Software Requirements Specification Design Document

HACKATHON HEROES

presented by:

Minhaj Ur Rehman

SP21 – BSE – 073

Qurat Ul Ain Javed

SP21 – BSE – 067

supervised by:

Dr. Wasif Nisar

Department of Computer Science

Comsats University Islamabad, Wah Campus

Contents

1. Introduction

1.1. System Introduction

Hackathon Heroes is an innovative online platform designed to bring together individuals passionate about technology and coding challenges. Its primary purpose is to provide a centralized space where users can engage in social networking activities while participating in coding competitions. The platform serves as a hub for tech enthusiasts and developers to share ideas, collaborate on projects, and showcase their coding skills. By combining social networking features with coding challenges, Hackathon Heroes aims to foster a vibrant and supportive community where users can learn, grow, and connect with like-minded individuals.

Whether users are beginners looking to improve their coding skills or experienced professionals seeking new challenges, Hackathon Heroes offers a dynamic and inclusive environment for all levels of expertise.

1.2. Background of the System

In the realm of online platforms catering to tech enthusiasts and developers, there are existing solutions that offer features such as social networking and coding challenges. However, many of these platforms often lack seamless integration between these two components, leading to disjointed user experiences. Additionally, some platforms may focus primarily on either social networking or coding challenges, rather than providing a balanced combination of both.

Hackathon Heroes sets itself apart from existing software by offering a unique blend of social networking and coding challenges within a single, cohesive platform. Unlike other platforms that may prioritize one aspect over the other, Hackathon Heroes places equal emphasis on fostering a sense of community and facilitating coding competitions. By seamlessly integrating social networking features like newsfeeds and user profiles with coding challenges and tournaments, Hackathon Heroes creates an immersive environment where users can not only connect with like-minded individuals but also hone their coding skills and compete in real-time challenges. Additionally, Hackathon Heroes prioritizes user-friendly design and accessibility, ensuring that users of all skill levels can easily navigate the platform and participate in its offerings.

1.3. Objectives of the System

- Make a platform that's easy to use, where people can post and also do coding challenges.
- Create a friendly community where people can help each other and learn together.
- Give users lots of features so everyone can find something they like.
- Help people improve their skills, share what they know, and make friends.
- Keep users interested and happy with updates, personal pages, and fun things to do.
- Become a top place for tech fans and coders to show off their skills and win in coding contests.

1.4. Significance of the System

1.1.1 **Skill Enhancement:** Hackathon Heroes provides users with opportunities to improve their coding skills through practice, collaboration, and participation in coding challenges. By engaging in coding competitions and interacting with peers, users can enhance their programming abilities and stay updated with the latest trends and technologies.

1.1.2 **Community Building:** The platform fosters a sense of community among tech enthusiasts and developers, offering a space where they can connect, share knowledge, and support each other. Hackathon Heroes enables users to form meaningful connections, collaborate on projects, and engage in discussions, thus fostering a vibrant and supportive community.

1.1.3 **Innovation and Creativity:** Hackathon Heroes encourages innovation and creativity by providing users with a platform to showcase their coding skills and explore new ideas. Through coding challenges and competitions, users can experiment with different solutions, solve complex problems, and push the boundaries of technology.

1.1.4 **Career Development:** Engaging with Hackathon Heroes can have a positive impact on users' career development. By participating in coding challenges, users can build their portfolios, demonstrate their skills to potential employers, and explore job opportunities within the tech industry.

1.1.5 **Education and Learning:** Hackathon Heroes serves as a valuable educational resource for individuals interested in learning programming and software development. The platform offers tutorials, resources, and mentorship opportunities, enabling users to enhance their knowledge and skills in a supportive environment.

2. Overall Description

2.1. Product Perspective

Hackathon Heroes is a new, self-contained product designed to address the growing demand for a comprehensive platform that combines social networking with coding challenges. Unlike existing systems that may focus solely on either social networking or coding challenges, Hackathon Heroes offers a unique integration of both aspects within a single platform. The product is not intended as a replacement for existing systems but rather as a standalone solution that provides users with a seamless and engaging experience.

