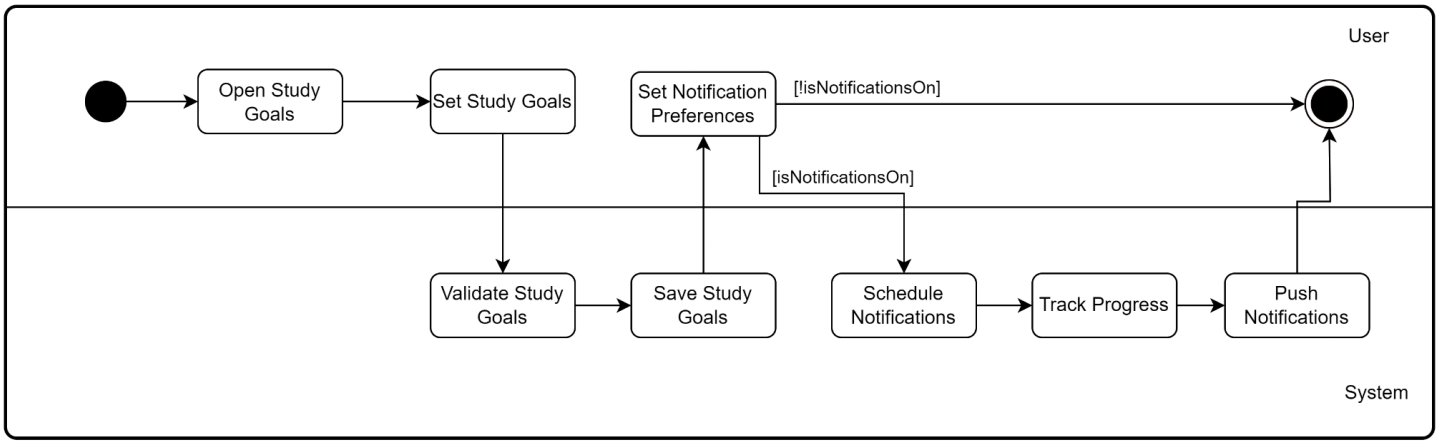


Fig. 10: Set Study Goals and Receive Notifications UML Activity Diagram

3.5.4.5. Solve Flashcard Using Spaced Repetition UML Activity Diagram

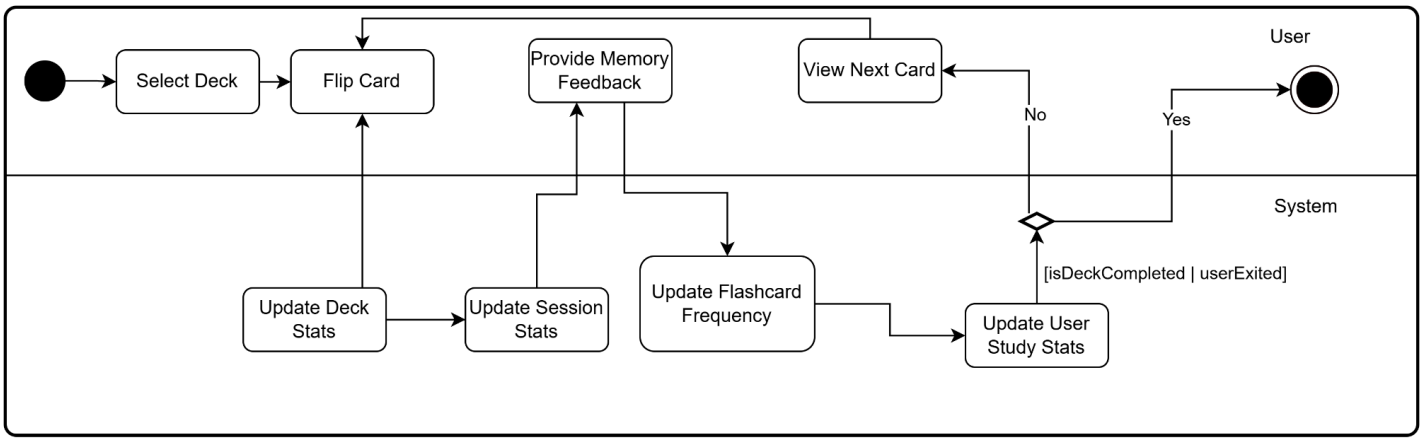


Fig. 11: Solve Flashcard Using Spaced Repetition UML Activity Diagram

3.5.4.6. Share Deck UML Activity Diagram

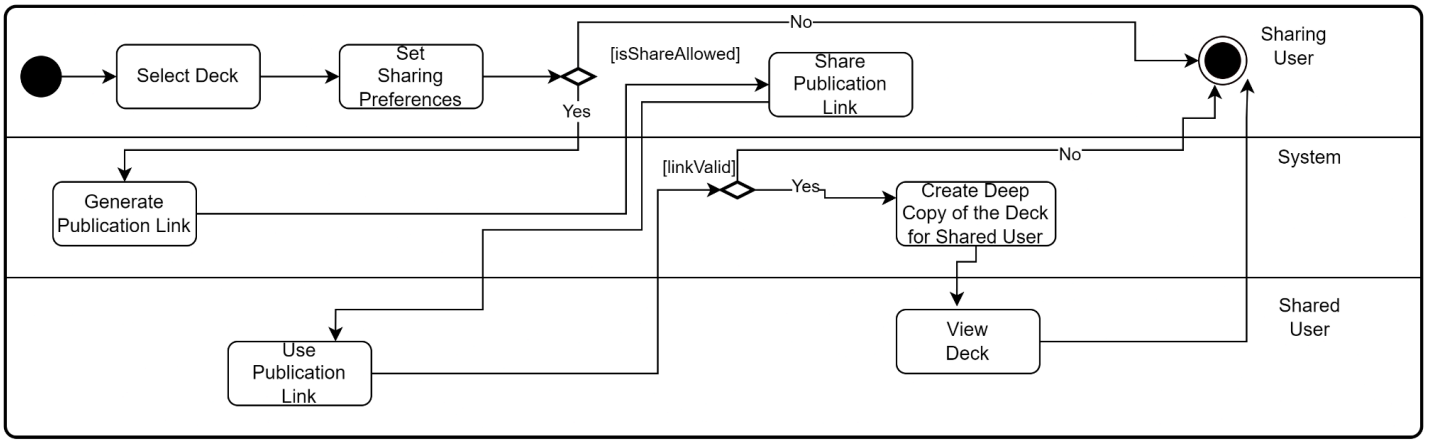


Fig. 12: Share Deck UML Activity Diagram

3.5.4.7. Flashcard Object UML State Diagram

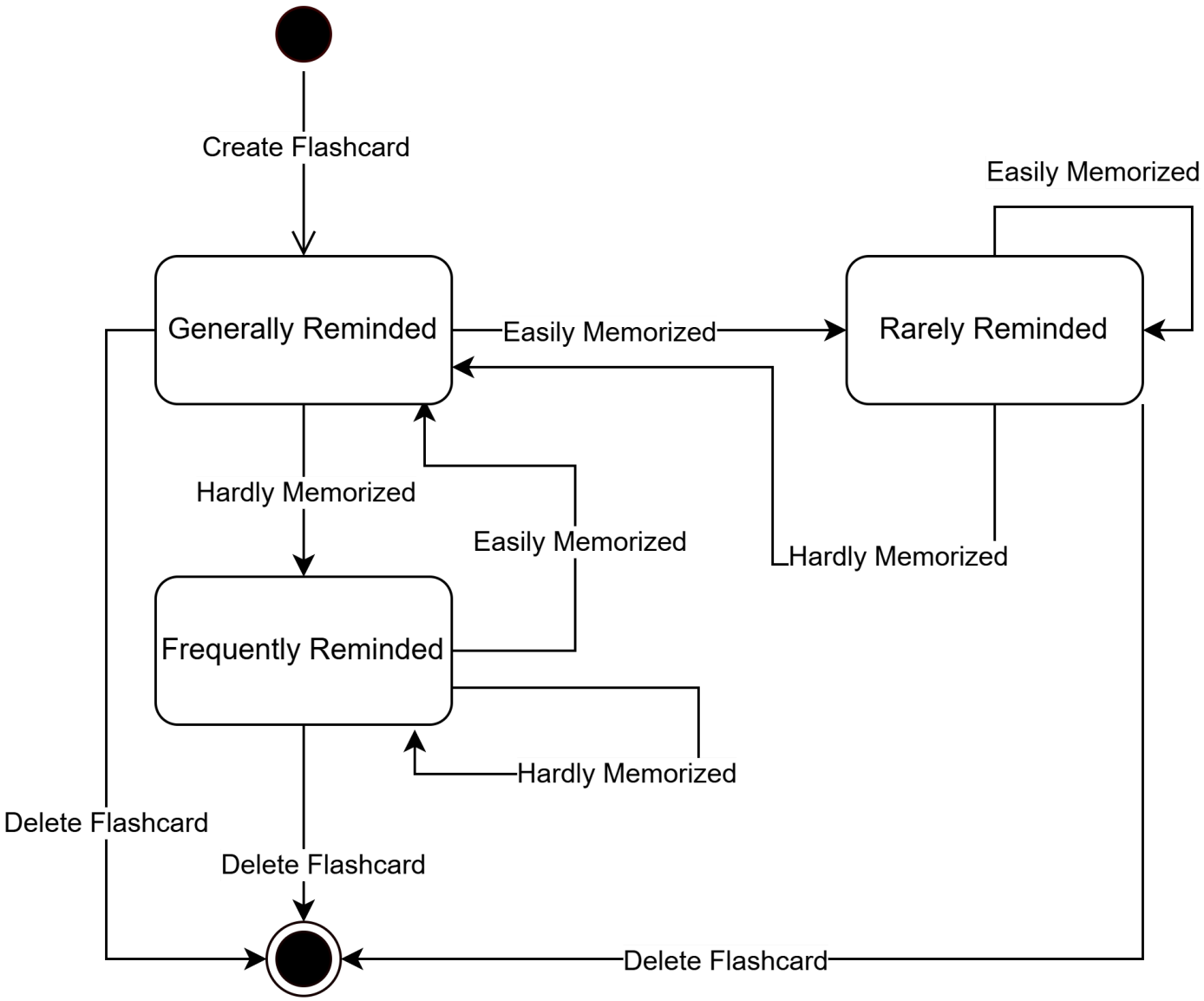


Fig. 13: Flashcard Object UML State Diagram

3.5.4.8. Flashcard Generation Pipeline UML State Diagram

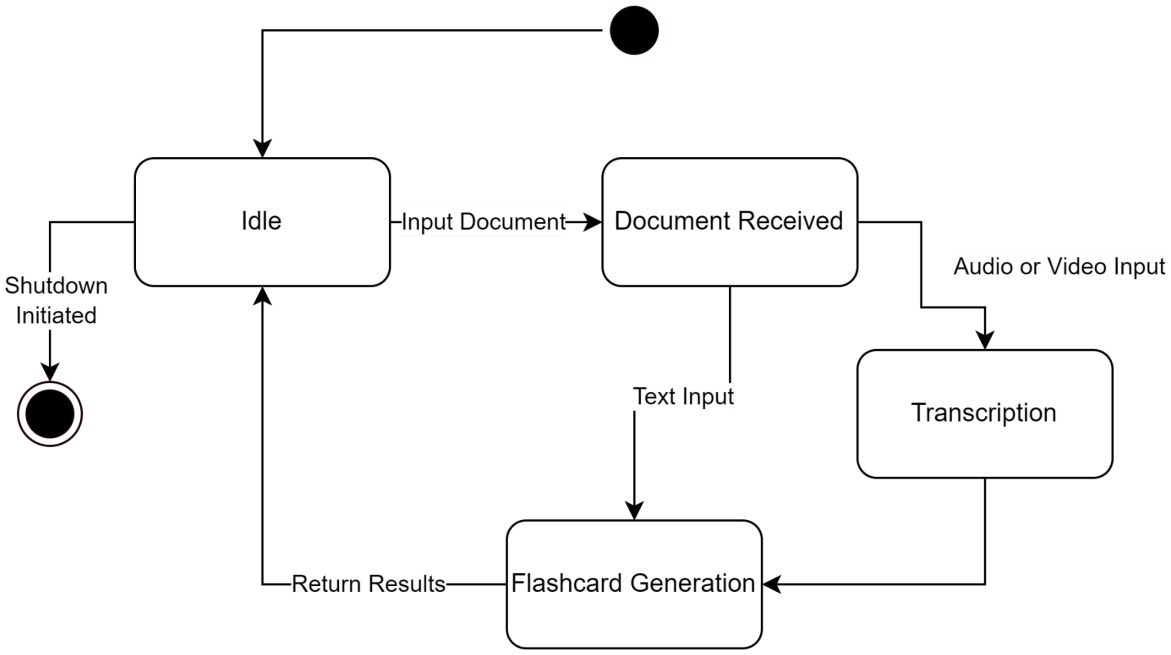


Fig. 14: Flashcard Generation Pipeline UML State Diagram

3.5.4.9. Question Object UML State Diagram

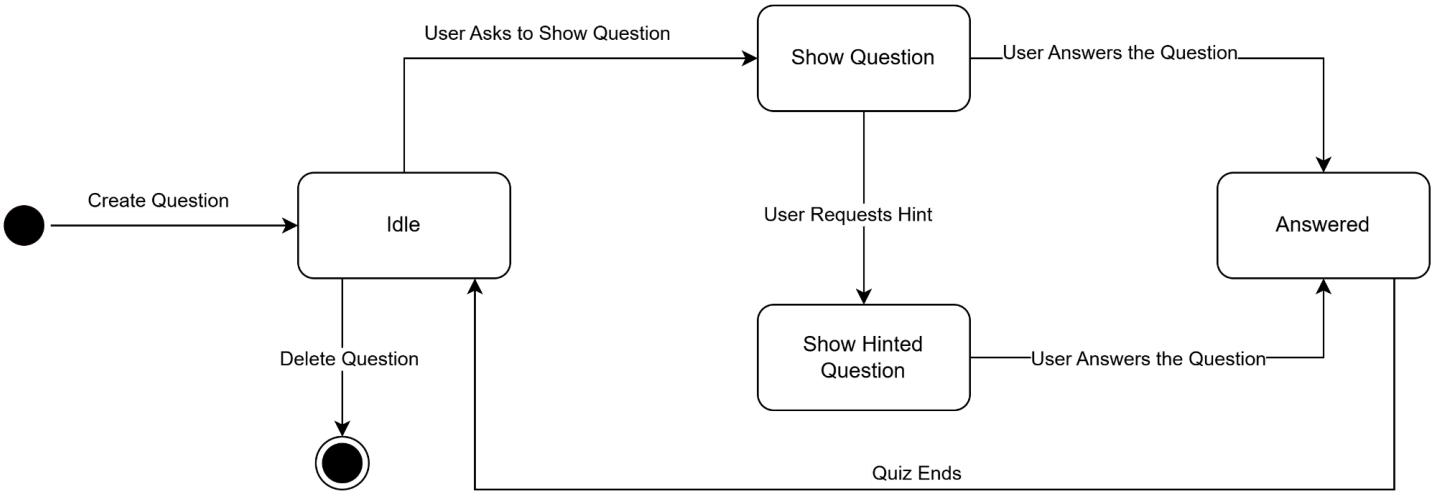


Fig. 15: Question Object UML State Diagram

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

3.5.5.1. Auth Pages

Consists of 2 pages: Register and Login. New users can register from the register screen in Figure 16 using their relevant information, which are username, email, and password, whereas existing users can use the login screen in Figure 17 with their email and password. Users can also use the Google registration service and use their Google accounts directly on the app.

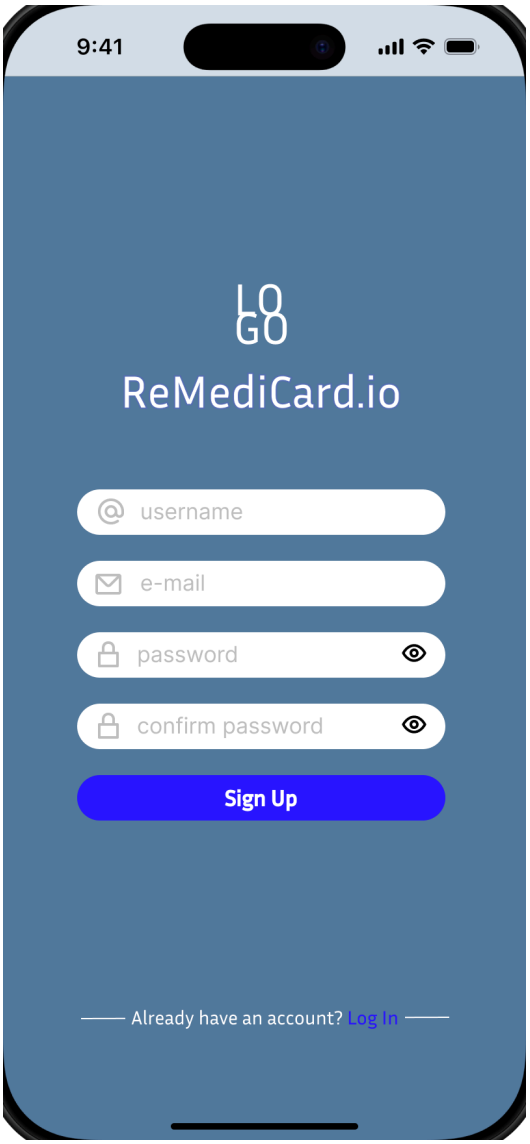


Fig. 16: Register Screen

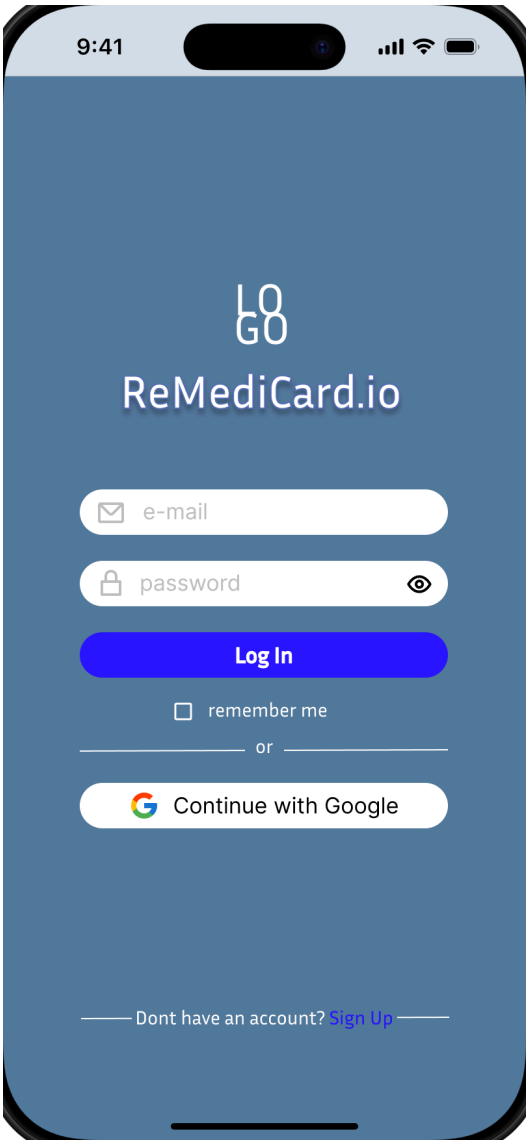


Fig. 17: Login Screen

3.5.5.2. Home Page

