Gamification Application Documentation

Contents

1 Description

The application allows the addition of game mechanics into a non-game context to increase participation and inspire the users to collaborate and interact. The users are rewarded with tokens and badges for completing quests. Any user can propose a quest as long as he has enough tokens to reward the participants (a quest must reward at least 10 tokens and adding a badge to it will increase the cost by 100 tokens). The quests must be first approved by an administrator. If a quest is approved, it will appear in the list and the creator will receive the quest's badge if there is one. If a quest is rejected, it will be removed and the creator will be refunded. The users can submit answers to the quests and their creators will be able to accept or reject the submissions. The users are ranked based on the number of quests they have completed. An user has a profile, where he can upload a profile picture, add a description or display a featured badge.

2 Functional Requirements

- Account system - the application must allow the users to create accounts, recover their data and manage them
- Administration system - the application must have an administrator role which allows the users having it to suspend or unsuspend accounts, accept or reject quests and delete accounts or quests
- Quest system - the application must have a quest system. The users can view a list of quests and their status (not submitted, submitted, rejected, accepted), they can click on a quest to view its details and to submit an answer(or display the submitted answer). The users can view all their submissions and their created quests(which will be displayed in red if there are unchecked submissions for them). By clicking on a quest created by them, the users will view the list of submissions and will be able to accept or reject them. Users can create quests and badges by paying tokens. The users can win tokens and badges by completing quests
- Profile system - the users can upload a profile picture, set their displayed name, set a description or display a badge they own
- Leaderboard system - the users will be ranked based on the number of completed quests

3 Use Cases

This section covers the use cases of the application.

3.1 Use Case Diagram

The use case diagram of the application is shown below:

