

Quiz Odyssey

Requirements Analysis

Group: 26

Eren Taşdemir 150200035

Berkant Bakışlı 150200069

Karol Jan Charchut 912310003

Pijus Jonas Navasaitis 912310007

Egi Gjineci 150190910

Table of Contents

Change Log Table	2
1. Introduction	3
1.1. Goal	3
1.2. Content	3
2. System Requirements	3
2.1. Functional Requirements	3
2.2. Non-functional Requirements	3
3. Use Cases	4
3.1. User Types	4
3.2. User Scenarios	5
3.3. Use Case Diagram	7
3.4. Use Cases with Flows	9
3.4.1. Registration	9
3.4.2. Logging in	9
3.4.3. Submit a question	10
3.4.4. Change username	10
3.4.5. Change password	10
3.4.6. Play a daily challenge	11
3.4.7. Play trivia game	11
3.4.8. View followers	12
3.4.9. View followed users	12
3.4.10. Review other user's profile	12
3.4.11. Remove user	13
3.4.12. Remove question	13
3.4.13. Review question list	14
3.4.14. Review community questions	14
3.4.15. Compare statistics between users	14
3.4.16. View users list	15
4. User Interface Model	15
5. Flow Diagrams	21
5.1. General Data Model	21
5.2. Important Data Considerations	23
5.3. Data Flow Diagrams	24
5.3.1. Level 0 (Context Level)	24
5.3.2. Level 1	24
5.3.3. Level 2	25

Change Log Table

Changed Item	Description	Date of Change
3.2.7: User Scenario - Play trivia game	Made the scenario more detailed by adding more sentences about the background.	29/11/2023
3.2.8: User Scenario - View followers	Made the scenario more detailed by adding more sentences about the background.	01/12/2023
3.2.9: User Scenario - View followed users	Made the scenario more detailed by adding more sentences about the background.	01/12/2023
3.3: Use Case Diagrams	Refactored a main use case diagram into two. One represents an abstract system with actors and different subsystems. The other diagrams are smaller and are more detailed versions of subsystems explaining their operations more in depth.	01/12/2023

1. Introduction

1.1. Goal

This project aims to design and implement a trivia-style mobile game that offers an engaging and educational experience. The game will allow users to answer questions on various topics, such as art, history, science, sports, entertainment, etc., and earn points based on their performance. The game will also feature daily and weekly challenges, where users can compete with other players in timed quizzes and rank in the leaderboards. The game will be compatible with both Android and iOS platforms and will follow the best practices of user interface design and accessibility.

1.2. Content

This document presents the requirements analysis for Quiz Odyssey. It describes the system requirements, use cases, user interface model, and flow diagrams of the software. The document is organized into five sections, each corresponding to one of the topics mentioned above. The purpose of this document is to provide a clear and comprehensive specification of the software functionality, design, and constraints, as well as to facilitate communication and collaboration among the stakeholders of the project.

2. System Requirements

2.1. Functional Requirements

- The system shall allow users to register
- The system shall allow previously register users to login
- The system shall allow users to take a quiz in a chosen category
- The system shall allow users to take a timed challenge quiz in a chosen category
- The system shall allow users to participate in daily challenges
- The system should show the users their streak count
- The system shall allow users to see their profiles, change their username or password
- Users shall be able to follow other users
- Users should be able to submit new questions
- Users shall be able to view their followers and people that they follow
- The system shall allow admins to view and ban users
- The system shall allow admins to promote users to moderators
- The system shall allow admins and moderators to view, edit, add and remove quiz questions.

2.2. Non-functional Requirements

- Performance:
 - Response Time: The response time of the system should always be lower than 5 seconds.

- Scalability: The system should initially support 1000 users and then scale to accommodate 10000 users within 6 months using cloud based options.
- Load Testing: Response time should meet specified requirements when tested with 2000 concurrent users in 24 hours.
- Security:
 - Data Encryption: The user's sensitive data should be encrypted using AES.
 - Authentication: The password should contain a minimum of 8 characters, including uppercase, lowercase, numbers, and special characters.
 - Authorization: Administrators shall have access to specific sections for player management and game monitoring.
 - Security Testing: Different testing methods shall be used throughout and after the project's development.
 - Compliance: The application should be GDPR-compliant.
- Usability and User Experience:
 - Accessibility: The system should use voice commands to support users with special needs(low vision).
 - User Interface Consistency: The user interface should provide easy language, familiar terms, and standard interface elements.
 - Error Handling: The system should provide error messages when wrong input is received.

3. Use Cases

3.1. User Types

Guest:	Can see the landing page Can register
Registered User:	Known by the system Can participate in challenges Can follow other users Can submit new questions Can see the statistics of other users
Moderator:	Can approve submitted questions Can add tags to questions
Administrator:	Can view the list of all users Can do everything a Moderator can do Can ban users Can make other users Moderator or Administrator

3.2. User Scenarios

3.2.1. The system shall allow users to register

Alice launches the application and is greeted with the welcome screen leading to a **registration form**. She enters her email address, picks a username and chooses her account password. She then sees a message indicating successful registration.

3.2.2. The system shall allow previously registered users to login

Alice, an already registered user, opens the app and gets to the login screen. She enters her email and password correctly. She is **authenticated** and redirected to the home screen of the application.

3.2.3. The system shall allow users to take a quiz in a chosen category

Alice, on the homepage of the application, sees a **list of categories** to choose from, such as history, science, sports, etc. She selects history and taps on the take quiz button. The app shows her **a question** with four possible answers. She answers the first question correctly, she **earns points** and moves on to the next question. She answers the second question incorrectly and the quiz moves on to the next question without giving any points. She completes the quiz and after the quiz ends, she sees her **score** and **rank** on the leaderboard.

3.2.4. The system shall allow users to take a timed challenge quiz in a chosen category

Alice, on the homepage of the application, sees a **list of categories** to choose from, such as history, science, sports, etc. She selects history and taps on the take quiz button. The app shows her **a question** with four possible answers and a **timer**. She has 20 seconds to select the correct answer. She answers the first question correctly, she earns points and moves on to the next question. She answers the second question incorrectly and runs out of time in the third question, the quiz moves on to the next question without giving any points. She completes the quiz and after the quiz ends, she sees her **score** and **rank** on the leaderboard.

3.2.5. The system shall allow users to participate in daily challenges

Alice chooses to enter the daily challenge quiz and clicks on the **daily challenge** button on the homepage. The app shows her 10 mixed-category questions one by one. For every correct answer, she earns points. After she completes all the questions, she sees the quiz score and her rank on the daily **leaderboard**.