The home page is designed in a way that users can access every other page and feature from the home page. There is a reminder on the top of the page which reminds users about their study goals. From the Home page, users can access decks, quizzes, study goals dashboard, and deck/quiz creation and editing. Also, the navbar located towards the lower end of the screen ensures users can access the home page and their profile at any time.

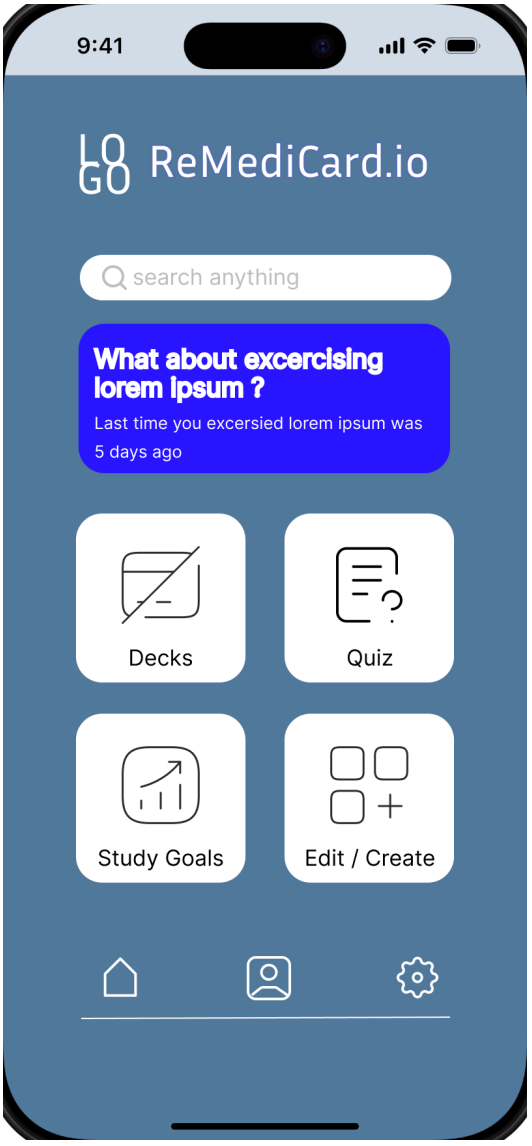


Fig. 18: Home Screen

3.5.5.3. Decks

The decks screen can be accessed from the home page and users can view different decks they have created and the information such as last access date, flashcard count, and their success rate. Users can search specific decks from the search bar located at the top of the screen. Users can also sort the decks according to the dates or success rate. In the decks screen, users can also switch between users own decks and shared decks from other users.

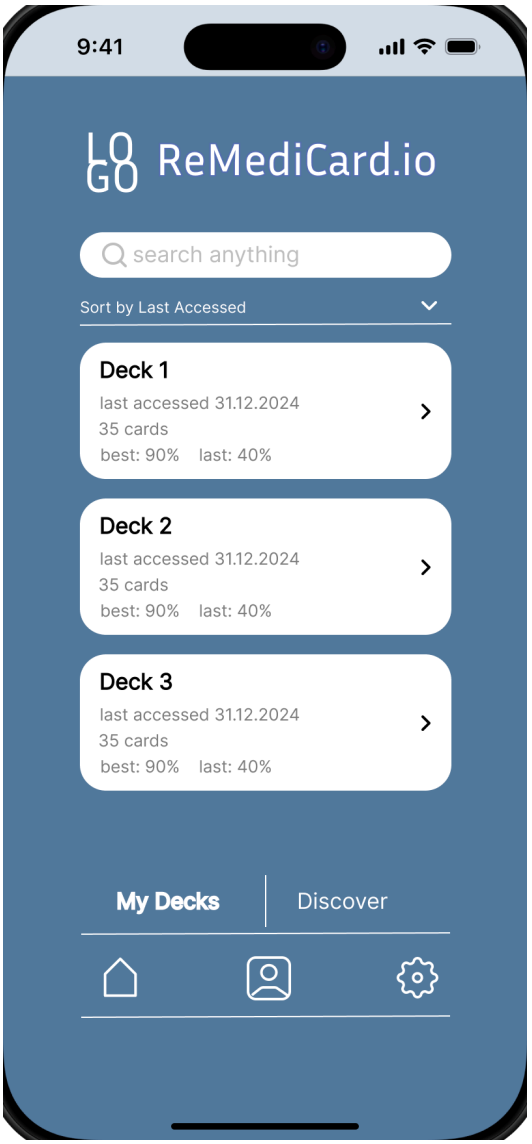


Fig. 19: Decks Screen

The decks consist of flashcards which can be seen in figure 20 and 21. Flashcards have two faces, one of them contains the question, and the other one contains a related answer. Users can touch the card to flip the other face of the card.
Users can mark the card as true or false according to whether they correctly answered or not. Correct and mistake numbers can be tracked as well as how many questions are left.