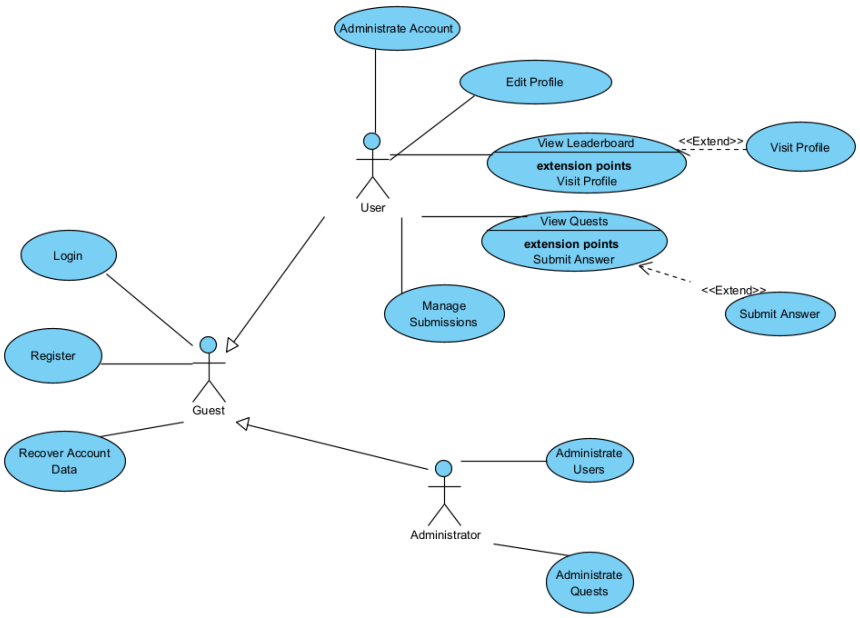

Figure 1: Use Case Diagram

3.2 Use Cases

Login

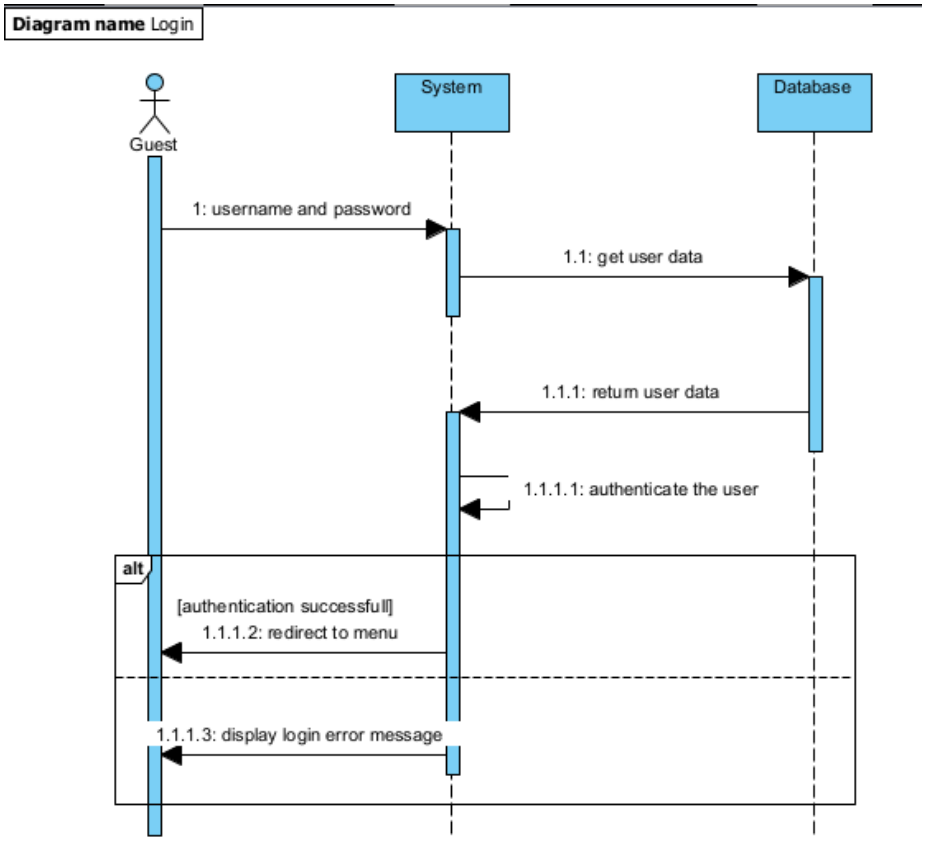

Figure 2: Login

The guest enters his username and password. If they are correct, he is logged in, else an error is displayed.
Register

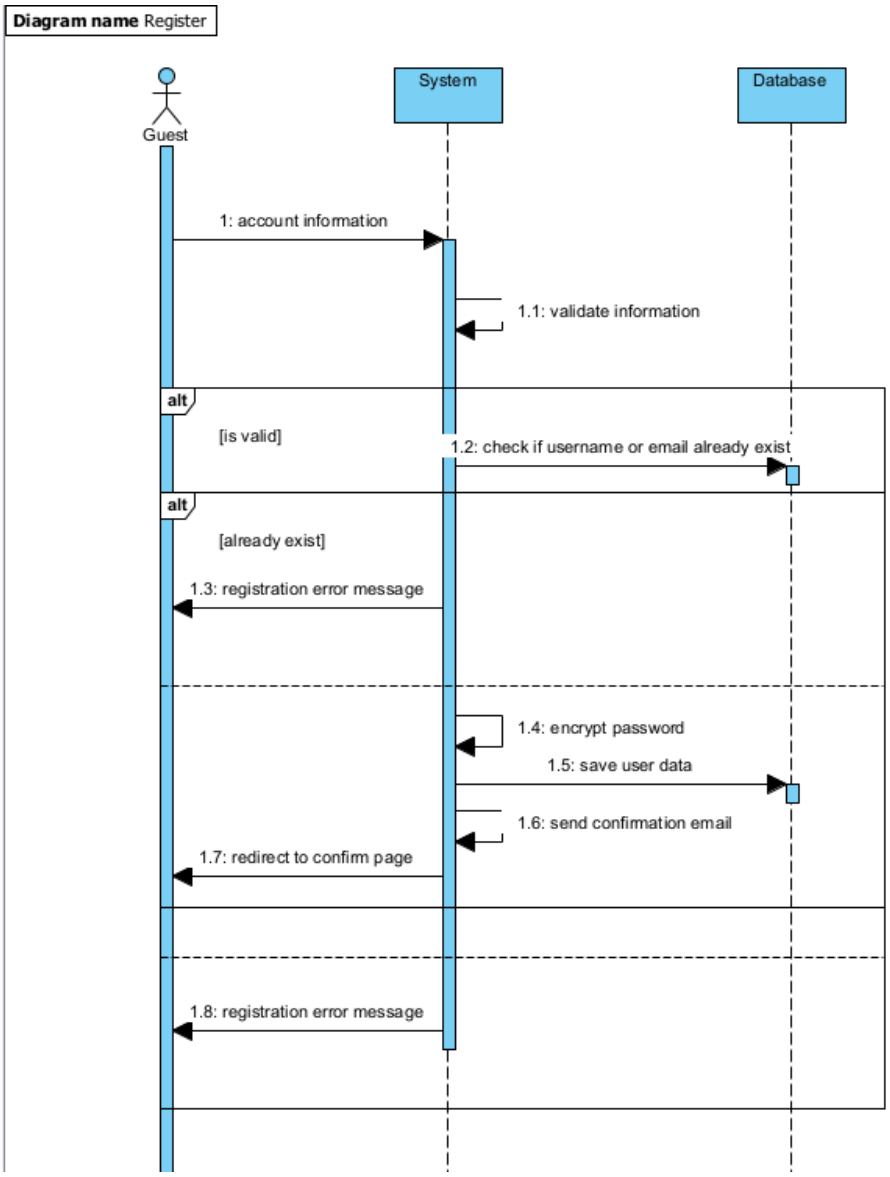

Figure 3: Register

The guest fills the registration form. If the data is valid and the username or email are not already taken, then an account is created, else an error is displayed.
Administrate Users