3.2.6. The system shall allow users to see their profiles, and change their username or password

Alice, thinking that her password is no longer secure, wants to change her password. She clicks on the profile button in the tab bar. She can see her username and several buttons named change username and change password. She clicks on the **change password button** and she is shown a form where she needs to enter her current password and the new password. After she correctly inputs passwords her password is changed.

3.2.7. The system should show the users their streak count

Alice opened the app and tapped on her profile icon. She saw her streak count, showing how many **consecutive days** she had completed at least one quiz on the app. She was proud of her progress and determined to keep it up. She scrolled through the list of categories available and chose one that interested her. She answered the quiz questions carefully and checked her score at the end. She had done well, and she felt a surge of satisfaction. She looked at her streak count again and saw that it had **increased by one**. She closed the app, feeling happy and motivated.

3.2.8. Users shall be able to follow other users

Alice has become familiar with how her own profile works. Now, she is curious to learn about other users and their progress. She also wants to check out how her best friend Bob, from abroad, is doing. She knows his username and finds him via the search tool. Inside his profile, she comes across the "follow" button. She is happy because the app helps her to stay connected with her friends regardless of location.

3.2.9. Users should be able to submit new questions

At this point, Alice has spent a while playing this game. She is an ambitious, creative and well-read person, who likes to share her knowledge with other people. She would like to contribute to the application development. She finds out that there is an option to submit her own custom question. Alice is a passionate history student and submits a couple of questions regarding the Warring States period in China. She is impatiently waiting to see her questions accepted and included in the history category. She is excited about sharing her deep interest with the whole community.

3.2.10. Users shall be able to view their followers and people that they follow

Alice can view people who follow her and the people she follows. She goes to her profile and heads to the appropriate section where these people are listed. To select between followers and followings, she just switches between tabs.

3.2.11. The system shall allow admins to view and ban users

Bob, an admin user, enters the system management interface. He can see **all the users in the system**. He sees that one of the users has an offensive username. He clicks on the **ban user** button, shown alongside the user in the list. The user with the offensive username can no longer access the application.

3.2.12. The system shall allow admins to promote users to moderators

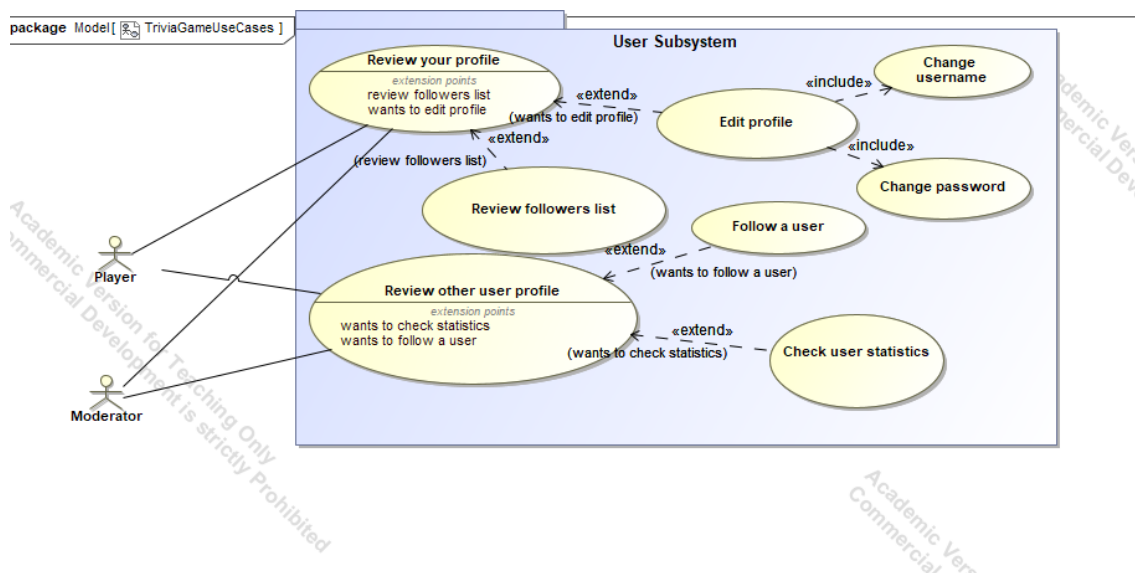
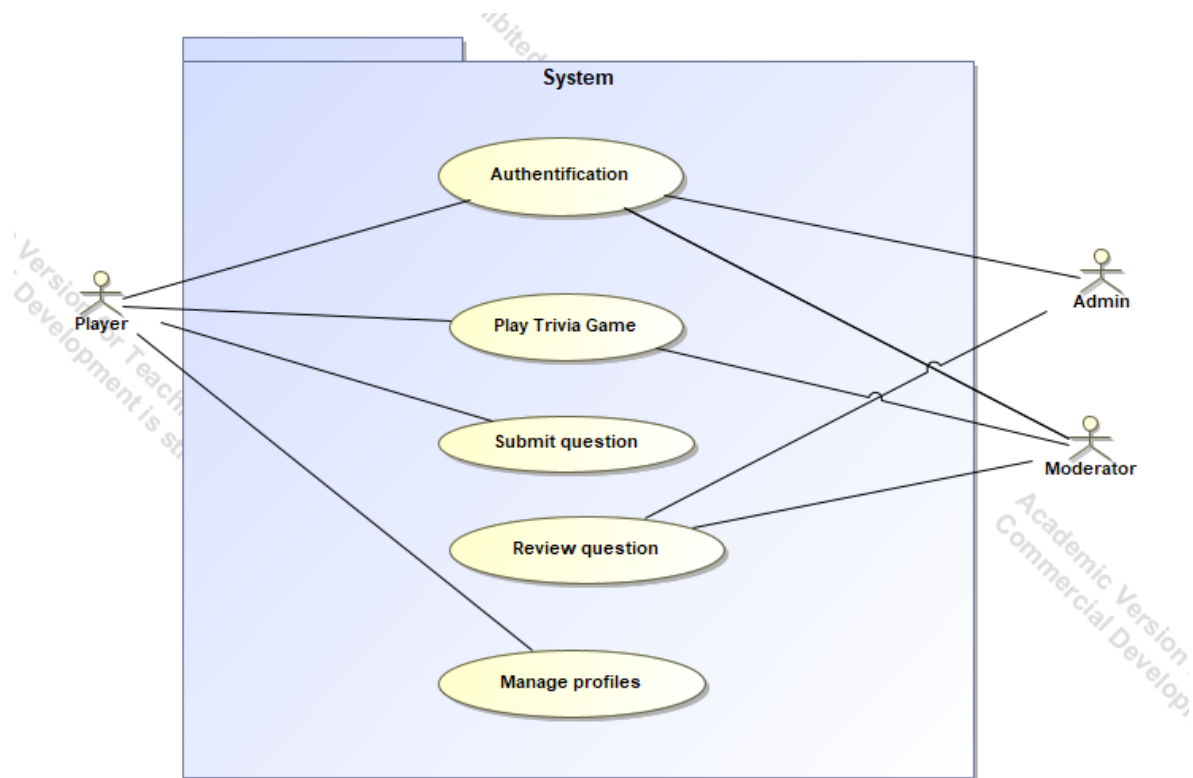
Bob, an admin user, wants to have someone as a moderator in the system. He opens up the user list in the management interface and clicks the **make moderator button** of the user Charlie. Charlie is now a moderator.