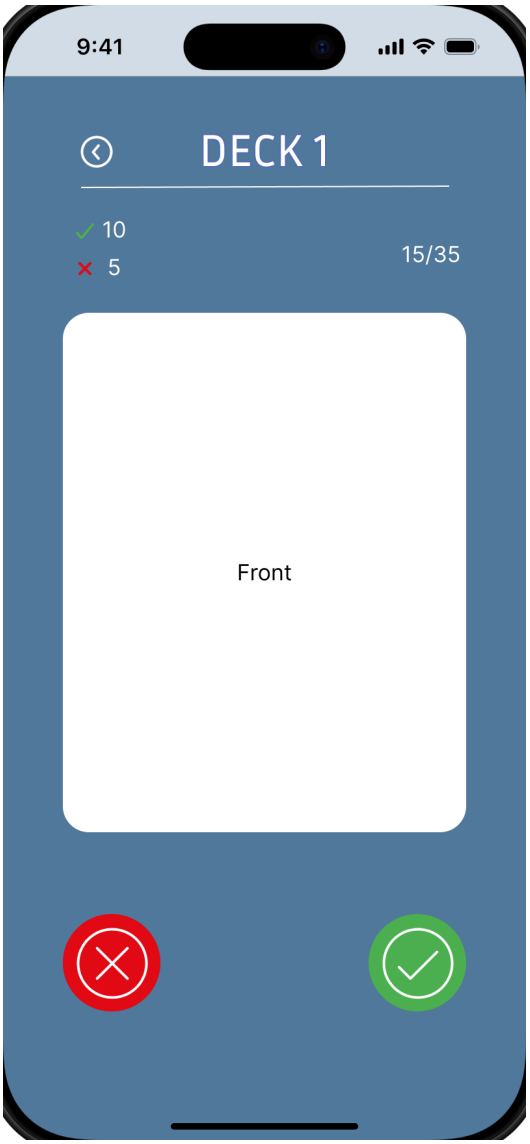


Fig. 20: Flashcard Front Face

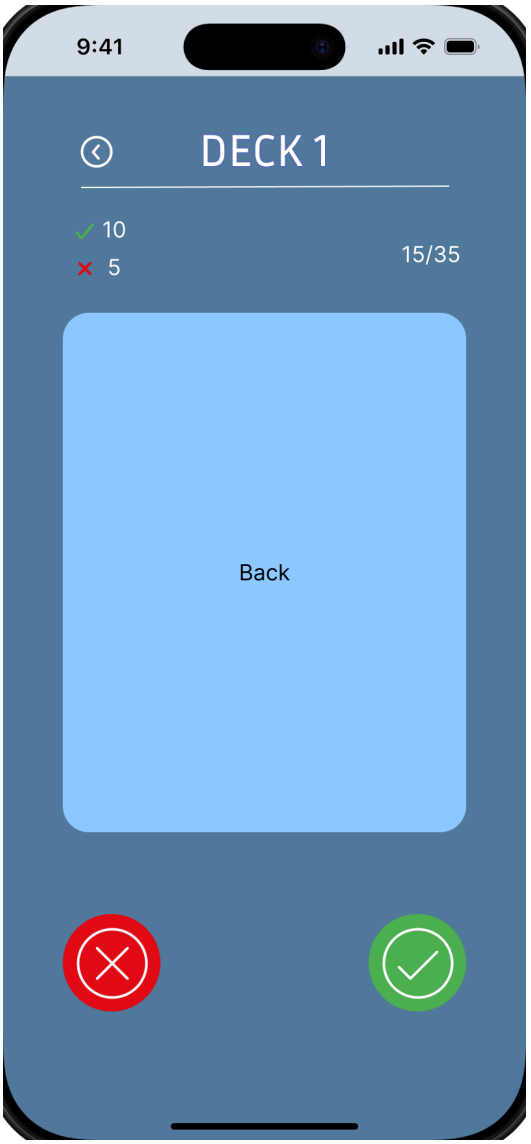


Fig. 21: Flashcard Back Face

3.5.5.4. Quizzes

The quizzes screen can be accessed from the home page, and users can view different quizzes they have created and the information such as last access date, flashcard count, and their success rate. Users can search specific quizzes from the search bar located at the top of the screen. Users can also sort the quizzes according to the dates or success rate. In the quizzes screen, users can also switch between users own quizzes and shared quizzes from other users.

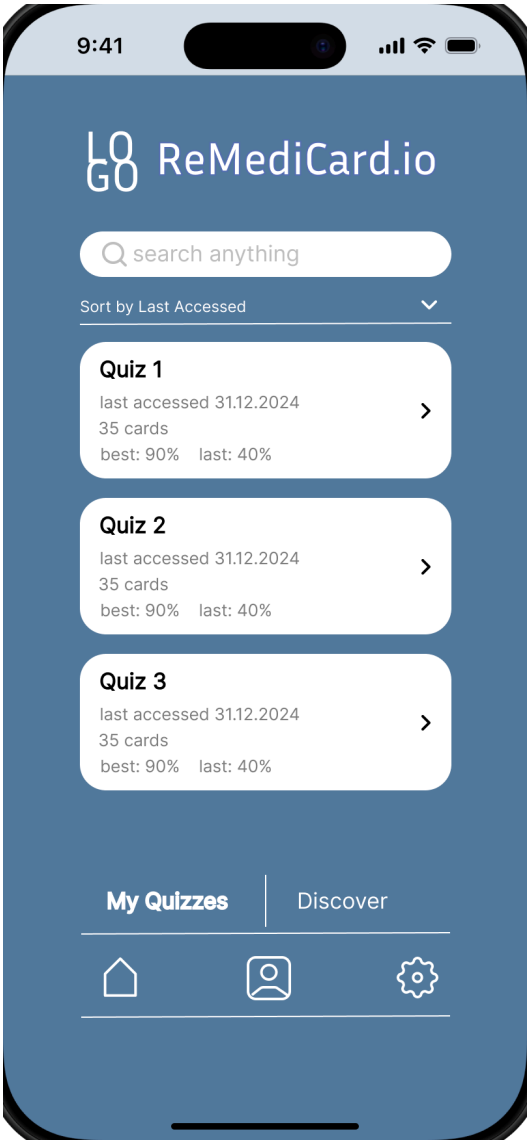


Fig. 22: Quizzes Screen

The quizzes consist of multiple choice questions which can be seen in figure 23. Users can select one choice and can move back and forth between questions using the next and previous buttons. Users can track how much time and questions are left.

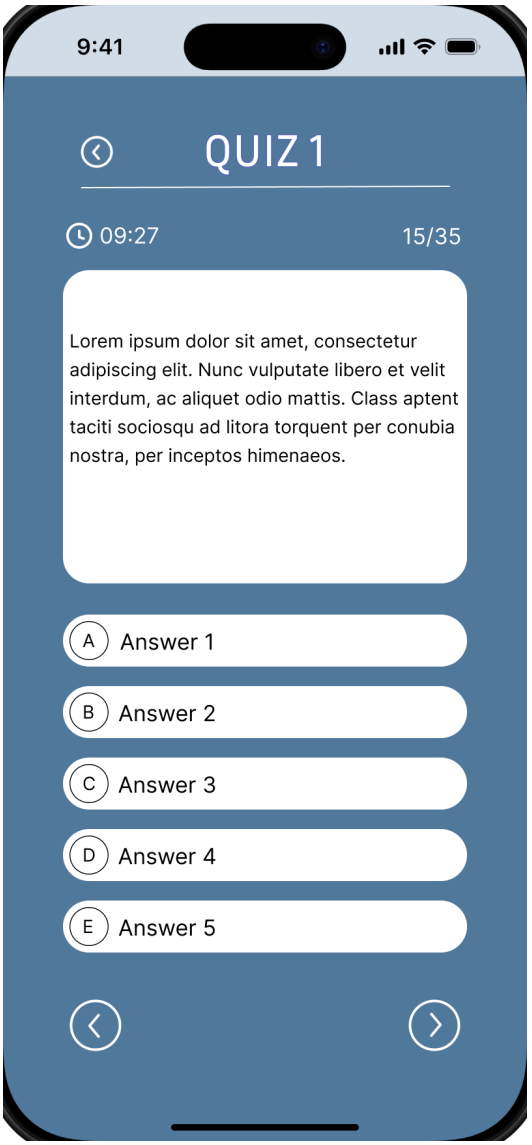


Fig. 23: Example Quiz Question

3.5.5.5. Profile Page

The profile page can be accessed from the navbar. On the profile page, users can manage their profile information and subscription plans. Users can also send a ticket to report an issue. According to the preference, language can be selected. Logging out from the account is also located on the profile page.