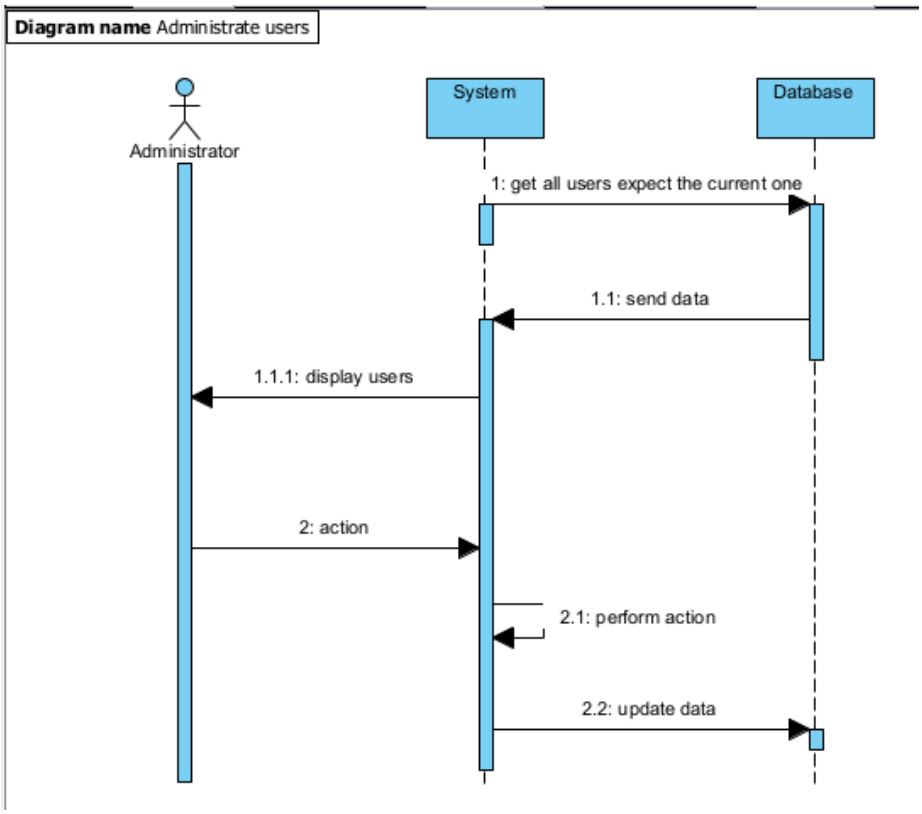

Figure 4: Administrate Users

The administrator views a list of users. He can suspend/unsuspend their accounts or delete them.
Administrate Quests

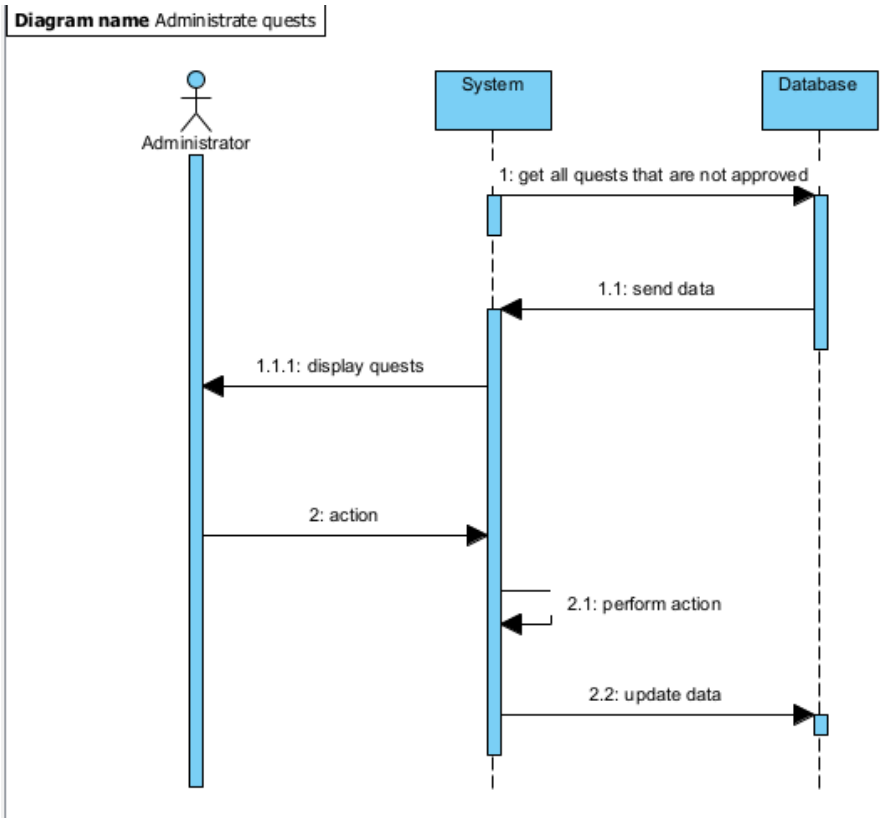

Figure 5: Administrate Quests

The administrator views a list of unapproved quests. He can approve them, which will make them active and give the quest's creator the badge associated with them if there is one, or reject them, which will delete the quest and the badge associated with it if there is any and will refund the creator.