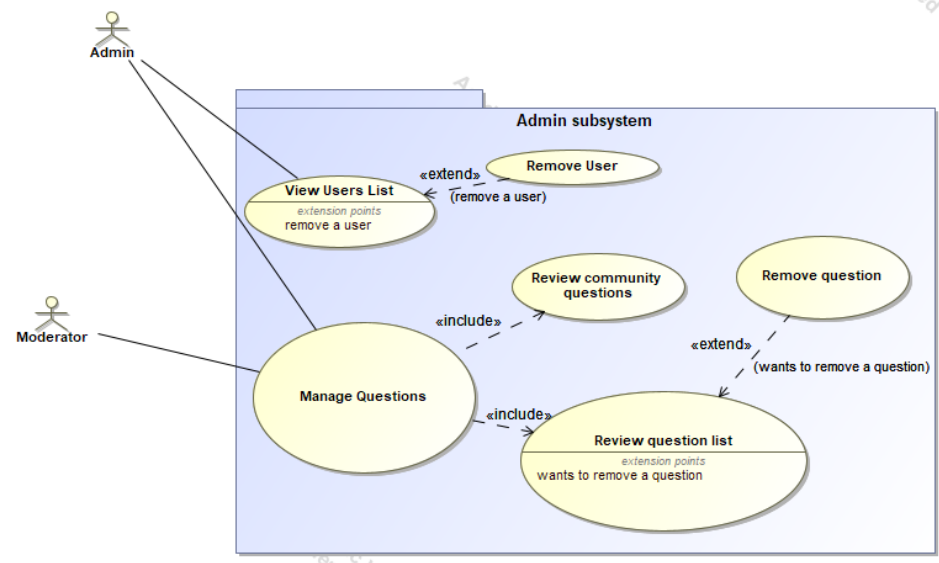
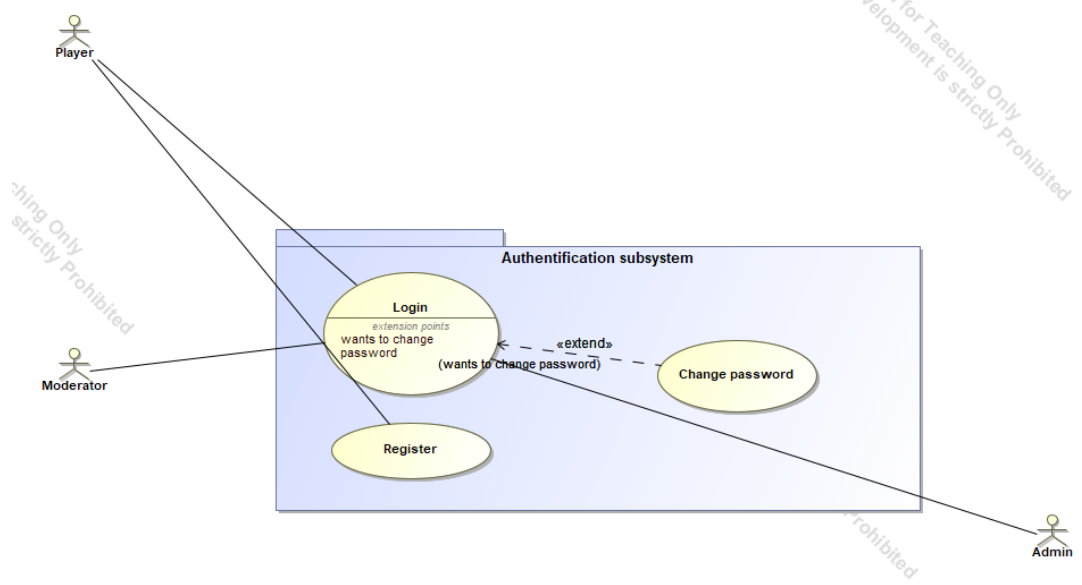
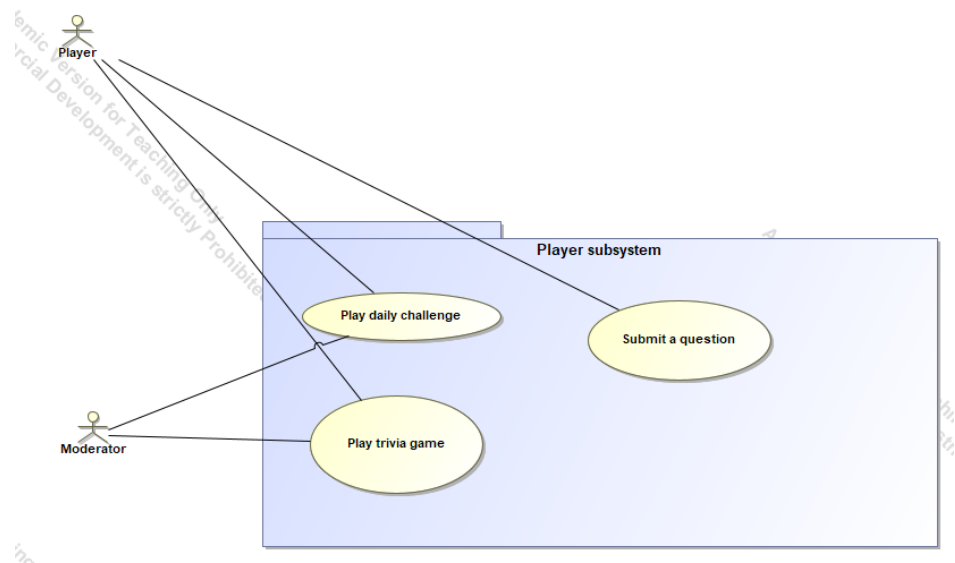
3.2.13. The system shall allow admins and moderators to view, edit, add and remove quiz questions.

Charlie, a moderator, wants to **remove a question** as he sees that the question has an ambiguous answer. He enters the management interface and clicks on the questions button. He is shown a **list of questions** with **category filters**. He finds the question and

clicks on the remove question button. The question is now removed from the system and **will no longer be included** in the quizzes.