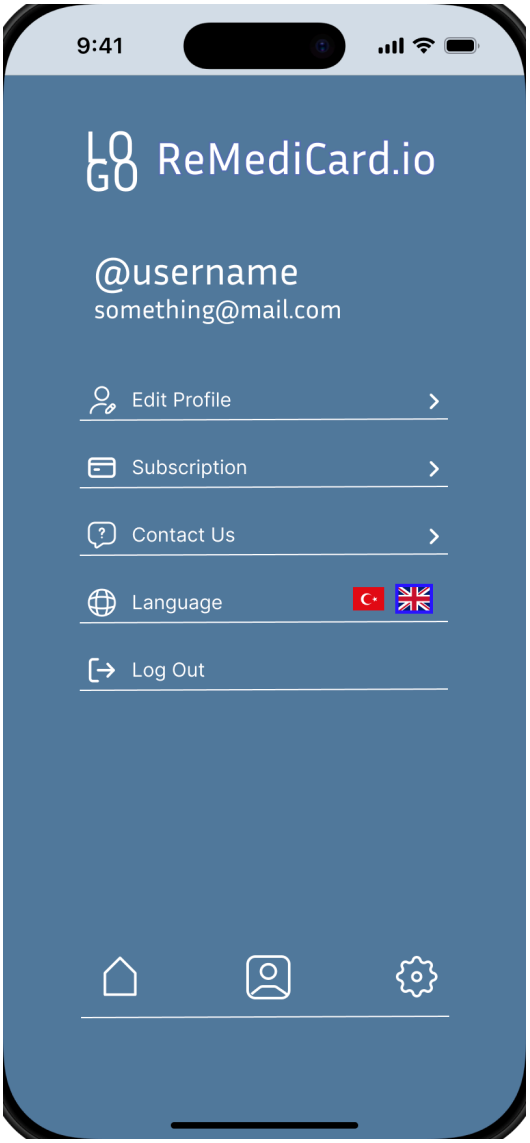


Fig. 24: Profile Page

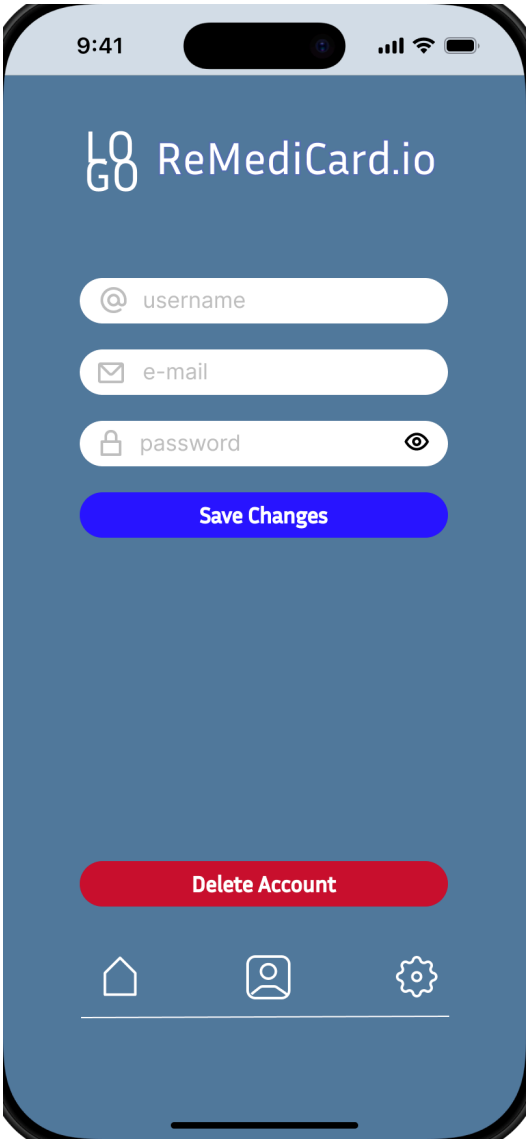


Fig. 25: Edit Profile Screen

3.5.5.6. Edit/Create

The edit/Create screen can be accessed from the home page. In this screen Users can create new decks or edit existing decks. Users can select existing decks to edit. In editing, the deck can be renamed. Every flashcard existing in the deck can be seen from here and they are editable. Additionally, new cards can be added manually or directly generated with AI

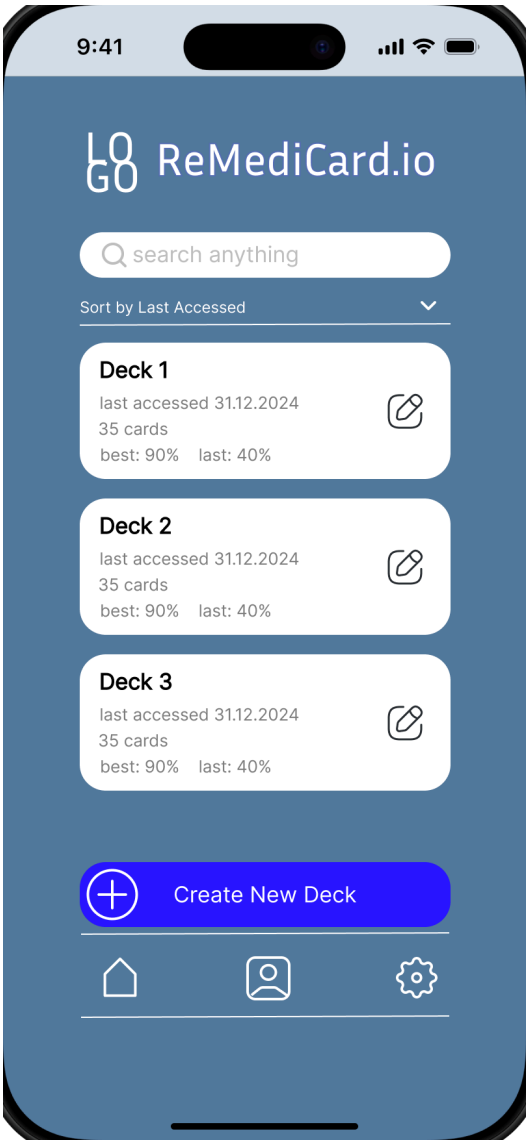


Fig. 26: Edit/Create Screen

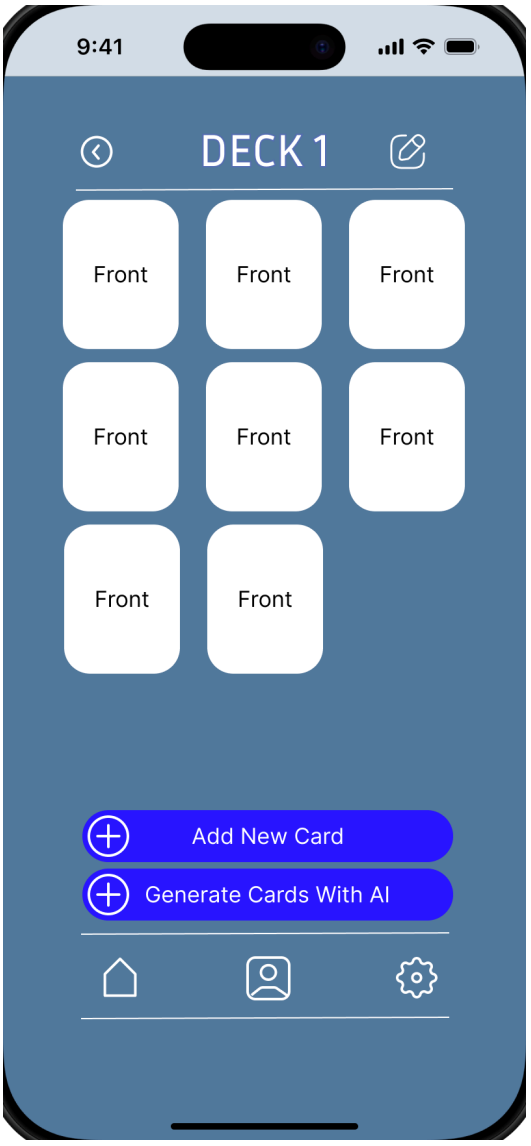


Fig. 27: Editing a Deck

Users can select one flashcard to edit. Users are able to edit both faces of the card. Further, users can delete the cards they want.

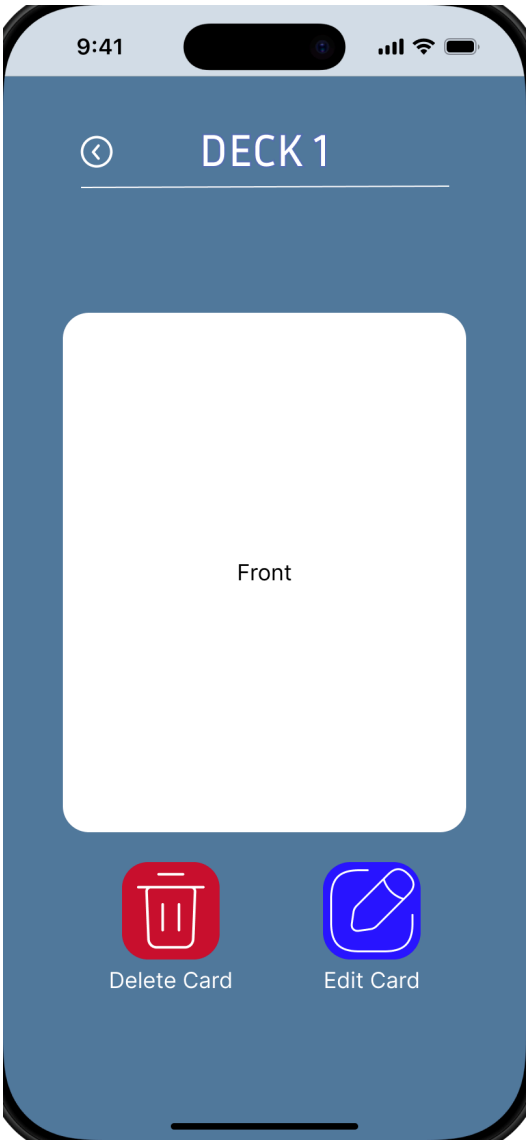


Fig. 28: Edit the front of the flashcard

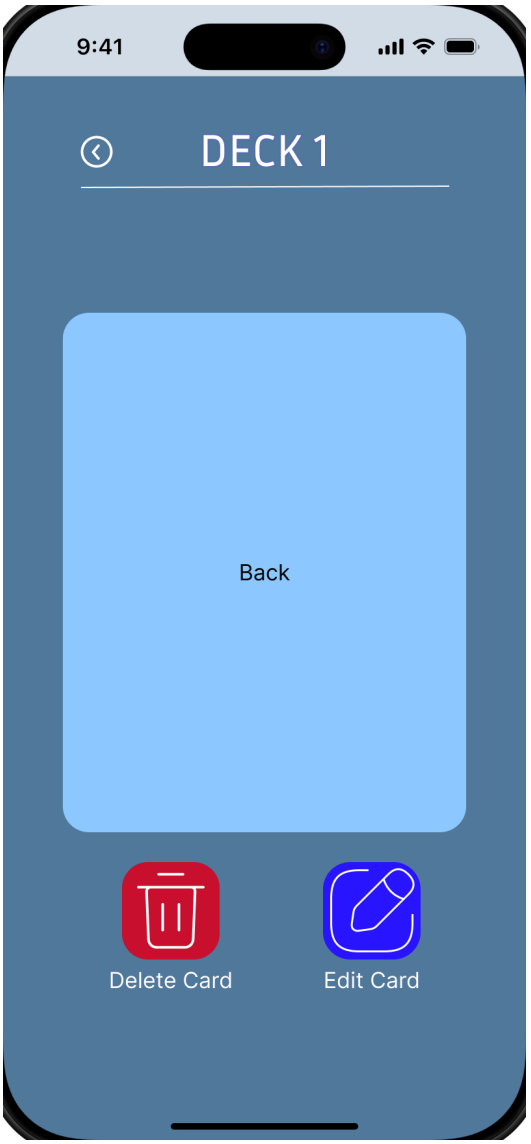


Fig. 29: Edit the back of the flashcard

Users can create new cards manually. Both faces of the new card should be filled with proper data manually.