Administrate Account

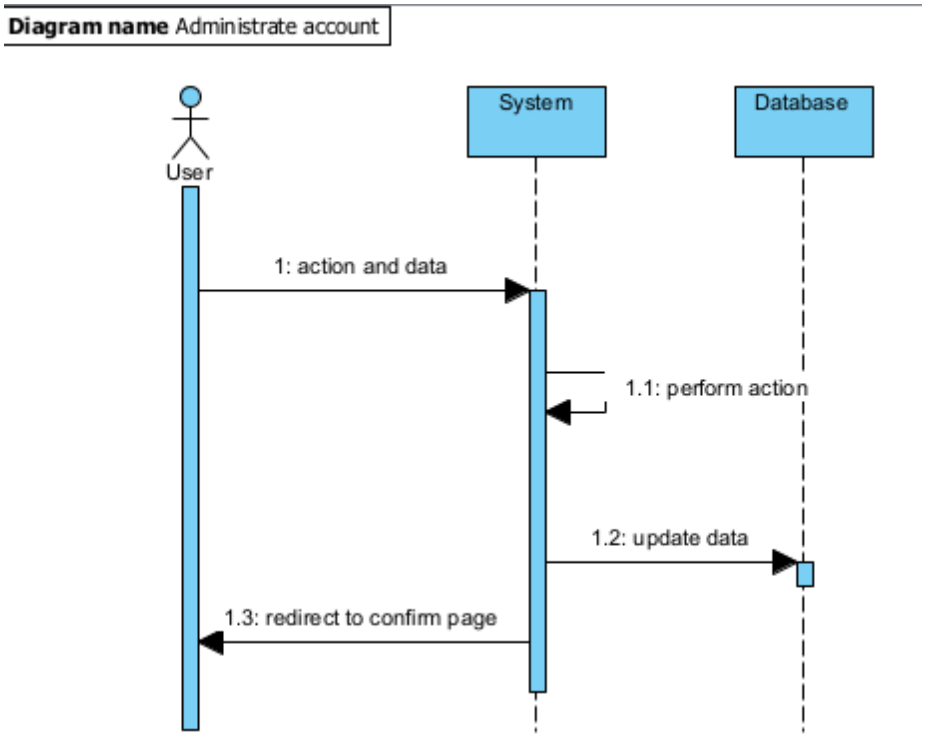

Figure 6: Administrate Account

The user can change his login details or delete his account.
Edit Profile

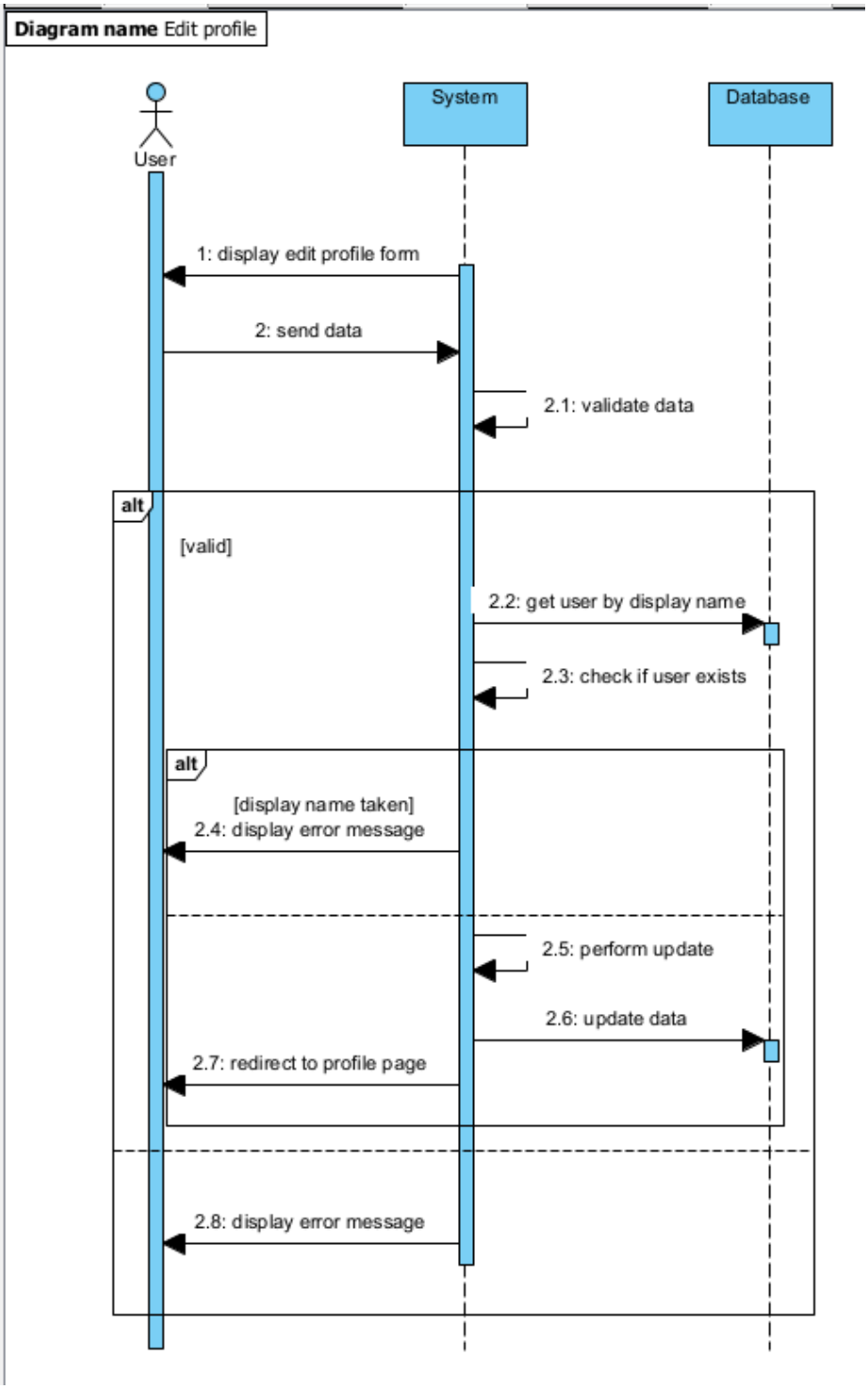

Figure 7: Edit Profile

The user can edit his profile by changing his display name, description, profile picture or featured badge.

View Leaderboard