3.3. Use Case Diagram





3.4. Use Cases with Flows

Register	Change password	View followed users	Review question list
Login	Play a daily challenge	Review other user's profile	Review community questions
Submit a question	Play trivia game	Remove user	Compare statistics between users
Change username	View followers	Remove question	View users list

3.4.1. Registration

Primary Actor:	Guest
Brief:	User should register in order to use the application
Basic Flow:	<ol style="list-style-type: none">1. Guest directed to the registration page2. Requested credentials (Email, Username, Password) supplied by user and submitted3. Credentials validated, added to database and user redirected to the page for registered users
Alternative Flow:	<ol style="list-style-type: none">1. Guest directed to the registration page2. Requested credentials (Email, Username, Password) supplied by user and submitted3. Credentials invalid (i.e invalid email, password not long enough, username already used etc.)4. An alert stating the error is shown to the user.

3.4.2. Logging in

Primary Actor:	Registered User
Brief:	User should login in order to use the application
Basic Flow:	<ol style="list-style-type: none">1. User directed to the login page2. User enters username and password3. Credentials valid4. User redirected to the page for registered users
Alternative Flow:	<ol style="list-style-type: none">1. User directed to the login page2. User enters username and password3. Credentials invalid4. An alert saying invalid credentials is shown to users

3.4.3. Submit a question

Primary Actor:	Registered User
Brief:	User should be able to submit questions to be included in the game
Basic Flow:	<ol style="list-style-type: none">1. User clicks the "Submit a Question" button2. User selects a category from the available ones3. User is asked to type the question, and the answer options4. User clicks the submit button5. A check is done to see if all the required fields are filled6. Check passed7. Question is sent to review by the moderator, and a success message is shown to the user
Alternative Flow:	<ol style="list-style-type: none">1. User clicks the "Submit a Question" button2. User selects a category from the available ones3. User is asked to type the question, and the answer options4. User clicks the submit button5. A check is done to see if all the required fields are filled6. Check failed7. An error message is shown to the user

3.4.4. Change username

Primary Actor:	Registered User
Brief:	User should be able to change their username
Basic Flow:	<ol style="list-style-type: none">1. User clicks the edit button next to their username2. A new username is typed by the user3. The username availability is checked4. A green tick is shown next to the field if the username is available5. User clicks confirm button6. Username is changed
Alternative Flow:	<ol style="list-style-type: none">1. User clicks the edit button next to their username2. A new username is typed by the user3. The username availability is checked4. A red cross is shown next to the field if the username is not available5. Confirm button is not clickable

3.4.5. Change password

Primary Actor:	Registered User
Brief:	User should be able to change their password
Basic Flow:	<ol style="list-style-type: none">1. User clicks the change password button2. The old password and new password is asked to the user, along with a confirmation of the new password

	<ol style="list-style-type: none"> 3. User clicks confirm button 4. The old password is validated and the typed passwords match 5. Password is changed and a message is shown to the user
Alternative Flow:	<ol style="list-style-type: none"> 1. User clicks the change password button 2. The old password and new password is asked to the user, along with a confirmation of the new password 3. User clicks confirm button 4. The old password is not validated, or the typed passwords do not match 5. Password is not changed and the relevant error message is shown to user

3.4.6. Play a daily challenge

Primary Actor:	Registered User
Brief:	User should be able to participate in daily challenges
Basic Flow:	<ol style="list-style-type: none"> 1. User clicks on the daily challenge button 2. A question is displayed to the user until all questions are over (10 questions) 3. User clicks on their answer for every question 4. If their answer is correct, points are added. If not, the correct answer is shown and no points are added. 5. Quiz finishes and the quiz score is displayed, along with the user's rank in the daily leaderboard.
Alternative Flow:	<ol style="list-style-type: none"> 1. User clicks on the daily challenge button 2. A question is displayed to the user until all questions are over (10 questions) 3. User decides to quit by pressing the proper button any time throughout the quiz 4. No points are added to account stats, and the user is returned to the home page.

3.4.7. Play trivia game

Primary Actor:	Registered User
Brief:	User should be able to start playing a quiz.
Basic Flow:	<ol style="list-style-type: none"> 1. User enters quiz categories section 2. User chooses category they want to play 3. The quiz starts 4. User selects answers and submits it for each individual question 5. Correct answer is shown after submitting the answer 6. Quiz finishes and the summary is displayed. 7. User is awarded points for each correctly answered question. 8. User score is updated - point for the quiz added to his score.

Alternative Flow:	<ol style="list-style-type: none"> 1. User enters quiz categories section 2. User chooses category he wants to play 3. The quiz starts 4. User decides to quit by pressing the proper button 5. No points are added to account stats
--------------------------	---

3.4.8. View followers

Primary Actor:	Registered User
Brief:	Allows user to view people who follow him
Basic Flow:	<ol style="list-style-type: none"> 1. User enters his profile section 2. User clicks followers button 3. List of followers is displayed
Alternative Flow:	None

3.4.9. View followed users

Primary Actor:	Registered User
Brief:	Allows user to view people that he follows
Basic Flow:	<ol style="list-style-type: none"> 1. User enters his profile section 2. User clicks following button 3. List of followed users is displayed
Alternative Flow:	None

3.4.10. Review other user's profile

Primary Actor:	Registered User
Brief:	Allows to view other user's profile page – information about that user
Basic Flow:	<ol style="list-style-type: none"> 1. User searches for and finds other user 2. User clicks on that user 3. User is redirected to the other user's profile 4. The profile displayed statistics about that user as well as comparisons between them.
Alternative Flow:	<ol style="list-style-type: none"> 1. User finds the user in his followers 2. User clicks on that user 3. User is redirected to the other user's profile 4. The profile displayed statistics about that user as well as comparisons between them.

3.4.11. Remove user

Primary Actor:	Administrator
-----------------------	---------------

Brief:	User requests to delete their account or administrator removes a user due to violations, inactivity or other reasons.
Basic Flow:	<ol style="list-style-type: none"> 1. Administrator logs into the admin dashboard and navigates to the user management section. 2. Administrator searches for a user and removes them. 3. System asks for confirmation. 4. Administrator confirms the removal. 5. System deletes the account and associated data from the database.
Alternative Flow:	<ol style="list-style-type: none"> 1. Administrator logs into the admin dashboard and navigates to the user management section. 2. Administrator searches for a user to remove. 3. User does not exist or has already been removed, so the system prompts an error message.