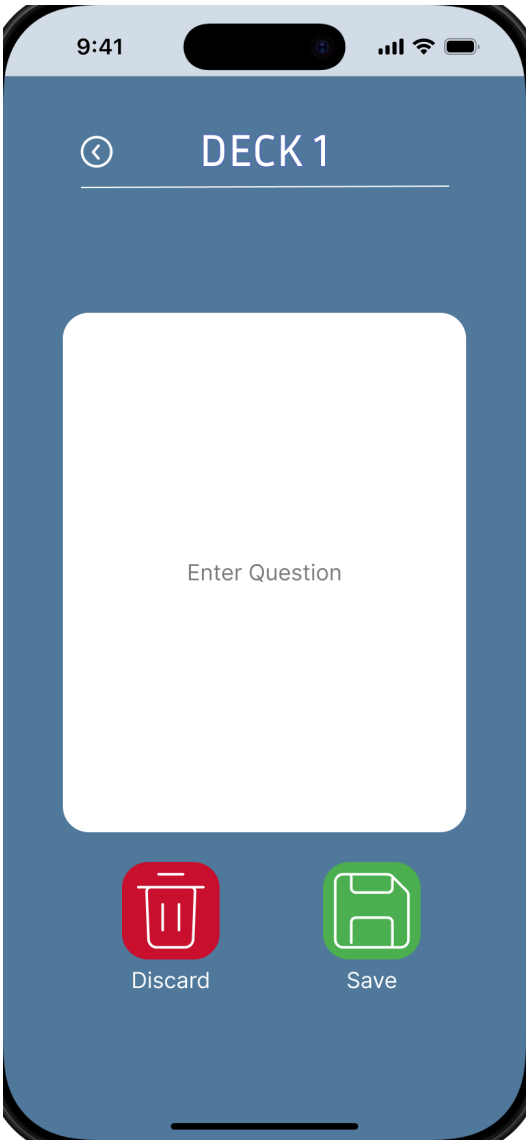


Fig. 30: Create front of a flashcard

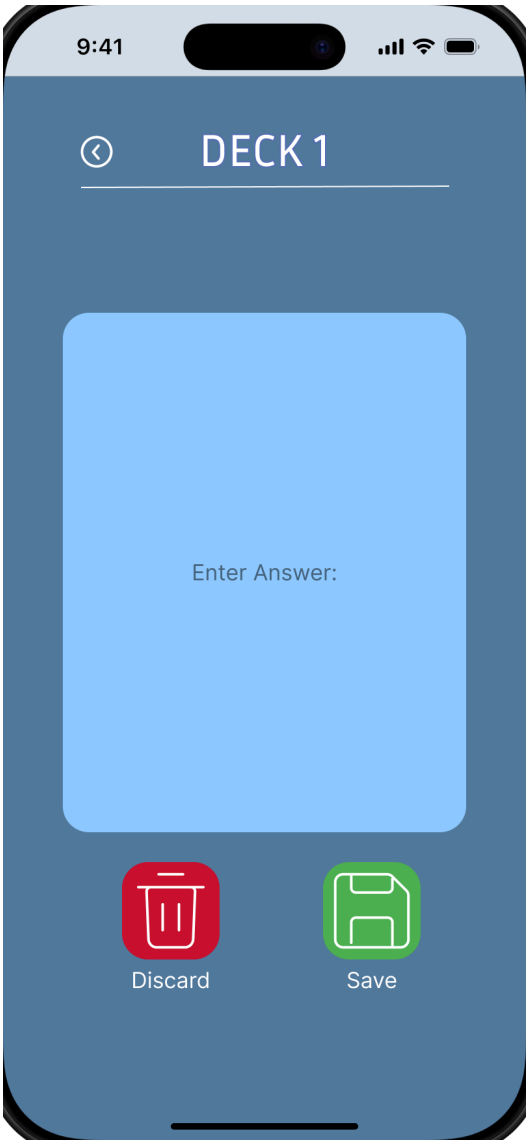


Fig. 31: Create back of a flashcard

Users can prefer AI to generate the flashcards. Users should upload a suitable file and the AI model will process the file. Estimated process time is informed to the user. After the generation, users can edit and delete the flashcards like they can do to manually created flashcards.

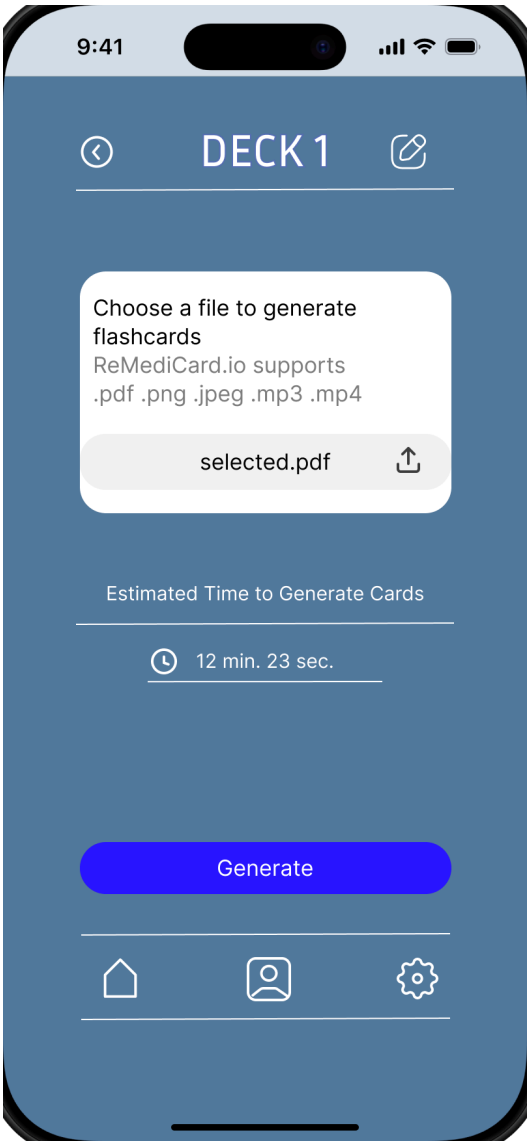


Fig. 32: Generate Flashcards with AI

3.5.5.7. Study Goals

Study Goals can be accessed from the home screen. Users can see their recommended study goals for specific decks according to their studies which can be seen in Figure 33, and directly create new goals via these recommendations.
Users can access active study goals from the study goals dashboard, this screen lists the goals that the user is actively pursuing. Here, they can see the duration of these targets, their expected performance if any, and when the goals start and end. Additionally, these goals are editable.

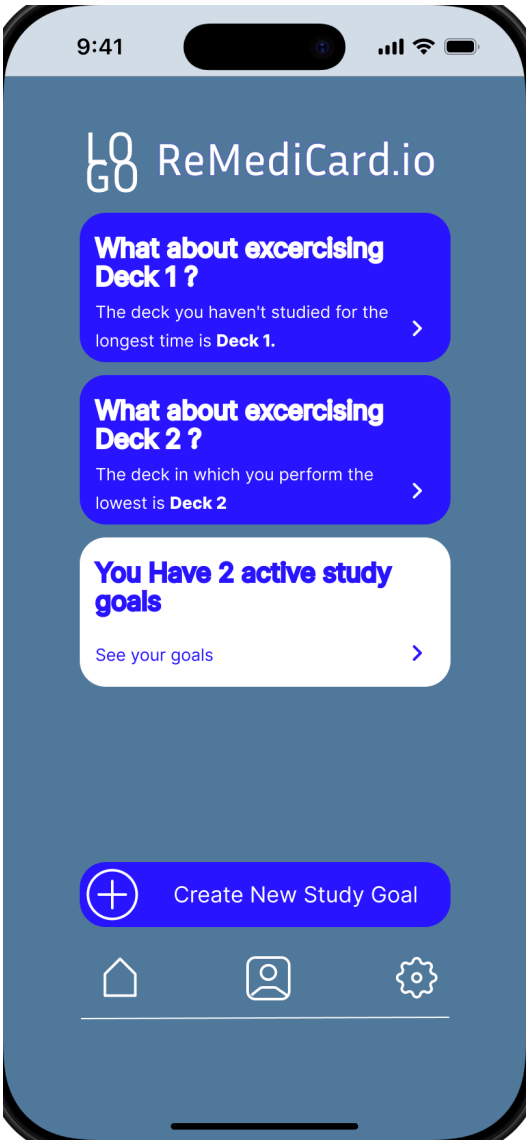


Fig. 33: Study Goals Dashboard

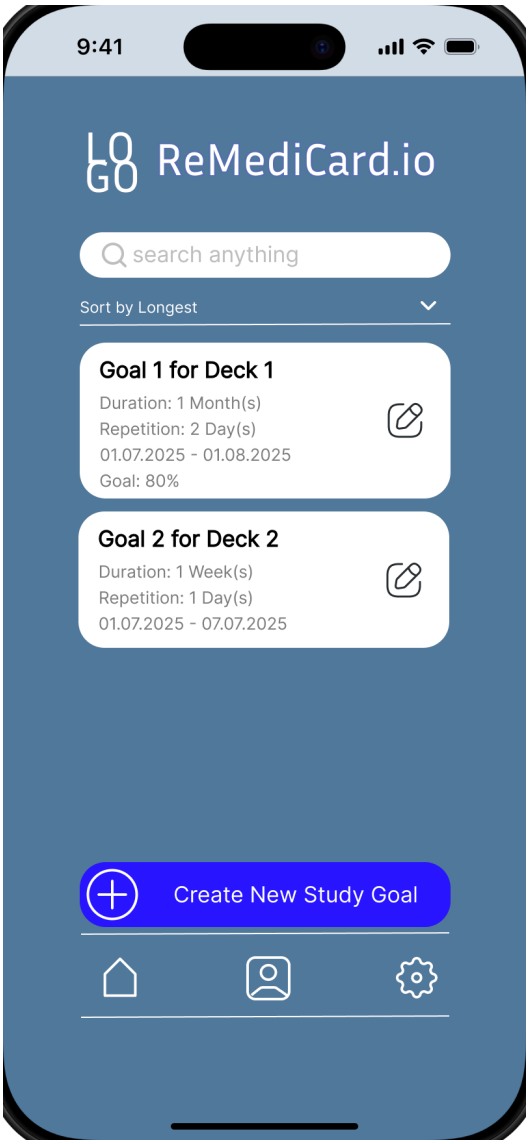


Fig. 34: Active Study Goals

Users can create new study goals manually by filling in the required information. Users need to select a deck to create a new goal on. Also, the duration and repetition of this goal need to be selected. Users can enter a performance score optionally. Created study goals will send notification about the goal to the user according to their specifications.

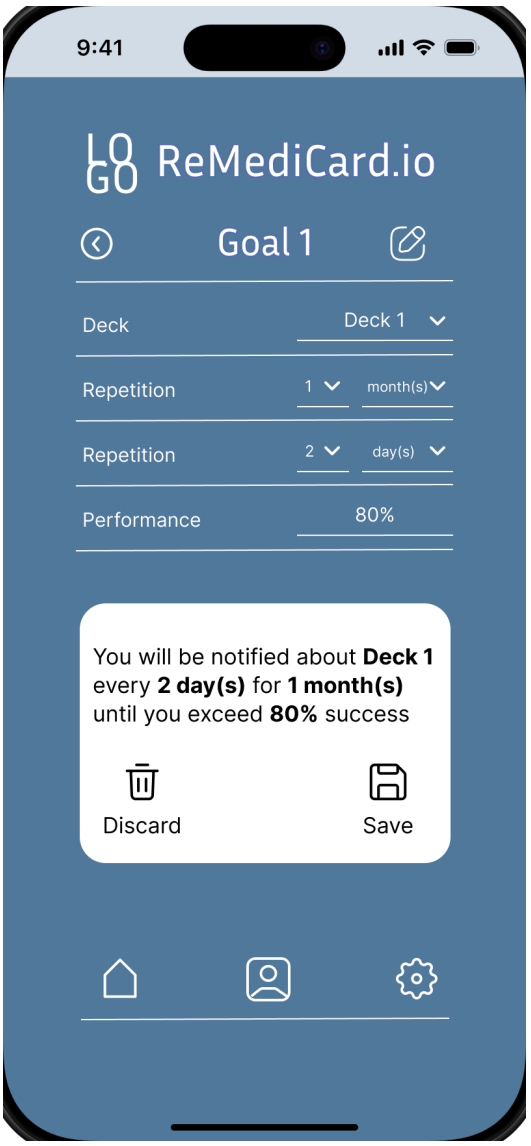


Fig. 35: Create New Goal with performance tracking

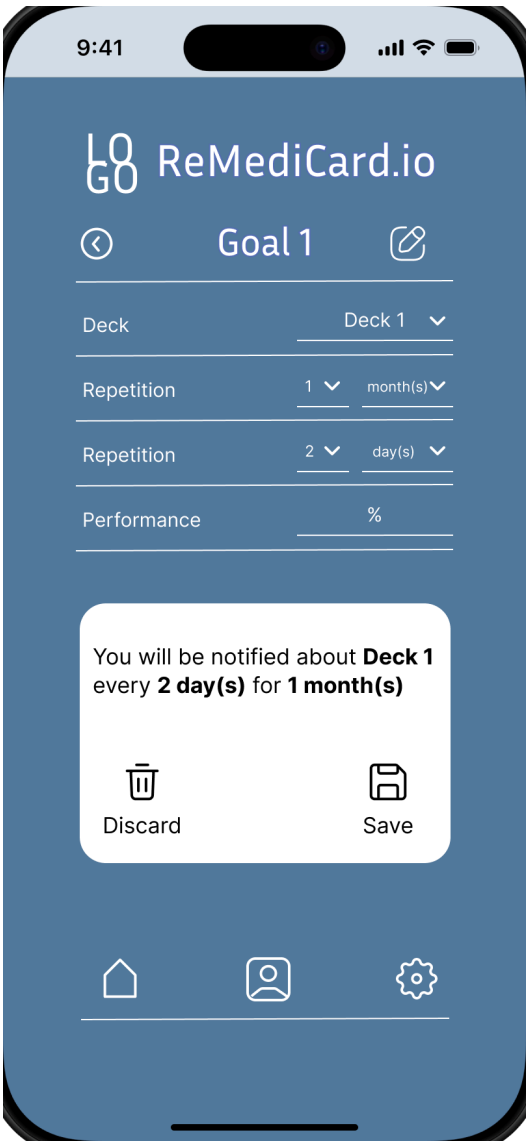


Fig. 36: Create New Goal without performance tracking

4. Other Analysis Elements

4.1. Consideration of Various Factors in Engineering Design

4.1.1. Constraints

We have defined some limitations within which the application must operate. Ensuring that our resources, technical capabilities, and ethical concerns align with the project goals and external requirements.