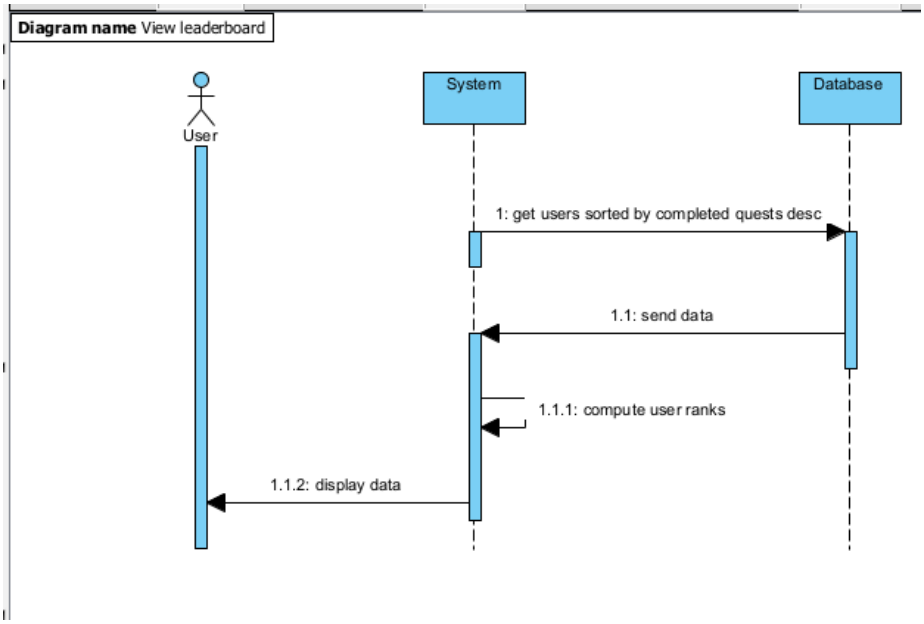

Figure 8: View Leaderboard

The user can visualize the leaderboard, where all the users are ranked based on the number of quests they have completed.

Visit Profile

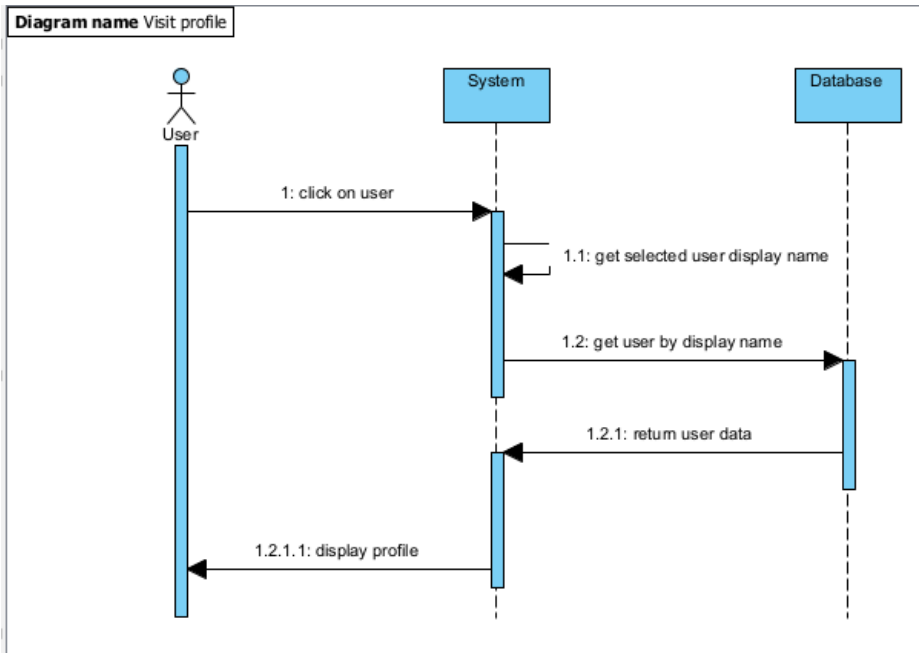

Figure 9: Visit Profile

In the leaderboard, the user can click an username to visit another user's profile.