3.4.12. Remove question

Primary Actor:	Moderator
Brief:	Moderator removes a question from the game due to inaccuracies or irrelevance.
Basic Flow:	<ol style="list-style-type: none"> 1. Moderator accesses the moderator dashboard and navigates to the question management section. 2. System displays the list of questions. 3. Moderator searches and selects the question to remove. 4. System asks for confirmation and the moderator confirms the removal. 5. System removes the question from the database.
Alternative Flow:	<ol style="list-style-type: none"> 1. Moderator accesses the moderator dashboard and navigates to the question management section. 2. System displays the list of questions. 3. Moderator searches for the question to remove. 4. Question does not exist so the system prompts an error message.

3.4.13. Review question list

Primary Actor:	Moderator
-----------------------	-----------

Brief:	Moderator reviews the list of available questions in the app.
Basic Flow:	<ol style="list-style-type: none"> 1. Moderator logs into the moderator dashboard and navigates to the question management section. 2. System displays the list of available questions. 3. Moderator views the questions, sorted by categories or other filters. 4. Moderator may edit, add, or remove questions as necessary.
Alternative Flow:	None

3.4.14. Review community questions

Primary Actor:	Moderator
Brief:	Moderator reviews user-submitted questions before inclusion in the question database.
Basic Flow:	<ol style="list-style-type: none"> 1. Moderator accesses the community question review section in the moderator dashboard. 2. System displays a list of pending community-submitted questions. 3. Moderator reviews each question for accuracy and relevance. 4. Question is approved and is added to the database. 5. The user that has submitted the question gets notified.
Alternative Flow:	<ol style="list-style-type: none"> 1. Moderator accesses the community question review section in the moderator dashboard. 2. System displays a list of pending community-submitted questions. 3. Moderator reviews each question for accuracy and relevance. 4. Question is rejected. 5. User gets notification for the rejected question.

3.4.15. Compare statistics between users

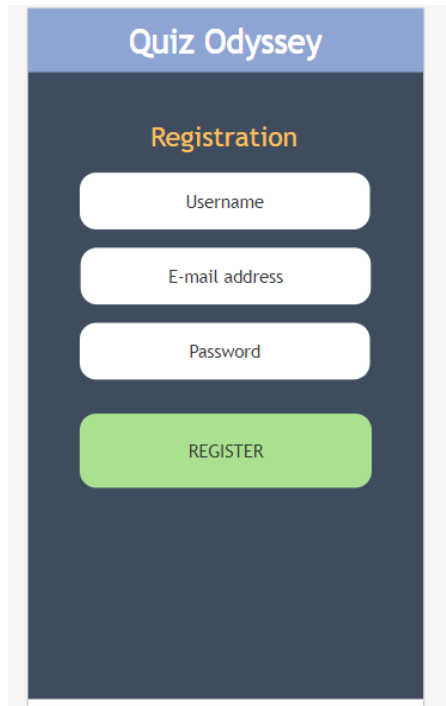
Primary Actor:	Registered user
Brief:	User checks for statistical comparison with another user.
Basic Flow:	<ol style="list-style-type: none"> 1. User visits another user's profile 2. User chooses to compare their statistics. 3. Weekly comparison visually displayed.
Alternative Flow:	<ol style="list-style-type: none"> 1. User visits another user's profile 2. User chooses to compare their statistics. 3. System prompts a message that no comparison can be made due to lack of statistics from the other user.

3.4.16. View users list

Primary Actor:	Administrator
-----------------------	---------------

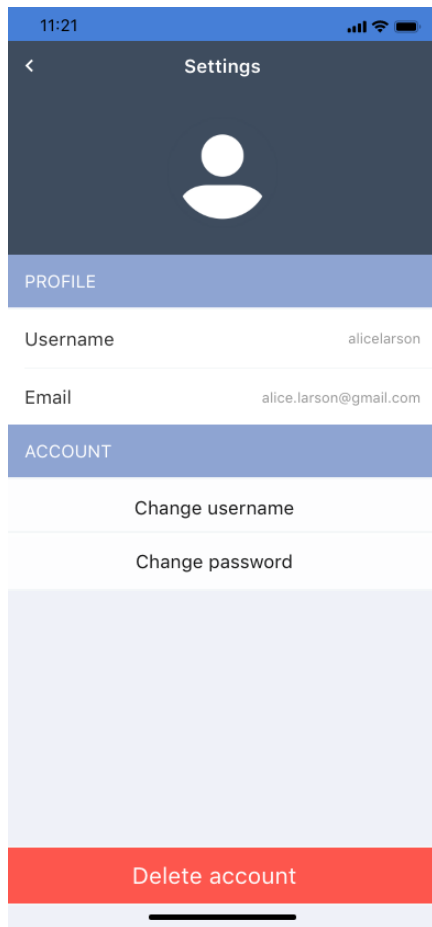
Brief:	Administrator accesses the complete list of users registered in the app.
Basic Flow:	<ol style="list-style-type: none"> 1. Administrator logs into the admin dashboard and navigates to the user management section. 2. System displays the complete list of registered users and the associated data. 3. Administrator can sort, filter, search, delete specific users or choose to promote a user to moderator.
Alternative Flow:	<ol style="list-style-type: none"> 1. Administrator logs into the admin dashboard and navigates to the user management section. 2. System prompts a message that there are no registered users.

4. User Interface Model



The image shows a mobile app interface for 'Quiz Odyssey'. At the top is a blue header with the text 'Quiz Odyssey'. Below the header is a dark blue background with the word 'Registration' in orange. There are four white input fields stacked vertically: 'Username', 'E-mail address', and 'Password'. Below these fields is a green button with the text 'REGISTER' in white capital letters.

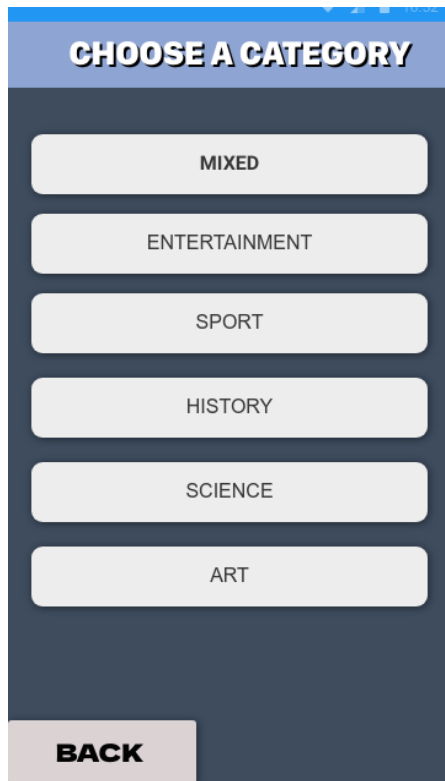
As a user, I want to be able to register and log in so that I can come back at any time and not lose my progress.