4.1.1.1. Implementation Constraints

Technological Requirements:

- The integration of machine learning models and computer vision tools requires robust frameworks such as TensorFlow, PyTorch, and OpenCV, along with substantial backend development for data processing.
- The AI model must handle various data types, including image, audio, and text, and provide similar output for each.

Computational Resources:

- High computational power is necessary for processing images, running ML algorithms, and transcribing audio to text. This may require the use of adequate cloud-based services.

Data Privacy:

- To protect user data and securely manage sensitive information, such as lecture videos and personal notes, it is essential to implement robust encryption and comply with established data protection protocols.

Platform Compatibility:

- The application needs to work across multiple mobile platforms (e.g., iOS, Android), leading to a need for cross-platform development tools and technologies.

4.1.1.2. Economic Constraints

Development Costs:

- Leveraging libraries or APIs for advanced functionality, such as speech-to-text conversion or specialized ML tools, could lead to licensing fees.
- There could be lease fees involved for the use of medical books, presentations, sample questions, and any other resources.

Server and Infrastructure Costs:

- Running AI models and storing multimedia data (images, audio, and video) may involve significant server costs, especially for real-time processing and storage.

Maintenance and Updates:

- Continuous updates, bug fixes, and model retraining to keep up with the latest advances in AI and maintain accuracy will require further funding.

4.1.1.3. Ethical Constraints

Bias and Fairness:

- Machine learning models might introduce biases if not properly trained on a diverse dataset.
- Generated flashcards and questions should be checked to ensure fairness and inclusivity.

User Consent:

- Users should be fully informed about data collection and use. User data can only be collected after the user consents to it.
- Consent for processing audio and image data is crucial, and transparency regarding data storage and privacy should be maintained.

Intellectual Property:

- Care must be taken to avoid infringing on proprietary medical content and copyrighted materials when generating content.
- Data uploaded by the users should be checked by administrative personnel to prevent any abuse.

Accuracy and Reliability:

- Providing students with accurate information is crucial; incorrect or misleading content could have serious implications for their education and future medical practice.
- Professional help from the medical field is required to check the content used in the application.

Mental Health Considerations:

- A balanced approach should be maintained to prevent stress and cognitive overload in students by not overwhelming them with difficult questions or excessive repetitions.

4.1.1.4. Global Factors

The application should be fit for a global audience with varying medical curricula and study practices used. Multi-language support is essential for students across the globe. This can be done with the help of contributors all around the world or with the use of natural language processing tools.

4.1.1.5. Cultural Factors

Cultural differences in medical practices and terminologies should be respected. Generated content could be reviewed by an Admin before it is deemed ready for use.

4.1.1.6. Social Factors

Features like sharing decks and quizzes could help build a collaborative community, enabling peer-to-peer learning that contributes to each and every student. The application should be accessible by students from varying socioeconomic backgrounds.

4.1.1.7. Environmental Factors

Cloud-based infrastructure with on-demand resource usage should reduce energy consumption compared to traditional methods. By digitizing study materials and studying sessions, we could contribute to the reduction of paper use.

4.1.1.8. Economic Factors

The base app will be free to use. The premium version of the application will offer affordable pricing to obtain unlimited use of the features offered. Leveraging advanced AI models requires significant use of resources but promises high returns in terms of educational outcomes.

Table 1: Factors that can affect analysis and design.

	Effect Level	Effect
Public health	7/10	Enhances medical education by improving study efficiency, indirectly contributing to better-trained healthcare professionals.
Public safety	2/10	Ensures accurate content delivery, reducing risks associated with misinformation in medical training, which could impact patient safety in later stages.

Public welfare	7/10	Offers equal access to educational tools, empowering students from diverse backgrounds and improving the quality of future medical practitioners globally.
Global factors	4/10	Adapts to diverse curricula and languages, promoting inclusivity and incentivizing global collaboration among medical students.
Social factors	8/10	Builds a collaborative community through sharing decks, quizzes and peer learning.
Environmental factors	5/10	Encourages sustainability by reducing paper usage and leveraging energy-efficient on-demand cloud resources for AI processing and storage.
Economic factors	8/10	Balances cost-efficiency with affordability, providing a free base app and a premium version for advanced features.

4.1.2. Standards

ReMediCard.io's standards cover user interaction, data handling, and software development:

4.1.2.1. Software Engineering Standards

- **IEEE 830-1998**: Used for documenting requirements specifications [5].
- **IEEE 1016-2009**: Used for system design description [6].
- **UML 2.5.1**: Used for system modeling, including use-case diagrams, class diagrams, activity diagrams, and state diagrams [7].

4.1.2.2. Data Security and Privacy Standards

- **ISO/IEC 27001**: Will be used for information security management to protect sensitive data and prevent unauthorized access [8].
- **GDPR Compliance**: Will be used to ensure the system is designed to meet European data protection standards for handling user data [4].

4.1.2.3. AI/ML Development Standards

- **IEEE 7001-2021**: Will be used for ethical considerations in AI design and development [9].

4.1.2.4. Coding Standards

- **Google Style Guide (Java)**: Will be used for Java development [10].
- **RESTful API Design Standards**: Will be used for integration and communication between system components.

4.1.2.5. Testing Standards

- **ISO/IEC/IEEE 29119**: Will be used for software testing to ensure reliability for testing of application features [11].

4.2. Risks and Alternatives

Risk	Likelihood	Effect on Project	B Plan
Integration of AI models (e.g., NLP, speech-to-text) produces inaccurate results.	High	Poor flashcard/question quality; user dissatisfaction.	Include manual verification or feedback mechanisms for generated content; retrain models using more robust datasets.
Application usage exceeds the limits	Moderate	The duration of the automated creation of flashcards increases.	Implement load balancing, and redundancy mechanisms, and conduct stress testing before launch.
High computational demand for AI processing and data handling overwhelms servers.	Moderate	Delays in generating flashcards from multimedia inputs; system performance drops.	Use scalable cloud-based solutions (e.g., AWS Auto-scaling) and optimize backend processing pipelines.

Users might want to use the app through computers	Low	The user count does not increase as expected.	Send notifications, and emails to the users that a web application will be coming soon.
Users find the application interface too complex to use.	Low	The user count does not increase as expected.	Conduct usability testing early; redesign UI/UX based on user feedback.
Data breaches compromise user data.	Low	Legal, ethical, and reputational damages.	Ensure data encryption and compliance with GDPR; conduct security audits regularly.

4.3. Project Plan

Work Package	Work Package Title	Leader	Members Involved
WP#1	Jira and GitHub Project Management Infrastructure	Faruk Uçgun	Akif Erdem Tanyeri, Ömer Fırat Bekiroğlu
WP#2	Full Stack Project Setup	Cenk Merih Olcay	Enes Bektaş, Faruk Uçgun
WP#3	User Authentication, Profiles and Account Development	Cenk Merih Olcay	Faruk Uçgun
WP#4	General UI Design	Ömer Fırat Bekiroğlu	Akif Erdem Tanyeri
WP#5	Platform Main Page, Account and Profile UI Development	Ömer Fırat Bekiroğlu	Akif Erdem Tanyeri
WP#6	Card Deck Creation and Editing Development	Enes Bektaş	Faruk Uçgun
WP#7	Deck and Flashcard Creation and Management Pages UI Development	Ömer Fırat Bekiroğlu	Akif Erdem Tanyeri
WP#8	Deck Sharing and Flashcard Media Attachment Feature Development	Cenk Merih Olcay	Enes Bektaş
WP#9	Summarizer and Flashcard Generator LLM Models Research	Enes Bektaş	Cenk Merih Olcay, Ömer Fırat Bekiroğlu

WP#10	Spaced Repetition and Study Tracking Development	Faruk Uçgun	Enes Bektaş, Cenk Merih Olcay
WP#11	AI-Generated Text Summaries and Flashcards Model Development	Enes Bektaş	Ömer Fırat Bekiroğlu
WP#12	Flashcard Generation from Lecture Notes Feature Development	Cenk Merih Olcay	Faruk Uçgun
WP#13	Study Dashboard, Goal Notifications, Statistics UI Development	Ömer Fırat Bekiroğlu	Akif Erdem Tanyeri, Cenk Merih Olcay
WP#14	Video/Voice Record Processing Development	Enes Bektaş	Cenk Merih Olcay
WP#15	AI Model Integration with Backend Deck/Flashcard Services	Faruk Uçgun	Enes Bektaş
WP#16	Quizzes and Questions Features UI Development	Akif Erdem Tanyeri	Ömer Fırat Bekiroğlu
WP#17	Quizzes and Questions Features Development	Faruk Uçgun	Enes Bektaş
WP#18	Dockerization and Microservice Architecture	Faruk Uçgun	Cenk Merih Olcay
WP#19	Case Study AI Model Development	Enes Bektaş	Akif Erdem Tanyeri
WP#20	Case Study Feature Development	Enes Bektaş	Faruk Uçgun
WP#21	Case Study UI Development	Akif Erdem Tanyeri	Ömer Fırat Bekiroğlu

WP#1: Jira and GitHub Project Management Infrastructure
Leader: Faruk Uçgun
Members Involved: Akif Erdem Tanyeri, Ömer Fırat Bekiroğlu
Start Date: 10/10/2024 **End Date:** 20/10/2020
Objective: To initialize project management infrastructure using Jira and Github to ensure project tracking and collaboration
Tasks:

- **T1.1:** Setup Jira environment
- **T1.2:** Setup Github organization and repository infrastructure
- **T1.3:** Setup project website hosting with GitHub Pages

Deliverables:

- **D1.1:** Jira Board
- **D1.2:** Github repository
- **D1.3:** Project website

WP#2: Full Stack Project Setup
Leader: Cenk Merih Olcay
Members Involved: Enes Bektaş, Faruk Uçgun
Start Date: 18/10/2024 **End Date:** 30/10/2024
Objective: Creation of the default Front-end & Back-end projects and Database setups for development
Tasks:

- **T2.1:** Create a React Native project with Expo and Android Studio
- **T2.2:** Create a Spring Boot project
- **T2.3:** Create MySQL and MongoDB containers using Docker

Deliverables:

- **D2.1:** React Native Project Source Code
- **D2.2:** Spring-Boot Project Source Code
- **D2.3:** Dockerfile for database containers

WP#3: User Authentication, Profiles and Account Development
Leader: Cenk Merih Olcay
Members Involved: Faruk Uçgun
Start Date: 18/10/2024 **End Date:** 30/10/2024
Objective: Implement the features for user accounts, authentication/authorization, and account settings
Tasks:

- **T3.1:** Define the classes and their relations to the accounts in the system
- **T3.2:** Implement a secure user registration and login functionality
- **T3.3:** Implement functionality for users to view, edit, and update their profiles
- **T3.4:** Design and implement role-based permissions for different user roles

Deliverables:

- **D3.1:** Class Diagram
- **D3.2:** Backend Authentication System
- **D3.3:** User profile management functionalities
- **D3.4:** Role-Based Authorization

WP#4: General UI Design
Leader: Ömer Fırat Bekiroğlu
Members Involved: Akif Erdem Tanyeri

Start Date: 18/11/2024 **End Date:** 12/12/2024
Objective: Design a visually appealing, simple, and useful user interface that enhances user experience
Tasks:
- **T4.1:** Draw simple design to decide on UI design approaches
- **T4.2:** Create mockups for core features

Deliverables:
- **D4.1:** UI Mockups

WP#5: Platform Main Page, Account, and Profile UI Development
Leader: Ömer Fırat Bekiroğlu
Members Involved: Akif Erdem Tanyeri
Start Date: 18/10/2024 **End Date:** 30/10/2024
Objective: Implement the design of the home page, account, and profile-related pages following designed mockups
Tasks:
- **T5.1:** Implement the home page for the application
- **T5.2:** Implement account-related UI elements for login, registration, and account settings
- **T5.3:** Implement profile-related UI elements for editing profile and user preferences
- **T5.4:** Integrate UI with Back-end Endpoints

Deliverables:
- **D5.1:** Home Page UI
- **D5.2:** Login Page UI
- **D5.3:** Registration Page UI
- **D5.4:** Profile Page UI
- **D5.5:** Profile Settings UI

WP#6: Card Deck Creation and Editing Development
Leader: Enes Bektaş
Members Involved: Faruk Uçgun
Start Date: 05/12/2024 **End Date:** 16/12/2024
Objective: Design and development of the deck and flashcard management features in the backend
Tasks:
- **T6.1:** Update the Class Diagram to involve the relevant Classes and Relations
- **T6.2:** Implementations of the features in the backend using Spring Data Library to create Entities, Repositories, and relevant service methods based on the design

Deliverables:
- **D6.1:** Updated Class Diagram
- **D6.2:** Back-end Source Code for New Services, Entities, and Repositories

WP#7: Deck and Flashcard Creation and Management Pages UI Development
Leader: Ömer Fırat Bekiroğlu
Members Involved: Akif Erdem Tanyeri
Start Date: 12/12/2024 **End Date:** 12/01/2025
Objective: Implement the design of flashcards and flashcard decks with their interactions and management pages

Tasks:

T7.1: Implement the design of a single flashcard and its edit and create pages as well as interactions of these designs

T7.2: Implement the design of a flashcard deck and its edit and create pages as well as interactions of these designs

T7.3: Integrate UI with Back-end Endpoints

Deliverables:

D7.1: Flashcard UI Source Code

D7.2: Edit Flashcard UI Source Code

D7.3: Create Flashcard UI Source Code

D7.4: Flashcard Deck UI Source Code

D7.5: Edit Flashcard Deck UI Source Code

D7.6: Create Flashcard Deck UI Source Code

WP#8: Deck Sharing and Flashcard Media Attachment Feature Development
Leader: Cenk Merih Olcay
Members Involved: Enes Bektaş
Start Date: 26/12/2024 **End Date:** 16/01/2025
Objective: Development of the shared-deck features between the users in the backend
Tasks:

T8.1: Update class diagram to cover shared decks and new related classes and relations

T8.2: Implement the features by updating the related backend services

Deliverables:

D8.1: Updated Class Diagram

D8.2: Updated Source Code for the Back-end Services Responsible for Decks and Flashcards

WP#9: Summarizer and Flashcard Generator LLM Models Research
Leader: Enes Bektaş
Members Involved: Cenk Merih Olcay, Ömer Fırat Bekiroğlu
Start Date: 26/12/2024 **End Date:** 15/01/2025
Objective: Finding suitable methods and data sets to fine-tune fundamental models for the summarization and text generation capabilities in medicine
Tasks:

T9.1: Research on the performance of fundamental models in the medical text

T9.2: Finding relevant datasets to optimize the fundamental models

T9.3: Searching for ways to enhance the optimization with different means of hyperparameter tuning

Deliverables:

D9.1: Medical Text Datasets

D9.2: Report on the performance of state of art fundamental models

D9.3: Report on the candidate methods to optimize the fine-tuning

WP#10: Spaced Repetition and Study Tracking Development
Leader: Faruk Uçgun
Members Involved: Enes Bektaş, Cenk Merih Olcay
Start Date: 26/12/2024 **End Date:** 15/01/2025
Objective: Find a suitable spaced repetition algorithm and implement the study

tracking by developing an existing spaced repetition algorithm
Tasks:

T10.1: Conduct research on spaced repetition algorithms
T10.2: Implement spaced repetition algorithm
T10.3: Develop study tracking functionality using a spaced repetition algorithm

Deliverables:

D10.1: Spaced repetition algorithm
D10.2: Study tracking functionality

WP#11: AI-Generated Text Summaries and Flashcards Model Development
Leader: Enes Bektaş
Members Involved: Ömer Fırat Bekiroğlu
Start Date: 20/01/2025 **End Date:** 30/01/2025
Objective: Fine-tune the fundamental models using the obtained reports, dataset, and methods from WP#9
Tasks:

T11.1: Choosing the target fundamental models from the reports
T11.2: Preparing the datasets for the training
T11.3: Training the target multiple models in multiple iterations with different methods
T11.4: Creating the services for inference

Deliverables:

D11.1: Fine-tuned AI Models
D11.2: Hosted AI Model Services for Inference

WP#12: Flashcard Generation from Lecture Notes Feature Development
Leader: Cenk Merih Olcay
Members Involved: Faruk Uçgun
Start Date: 30/01/2024 **End Date:** 09/02/2025
Objective: Implement the backend services and image processing service to extract the labels out of figure images and create the flashcards/puzzles through them
Tasks:

T12.1: Creating an OCR service for text extraction
T12.2: Creating backend services for figure flashcards

Deliverables:

D12.1: OCR Service Source Code
D12.2: Source Code of Back-end Services for figure-based flashcard management

WP#13: Study Dashboard, Goal Notifications, Statistics UI Development
Leader: Ömer Fırat Bekiroğlu
Members Involved: Akif Erdem Tanyeri, Cenk Merih Olcay
Start Date: 30/01/2025 **End Date:** 09/02/2025
Objective: Implement the design of a study tracking system, including study goals and statistics as well as their interactions with users, such as notifications
Tasks:

T13.1: Implement study dashboard UI for users to set study goals
T13.2: Implement goal notifications for interactions of study goals and users
T13.3: Implement a statistics page for users to show what they achieve

T13.4: Integrate UI with Back-end Endpoints

Deliverables:

D13.1: Study Dashboard UI
D13.2: Goal Notifications
D13.3: Statistics Page UI

WP#14: Video/Voice Record Processing Development
Leader: Enes Bektaş
Members Involved: Cenk Merih Olcay
Start Date: 09/02/2025 **End Date:** 19/02/2025
Objective: Implement the features for extracting the transcripts from videos and voice records and generating flashcards
Tasks:

T14.1: Create a Speech to Text
T14.2: Create a Back-end service to manage voice records and text-indexed transcripts

Deliverables:

D14.1: Speech-to-Text Service Source Code
D14.2: Source Codes for the Back-end Services

WP#15: AI Model Integration with Backend Deck/Flashcard Services
Leader: Faruk Uçgun
Members Involved: Enes Bektaş
Start Date: 19/02/2025 **End Date:** 05/03/2025
Objective: Integrating the implemented Back-end services for flashcard generation features with AI Models
Tasks:

T15.1: Integrating the Lecture Notes Services with AI Models
T15.2: Integrating the Figures Services with AI Models
T15.3: Integrating the Video/Voice Recorded Services with AI Models

Deliverables:

D15.1: Integrated Back-end Services Updated Source Codes

WP#16: Quizzes and Questions Features UI Development
Leader: Akif Erdem Tanyeri
Members Involved: Ömer Fırat Bekiroğlu
Start Date: 19/02/2025 **End Date:** 20/03/2025
Objective: Implement quiz questions and quizzes as well as quiz related UI pages such as creating and editing quiz questions and quizzes. Additionally interactions with quiz questions and quizzes going to be implemented by following the mockups
Tasks:

T16.1: Prepare visual appearance of quiz questions and quizzes
T16.2: Implement editing and creating quiz questions and quizzes
T16.3: Implement user interactions with questions and quizzes
T16.4: Integrate UI with Back-end Endpoints

Deliverables:

D16.1: Quiz Question UI
D16.2: Quiz UI
D16.3: Generating, Creating, and Editing Quiz Question Pages UI
D16.4: Generating, Creating and Editing Quiz Pages UI

D16.5: Question Interactions
D16.6: Quiz Interactions

WP#17: Quizzes and Questions Features Development
Leader: Faruk Uçgun
Members Involved: Enes Bektaş
Start Date: 05/03/2025 **End Date:** 15/03/2025
Objective: Implement the Features for saving Test questions and Question Generation
Tasks:
T17.1: Implement Back-end Services for Questions and Quizzes
T17.2: Integrate the Back-end Service with generational AI Models
Deliverables:
D17.1: Source Codes for the created Back-end Services

WP#18: Dockerization and Microservice Architecture Implementation
Leader: Faruk Uçgun
Members Involved: Cenk Merih Olcay
Start Date: 30/11/2024 **End Date:** 20/01/2025
Objective: Configure the execution environments of the back-end, database services, and input processing services(image, videos, voice records) for deployment with Microservice Architecture
Tasks:
T18.1: Dockerize Back-end, Database, and Input Processing Services
T18.2: Integrate All the Services with each other
Deliverables:
D18.1: Dockerfiles and Docker Compose File

WP#19: Case Study AI Model Development
Leader: Enes Bektaş
Members Involved: Akif Erdem Tanyeri
Start Date: 09/02/2025 **End Date:** 01/03/2025
Objective: Finding suitable machine learning approaches and develop an AI model capable of generating case studies using LLMs
Tasks:
T19.1: Conduct research on machine learning techniques that are suitable
T19.2: Gather a dataset that is sufficient to generate new case studies
T19.3: Try LLMs for case study generation
T19.4: Develop a sufficient AI model using LLMs
Deliverables:
D19.1: Case Study Generating AI Model

WP#20: Case Study Feature Development
Leader: Enes Bektaş
Members Involved: Faruk Uçgun
Start Date: 05/03/2025 **End Date:** 15/03/2025
Objective: Develop case study features using the model that will be built at WP#19
Tasks:
T20.1: Case study creation functionalities development
T20.2: Case study editing functionalities development

T20.3: Development of user actions on case studies
T20.4: Progress and performance tracking development

Deliverables:

D20.1: Fully Functional Back-end Case Study Service Source Code

WP#21: Case Study UI Development
Leader: Akif Erdem Tanyeri
Members Involved: Ömer Fırat Bekiroğlu
Start Date: 05/03/2024 **End Date:** 20/03/2025
Objective: Implement a case study design that enables users to interact with those scenarios