View Quests

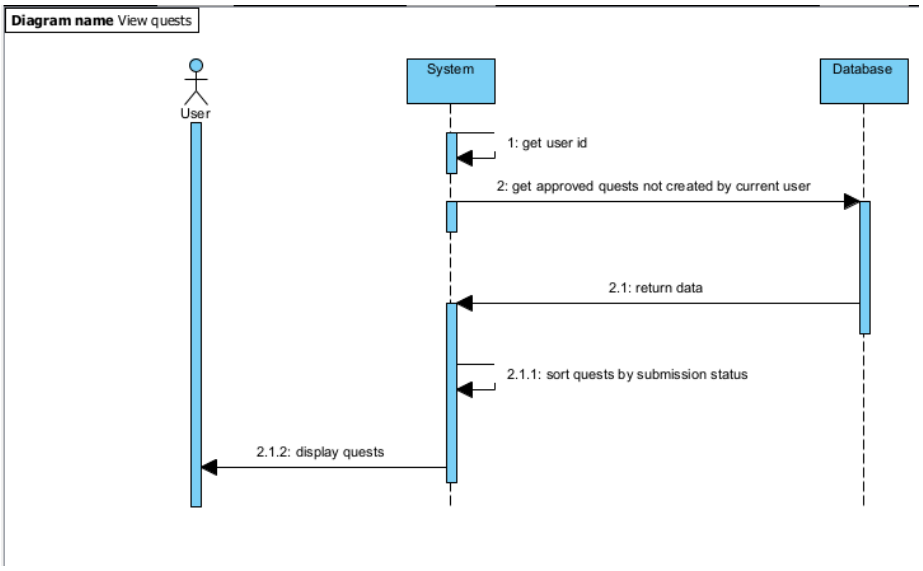

Figure 10: View Quests

The user can view all the quests created by others, sorted by the status of his submissions. The user can click on a quest to view its details and to submit an answer.

Submit Answer

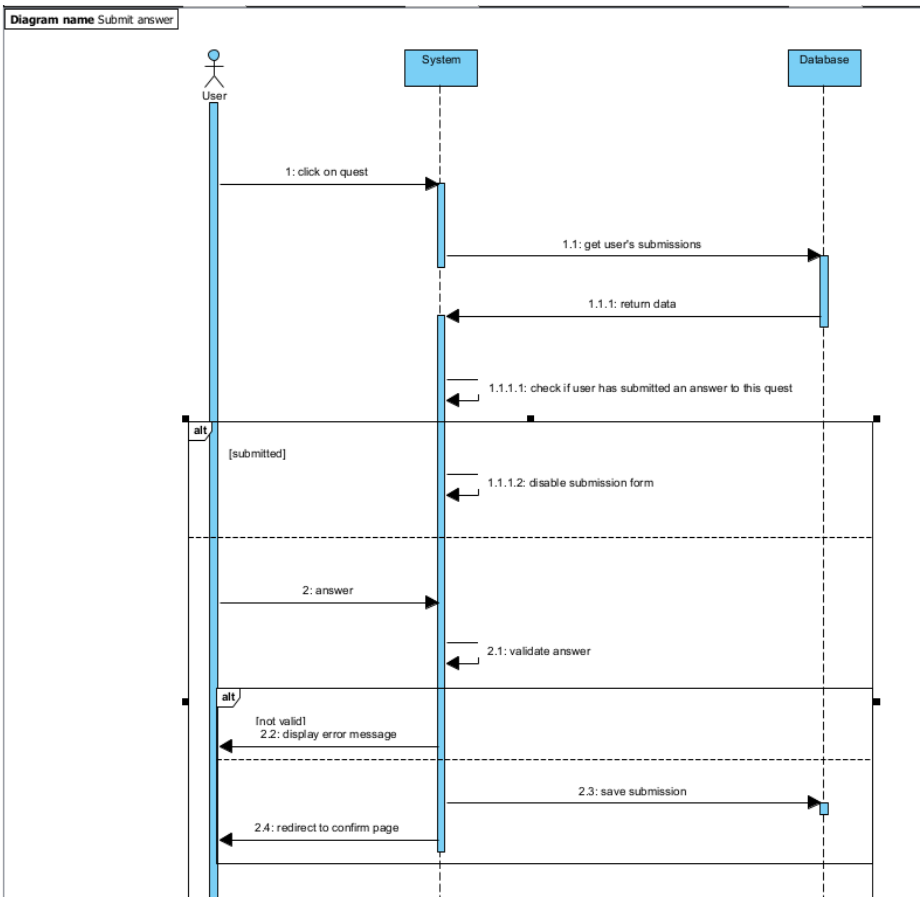

Figure 11: Submit Answer

When viewing a quest's details, the user can submit an answer to it if he did not do it already.