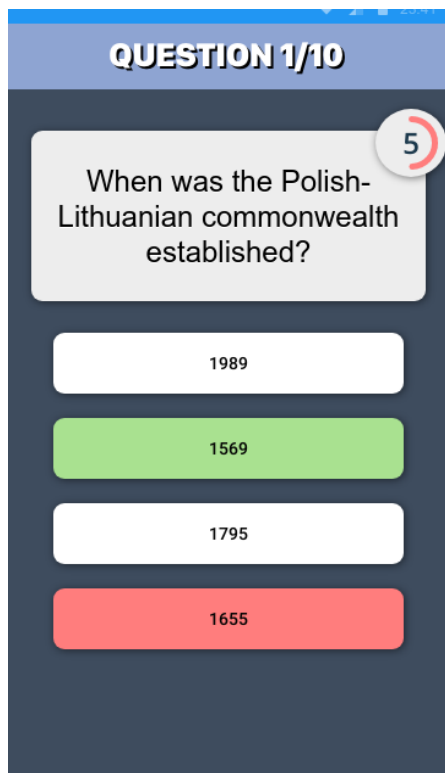
As a user, after creating an account, I want to be able to change my username and password.



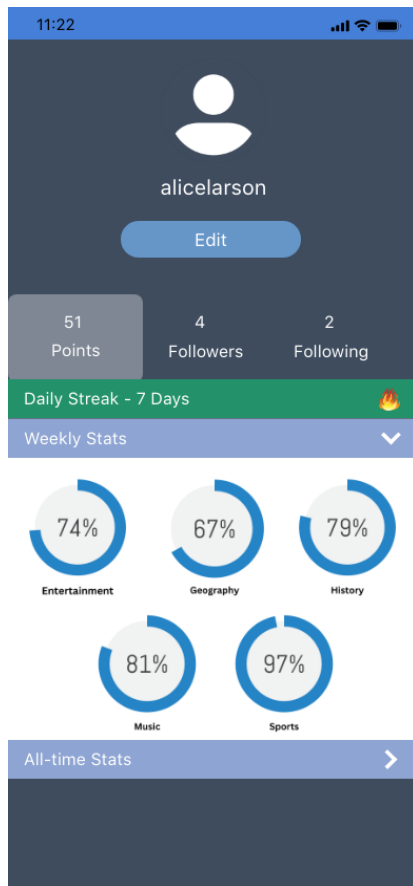
As a player, I want to be able to play daily quizzes to test my general knowledge.



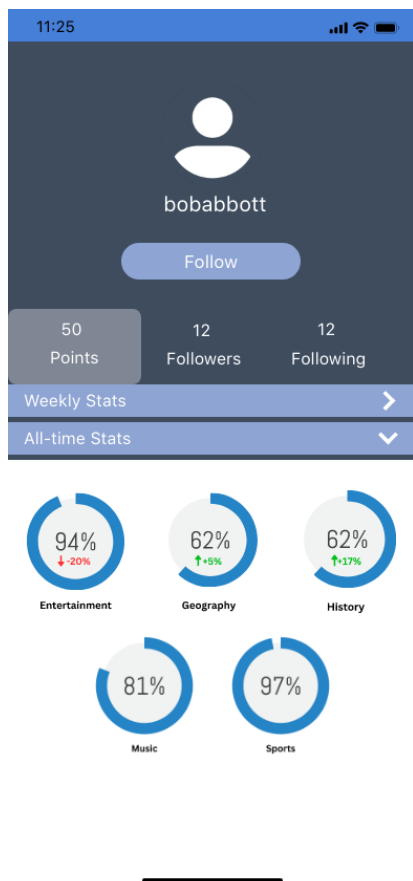
As a player, I want to be able to select different categories of trivia questions so that I can play in areas I'm interested in.



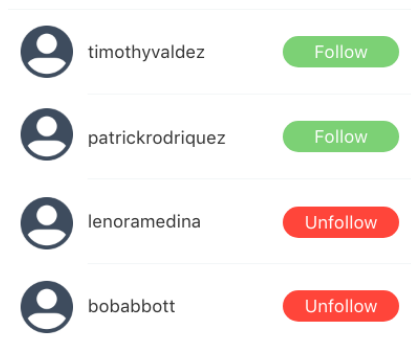
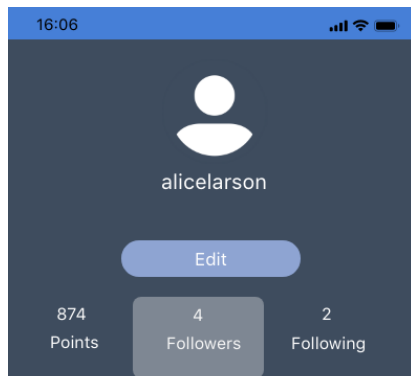
As a player, I want to play within a time limit to add excitement and prevent overthinking.



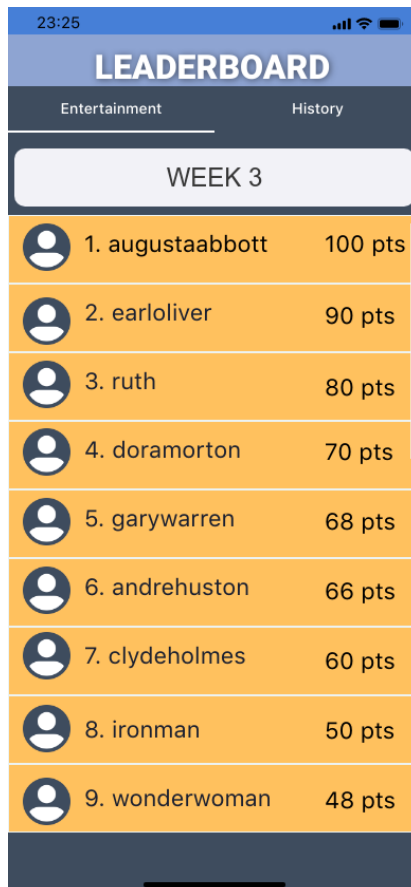
As a player, I want to check my weekly progress and all-time statistics as well as whether I am in a daily streak.



As a player, I want to be able to check other players' statistics, and see our differences.



As a player, I want to be able to see my followers, and the players that I follow.



As a player I want to be able to check top weekly rankings among players.

SUBMIT A QUESTION

Enter the question..

Enter answers and select correct ones

1989 ☐

1569 ☒

1795 ☐

Enter an answer.. ☐

CANCEL **SUBMIT**

As a player, I want to be able to submit questions so that I can make the game more competitive.