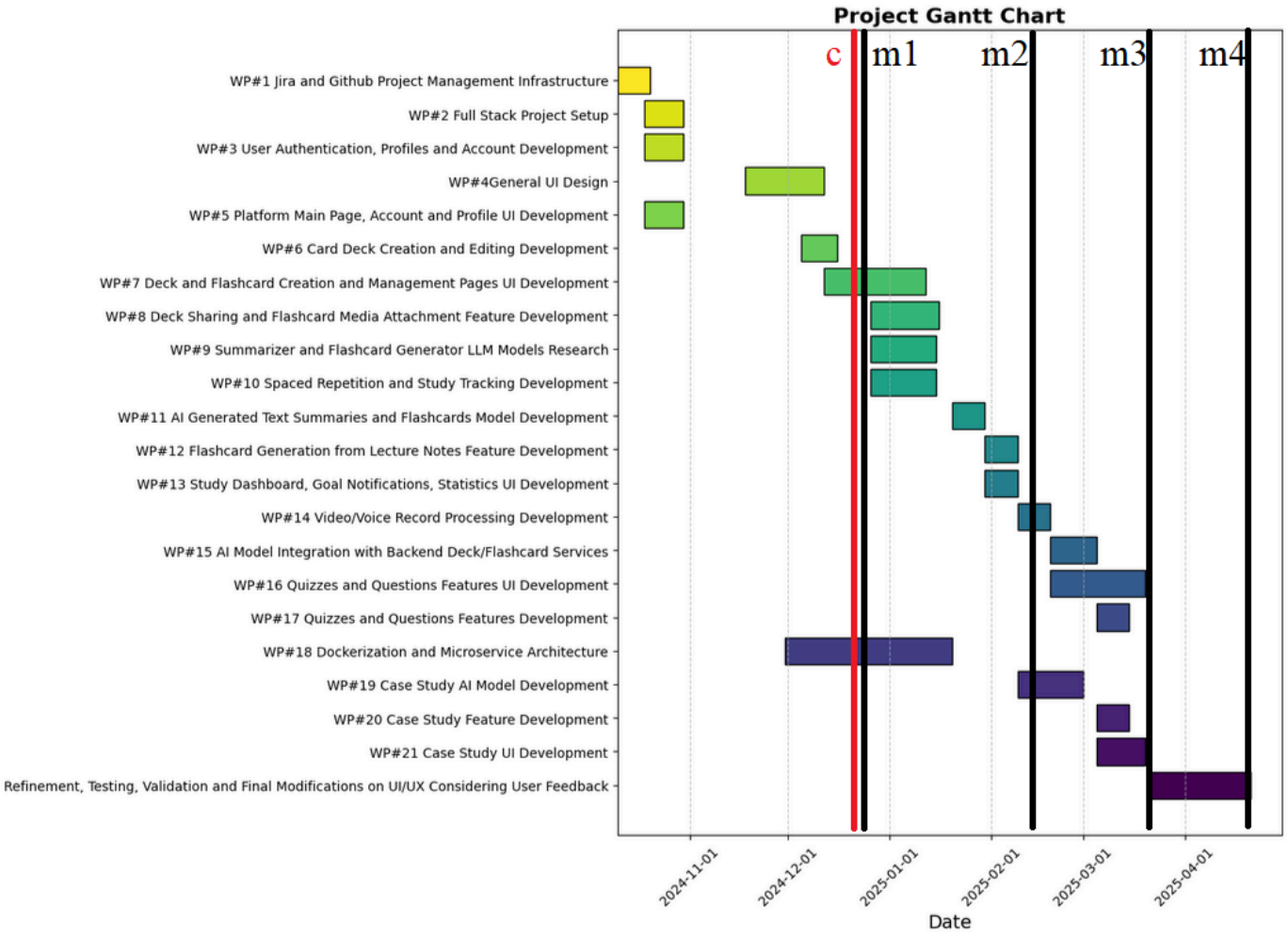
Tasks:

T21.1: Prepare case study main page
T21.2: Prepare interactive scenario UI
T21.3: Integrate UI with Back-end Endpoints

Deliverables:

D21.1: Case Study Main Page UI
D21.2: Interactive Scenario UI
D21.3: Dynamic Response and Statistics Components for Case Study

4.3.1. Gantt Chart



- c: Current
- m1: Minimum Viable Product
- m2: Ready For General User Beta Version
- m3: All Main Features Are Completed
- m4: CS Fair Final Version

Fig. 37: Project Gantt Chart

4.4. Ensuring Proper Teamwork

For our project, we have experienced how important proper teamwork is to create a high-quality product. Therefore, it is crucial to choose the best-fit methodology for teamwork. For this reason, we have searched those strategies available online, discussed and decided on the Agile/Scrum methodology. We follow Scrum practices with Jira software as our primary tool for task management, tracking progress, and maintaining transparency.

4.4.1. Sprint Planning

At the beginning of each sprint (2-week intervals), we are arranging a Planning Meeting for the objectives:

- Define the goals for the sprint.
- Break down responsibilities into Stories or Tasks.
- Assign priority levels for each task and estimation for the efforts.
- Ensure every team member understands their responsibilities and the objectives of the sprint.

4.4.2. Stand-up Meetings

Stand-ups are held 2 times a week (15 minutes maximum) where each member answers to the questions like:

- What have I accomplished since the last meeting?
- What am I working on next?
- Are there any problems preventing me from completing my tasks on time?

These meetings will help keep everyone on the schedule and ensure any challenges are identified and addressed.

4.4.3. Task Management Using Jira

Jira Sprint Boards is used to visualize our workflow and track progress in real-time. Tasks are organized into phases: To Do, In Progress, In Verification, and Done. Team members update their task status before each stand-up.

4.4.4. Sprint Retrospective

At the end of each Sprint, we hold a Retrospective Meeting to answer:

- What went well during this sprint?
- What could be improved for future sprints?

This retrospective ensures continuous improvement and helps the team address any recurring challenges.

4.5. Ethics and Professional Responsibilities

As ReMediCard.io is planned to be an everyday application for medicine students, it contains professional and ethical issues. Such issues can be grouped in the below subheadings:

4.5.1. Data Privacy and Security

The application will utilize private data such as lecture notes, voice recordings, and video recordings. Ensuring the safety and confidentiality of the user data is a must. In our project, we will encrypt all kinds of data during storage and transmission.

We will use data protection regulations such as the General Data Protection Regulation (GDPR) [4] to ensure user data privacy.

4.5.2. Potential Problems of AI Usage

The use of large language models (LLMs) such as ChatGPT or T5 may introduce bias in generated content. We will take steps to minimize the harmful biases in flashcards, questions, and feedback.
The system will include disclaimers about the limitations of generated content to prevent reliance on potentially incomplete or incorrect information.

4.5.3. Accessibility

The application aims to reach a large number of users. Hence, the system will support both Turkish and English to ensure people from different backgrounds can utilize the application.

4.5.4. Accuracy and Trustworthiness

As the accuracy of information in flashcards and quizzes is crucial, the application will incorporate quality assurance measures, such as validating content against trusted medical sources.

4.5.5. Copyright

We will ensure that copyright laws are compiled by reviewing any third-party content, such as medical images, external datasets or lecture materials carefully.

4.6. Planning for New Knowledge and Learning Strategies

4.6.1. Technical Skills

Mobile application development:
Our project aims to establish a comfortable and accessible platform for medical students and it was decided that mobile development is more suitable for this purpose. However, no group member has any experience in this field; thus, the team needs to follow specific ways to learn the skills needed. After the research, it was decided to use React Native. To learn React Native, online resources are used, such as publicly available online course videos and documentation.

Designing a Modular and Scalable Microservice architecture:
The project has a comprehensive scope, including different technological tools and approaches. This inclusiveness requires complex connectivity and communication between the parts. ToS handle this situation, our team had to use modular and scalable microservice architecture. Learning to design modular and scalable microservice architectures required combined theoretical knowledge with practical

implementation. To accomplish this, we started by defining distinct boundaries of different microservices and modules. These subtasks are divided among the group members to reduce the complexity and time spent during the learning and implementation phase. Members appointed to a subtask need to learn the concepts related to that subtask more deeply than others.

Configuring Extendable but Compact Database Structures:
We focused on understanding both relational and nonrelational databases. Recognizing their differences, strengths, and weaknesses is crucial for their respective uses during the construction of our project. Every group member has familiarity with both database styles at a certain level. This previous knowledge comes from both theoretical and practical experiences. With the help of this experience and online resources, desired database structure can be built.

Provisioning the Relevance and Authenticity of generated Content for Medical Studies:
We collected different kinds of educational materials for medical studies ranging from textbooks to lecture notes and online resources. By analyzing these materials and also experiencing the AI-based features of similar memorization products in the market, we identified the fundamental aspects of the study contents that must be regarded to ensure the generated content by AI Models will be relevant, inferrable, and authentic. These analyses will be prominent and handy when developing AI Models tailored to the needs of medical students.

Finding Ways to Convert Legacy Lecture Materials into Engaging Interactive Content:
We tried to find the absent attributes of the traditional lecture materials to understand why they failed to be sufficient for exam preparation. Also, we interviewed medical students to be aware of the additional aspects that traditional lecture materials lacked. This research will play a crucial role in modernizing medical studies with our generated content in ReMediCard.io’s features.

4.6.2. Non-Technical Skills

Finding the nature of the medical studies and the use of terminology in examination:
We obtained samples of written medical exams from the web, from our friends and acquaintances who have been engaged in medical studies, and observed these samples to better understand the nature and the general format of the way students are assessed to formulate our solutions more adaptively to the context of medical studies.

Finding the common practices and study behaviors of medical students:
We have conducted a survey to grasp the general preferences of medical students in their personal studies, determining features and their priorities as well as the way our mobile application is designed for personal use.

5. Glossary

AI (Artificial Intelligence): AI is the field of computer science that investigates how machines provide intelligent responses to encountered situations.

Computer Vision: Computer vision is a field of AI that uses machine learning and neural networks to teach computers and systems to derive meaningful information out of visual inputs.

Flashcards: Flashcards are cards with both sides containing information related to each other. Usually, one side is for questions, and the other side is for answers. These cards are used for studying.

LLM: Large Language Models are advanced AI systems trained on huge datasets to understand and generate human-like text.

Machine Learning: Machine learning is a branch of AI that focuses on using data and algorithms to enable AI to imitate the way humans learn.

Natural Language Processing (NLP): NLP is the field of computer science that focuses on making machines understand and use the languages used by humans.

Spaced repetition algorithm: Evidence-based learning technique where newly introduced or harder problems are shown more frequently. On the other hand, older and well-understood problems are shown to the user less frequently.

Speech-to-Text: A technology that converts spoken language into written text using automatic speech recognition systems.

6. References

[1] P. Dattathreya and S. Shillingford, “Identifying the ineffective study strategies of first year medical school students,” *Medical Science Educator*, vol. 27, no. 2, pp. 295–307, Mar. 2017, doi: 10.1007/s40670-017-0396-2.

[2] M. A. Aljaffer *et al.*, “The impact of study habits and personal factors on the academic achievement performances of medical students,” *BMC Medical Education*, vol. 24, no. 1, Aug. 2024, doi: 10.1186/s12909-024-05889-y.

[3] K. Baba, N. Cheimanoff, and N.-E. E. Faddouli, “A comparative study of active and passive learning approaches in hybrid learning, undergraduate, educational programs,” in *Advances in intelligent systems and computing*, 2020, pp. 715–725. doi: 10.1007/978-3-030-52249-0_48.

[4] “General Data Protection Regulation”. https://gdpr-info.eu. [Accessed: Nov 18, 2024].

[5] “IEEE Recommended Practice for Software Requirements Specifications”. https://standards.ieee.org/ieee/830/1222/. [Accessed: Nov 18, 2024].

[6] “IEEE Standard for Information Technology--Systems Design--Software Design Descriptions”. https://standards.ieee.org/ieee/1016/4502/. [Accessed: Nov 18, 2024].

[7] “About the Unified Modeling Language Specifications Version 2.5.1”. https://www.omg.org/spec/UML/2.5.1/About-UML. [Accessed: Nov 18, 2024].

[8] “ISO/IEC 27001:2022 – Information Security Management”. https://www.itgovernance.co.uk/iso27001. [Accessed: Nov 18, 2024].

[9] “IEEE Standard for Transparency of Autonomous Systems”. https://standards.ieee.org/ieee/7001/6929/. [Accessed: Nov 18, 2024].

[10] “Google Java Style Guide”. https://google.github.io/styleguide/javaguide.html. [Accessed: Nov 18, 2024].

[11] “ISO/IEC/IEEE International Standard - Software and systems engineering --Software testing”. https://standards.ieee.org/ieee/29119-1/10779/. [Accessed: Nov 18, 2024].