Manage Submissions

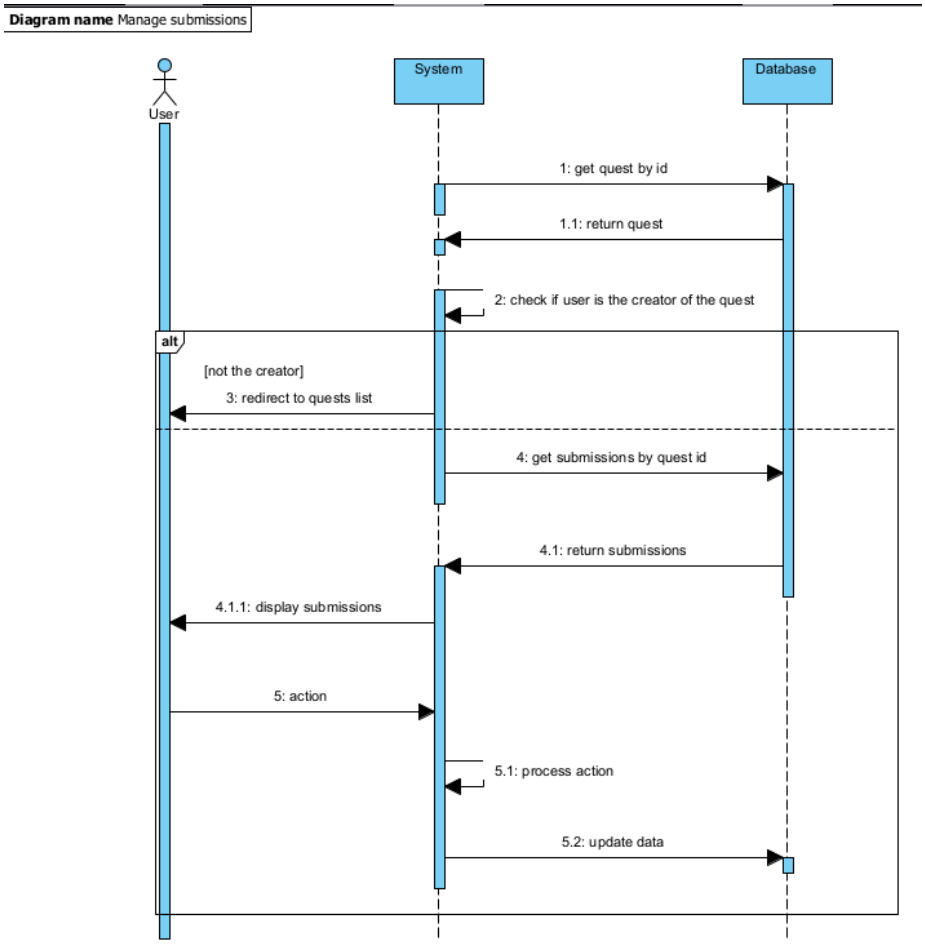

Figure 12: Manage Submissions

The user can view all the submissions to his created quests and approve or reject them.

4 Technologies

This section describes the technologies used in the application's development.

4.1 Spring Boot

Spring Boot is a Java-based framework which allows the developers to develop stand-alone Spring applications that can be just run. Its purpose is to allow the developers to get started with minimum configurations.

4.2 Spring Boot Starter Web

Spring Boot Starter Web pulls in all dependencies related to web development. It uses Spring Web MVC, which is a Java framework used for building web applications. It implements all the basic features of a Spring framework and follows the MVC(Model-View-Controller) design pattern. It uses Tomcat as a default embedded server.

4.3 Spring Boot Starter Email

The Spring Boot Starter Email provides helpful utility functions for sending emails.

4.4 Spring Boot Starter Security

Spring Boot Starter Security pulls in all dependencies related to Spring Security. Spring Security is a framework which provides authentication and authorization.

4.5 Spring Boot Starter Test

Spring Boot Starter Test is used for testing and imports the Spring Boot test modules together with JUnit and other testing libraries.

4.6 MySQL

MySQl is a relational database management system.

4.7 Hibernate

Hibernate is an implementation of Java Persistence API (JPA). It allows mapping objects to tables, ensuring that data is retrieved or stored from the database based on the mapping. It also allows queries through HQL(Hibernate Query Language).

4.8 Hibernate Validator

Hibernate Validator allows the validation of data through annotations.

4.9 Thymeleaf

Thymeleaf is a templating engine which provides full integration with the Spring Framework.