QUESTIONS LIST

Category 1

Category 2

Category 3

[Place for question]
- answer 1
- answer 2
...
- answer n-th

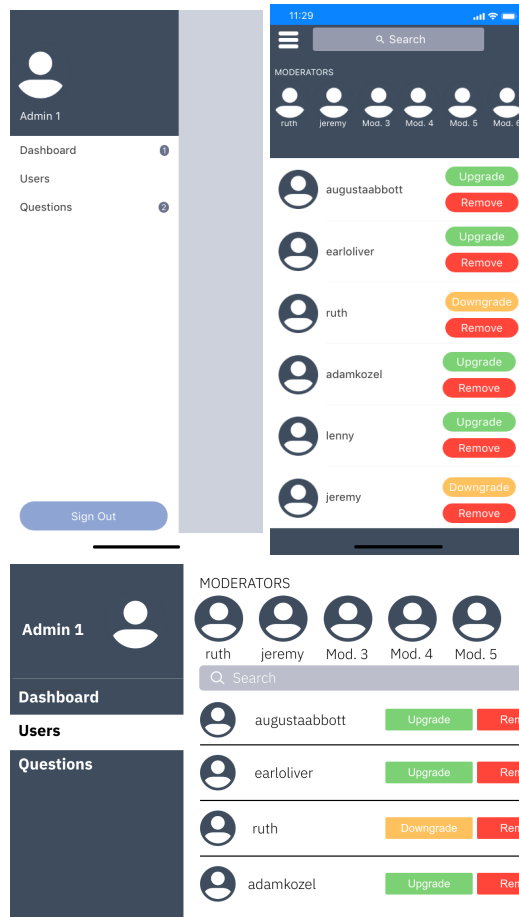
☒ ☐

[Place for question]
- answer 1
- answer 2
...
- answer n-th

☒ ☐

Category 4

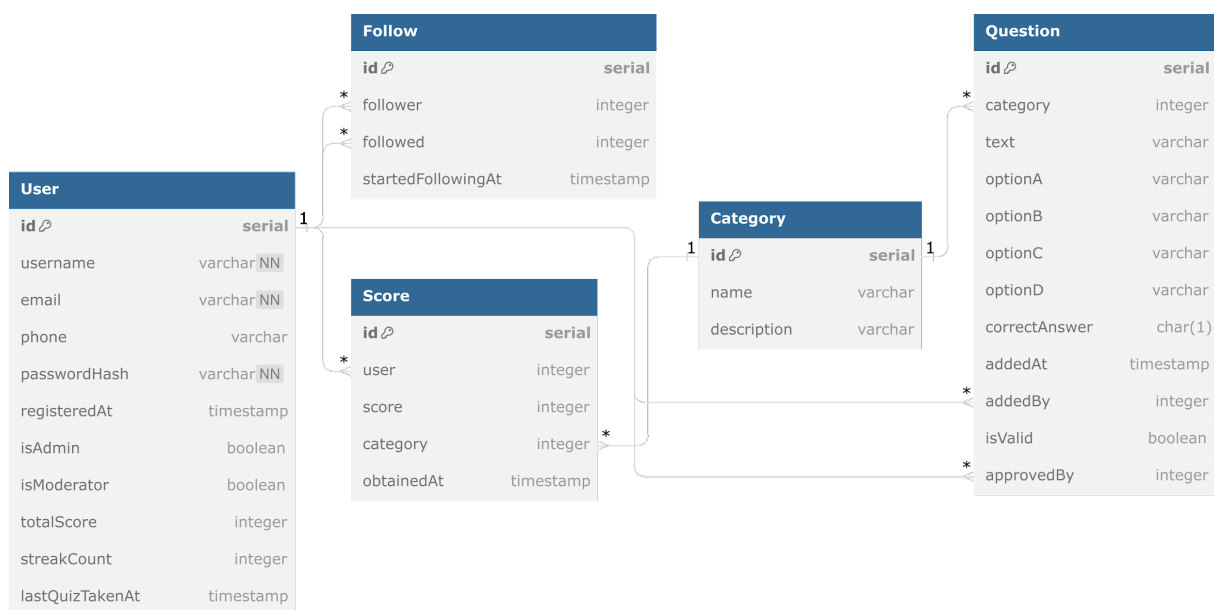
As a moderator, I want to be able to accept or reject questions submitted by players.



As an admin, I want to be able to manage users by searching, removing and upgrading/downgrading them to moderators.

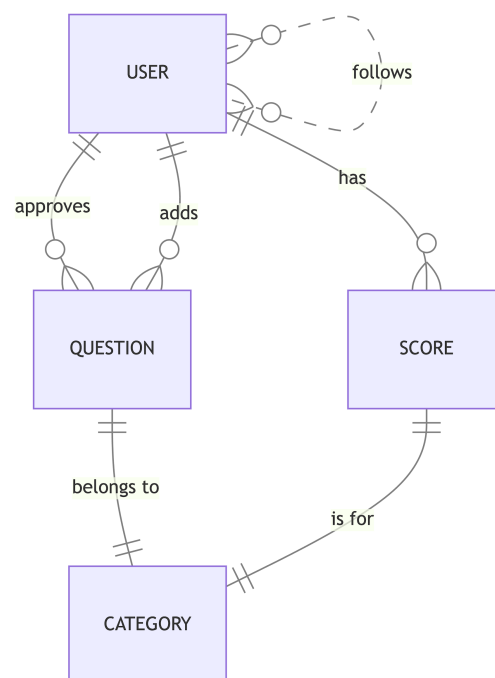
5. Flow Diagrams

5.1. General Data Model



- The users are kept in the “User” table
 - Their data such as email, phone, username and their hashed passwords are kept in this table, along with their user type in the form of boolean flags.
 - To make total score calculation more efficient, their total score is kept inside the User table as well, instead of recalculating it every time it is needed.
 - There are various other tables referencing the ID in this table as foreign keys.
- The users’ following/followers information is kept in the “Follow” table.
 - Data is kept in a format like (FollowerID, FollowedUserID, TimeOfFollow)
 - This table will be iterated through when listing the following information of any user.
- The questions are kept in the “Questions” table.
 - Every question has an ID, the question text and room for up to 4 options in the database.
 - The correct answer is also kept inside of the database.
 - Other flags are also kept, such as question addition time, the submitter of the question and the moderator who approved the question. The question’s approval status is kept in the “isValid” entry.
 - The question category is kept as a foreign key referencing the “Category” table
- The question categories are kept in the “Category” table to enable the addition of new categories without the need for updating the frontend.
- The score related information is kept in the “Score” table.
 - Each score entry has an ID, a timestamp and a score value.
 - Each entry is also tagged by category, so that comparing scores within a certain category is possible.