4.10 Spring Boot Dev Tools

Spring Boot Dev Tools includes additional tools that can make the development process easier, such as automatically trigerring a browser refresh when a resource is changed.

5 Architecture

The architecture represents the way an application is organized. The architecture of the application is presented in the following architecture diagram:

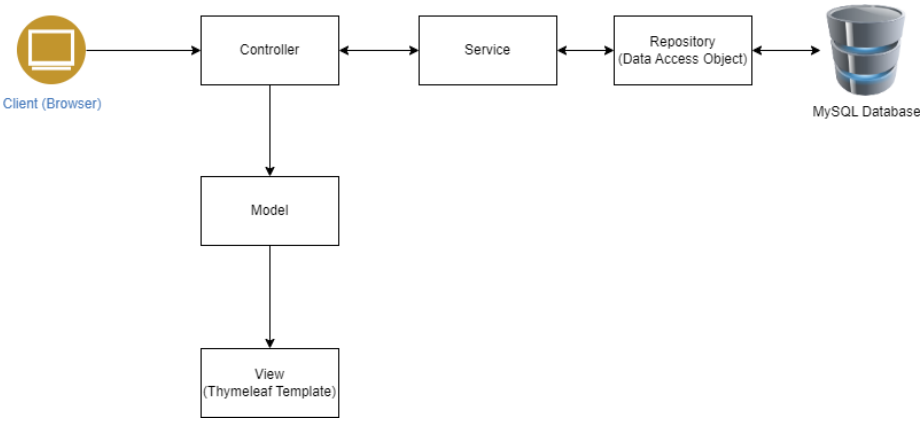

Figure 13: Architecture Diagram

5.1 Controller

The Controller layer receives the requests and displays the results.

5.2 Model

The Model layer contains the data of the application.

5.3 View

The view layer represents how the data is rendered to the view. Thymeleaf templates are used to display the data.

5.4 Service

The Service layer contains the functionality of the application. The services are highly reusable for controllers and other services. They store and retrieve data by delegating methods to the Data Access Objects.

5.4.1 Repository

The Repository layer is represented by Data Access Objects, which provide methods to store or retrieve data from the database.

5.5 MySQL Database

The database layer is responsible for storing the data. The structure of the database can be visualized in the following database diagram:

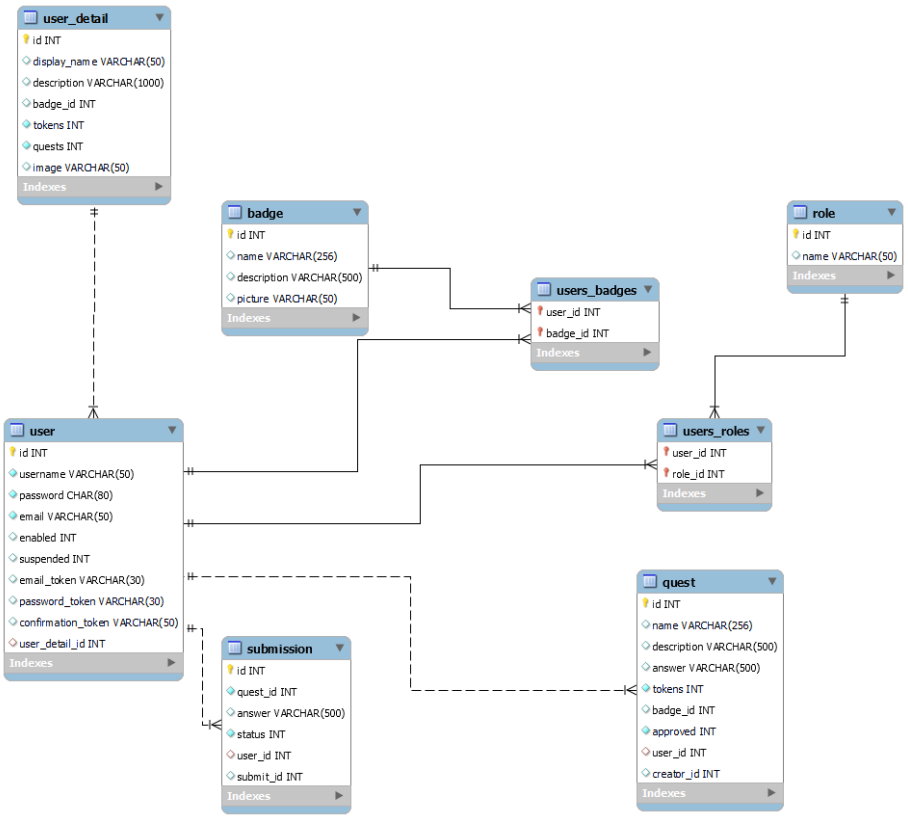

Figure 14: Database Diagram

Every user has one user_detail. Every user can have multiple roles and every role can have multiple users (represented by the join table users_roles). Every user can have multiple badges and every badge can have multiple users (represented by the join table users_badges). Every user can have multiple quests, each quest must have one user (the creator). Every user can have multiple submissions, each submission must have one user.