The relationship between these tables can be seen in the ER Diagram below:



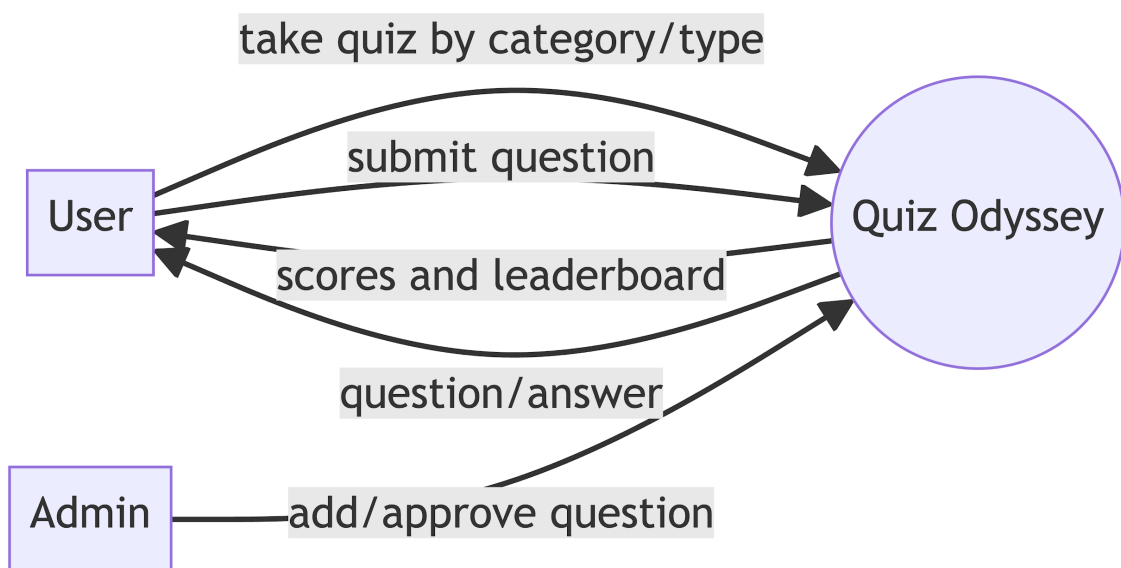
5.2. Important Data Considerations

We will be using React Native for our application, but to have room for expansion in the future, such as a possible creation of a web interface, we have decided to go with a server-client architecture. Therefore, we will have a REST API that uses JSON when handling requests and sending responses.

After our client sends a properly formatted request to the server, the server will generate the necessary response by performing SQL queries on the database. Afterwards, the response JSON will be parsed by the client to display the information in a user-friendly fashion.

5.3. Data Flow Diagrams

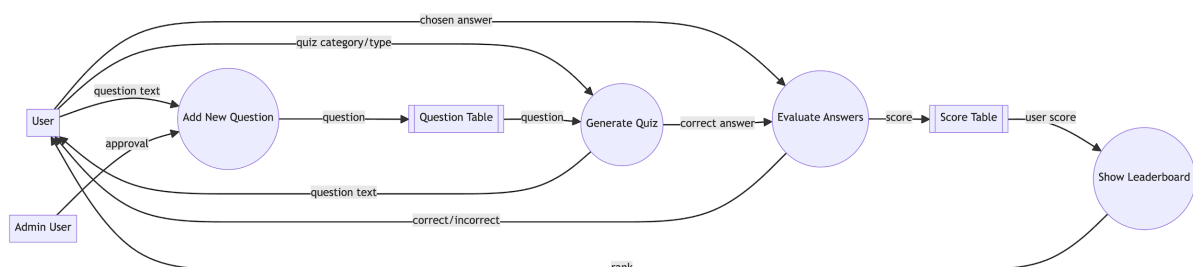
5.3.1. Level 0 (Context Level)



As seen from the above Level 0 Data Flow Diagram, users and admins of our application can do the following:

- Users can submit questions for review
 - Admins can add and approve these questions
- Users can take quizzes by choosing a category, or opting for a daily challenge.
- Users can see questions and correctness of their answers to each question.
- Users can see their scores and the respective leaderboard.

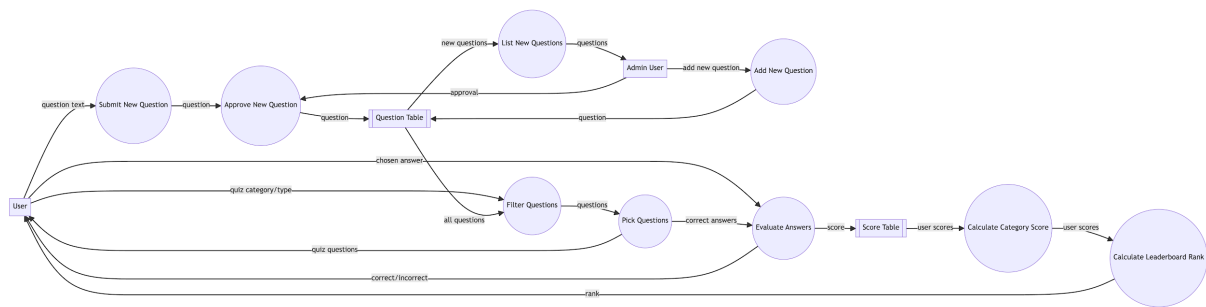
5.3.2. Level 1



Our system has four major processes in the quizzing mechanism, when broken down to Level 1. About the details of these processes:

- The user can send requests to these processes to get a response.
- The Add New Question process gets the question from a regular user and an approval from the Admin or Moderator, and updates the question table respectively.
- To play a quiz, the user sends a request to the Generate Quiz process, and the process pulls questions from the database. Afterwards, the questions are displayed to the user.
- When the user answers a question, the answer is sent to the Evaluate Answers process, and the correct answer is pulled from the generated quiz. The answer's correctness is shown to the user, and the Score table is updated with the calculated score.
- At the end of a quiz or on request, the Show Leaderboard process pulls the user scores from the Score table and displays that to the user.

5.3.3. Level 2



In the Level 2 Diagram, every process except for the evaluation of answers was broken down further. Their details are as follows:

- The addition of a question by a user goes to approval for an Admin, and the admin can list the new questions and add questions without the need for approval.
- When the user chooses a category or a type of quiz, firstly questions matching the filter are sent to the question picking process, and from there the questions and their answers are passed to the user and the evaluation process respectively.
- The calculation of the leaderboard rank first happens through the calculation of category scores, ultimately to be converted into a rank to be displayed to the user.