The diagram below illustrates the major components of the Hackathon Heroes system and their interconnections:

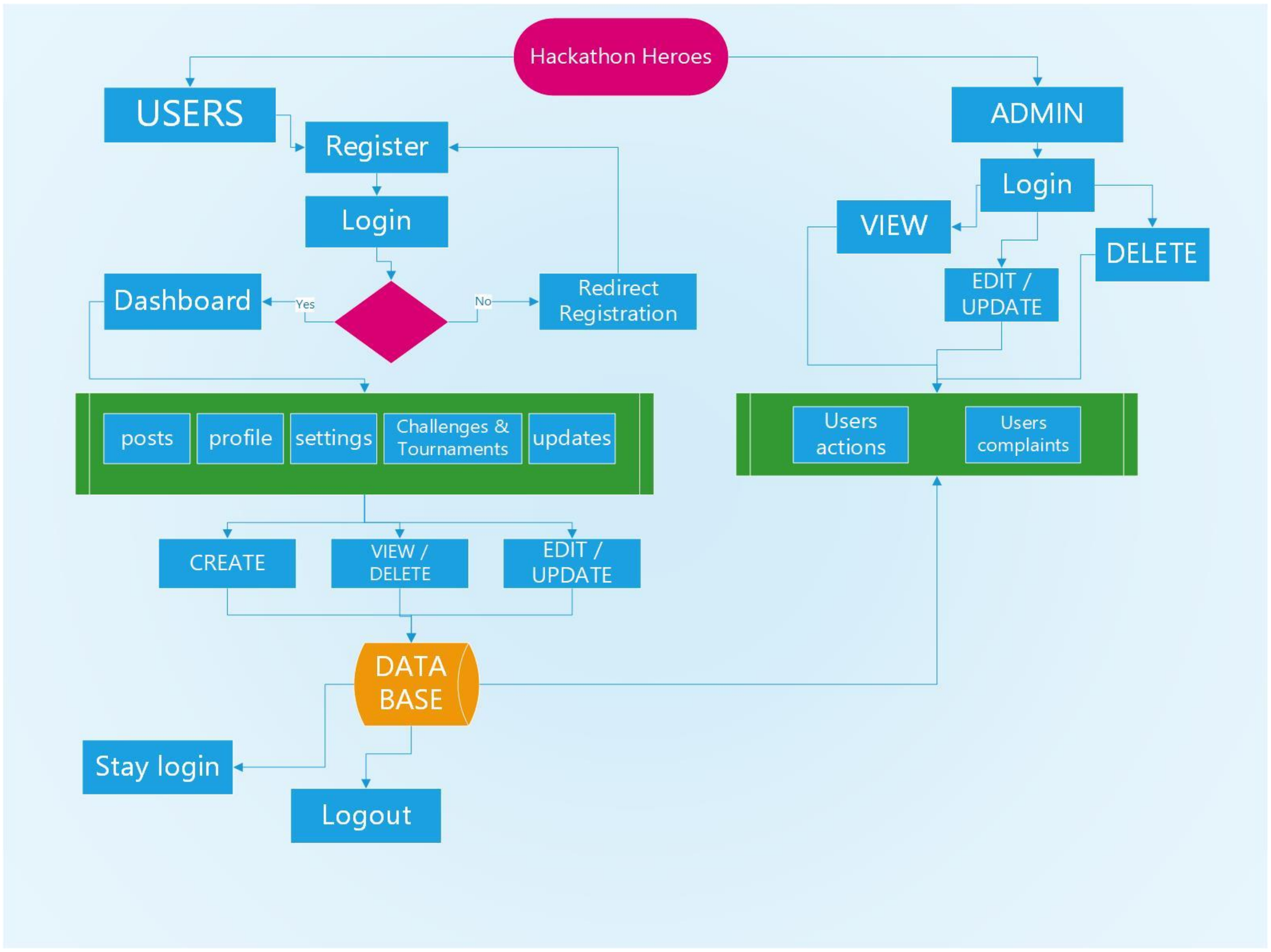


Figure 2.1 System Diagram

2.2. Product Scope

Hackathon Heroes is a website that brings together tech enthusiasts. It lets people sign up, log in, and do things like posting updates, joining coding challenges, and others. But we're not adding online shopping or games. Our goal is to keep the website focused on tech-related activities, so users can have a fun and useful experience without any distractions.

It's aims to be a go-to platform for anyone interested in tech. Users can expect a user-friendly interface where they can easily navigate through features like creating profiles, sharing ideas, and participating in coding tournaments. While we prioritize tech-related activities, we also ensure a safe and inclusive environment for users to connect, learn, and grow together.

2.3. Product Functionality

- **User authentication**: Allow users to create accounts, log in, and manage their profiles securely.
- **Posting updates:** Enable users to create, edit, and delete posts to share updates with the community.
- **Participating in challenges**: Provide users with the ability to join coding challenges and competitions.
- **Managing profiles:** Allow users to edit their profiles, upload profile pictures, and customize their settings.
- **Viewing leaderboard & updates:** Enable users to see updates from other users and stay informed about community activities.
- **Accessing tournaments:** Provide users with access to coding tournaments and competitions.
- **Admin functionalities:** Allow administrators to manage user accounts, challenges, tournaments, and other system settings.

2.4. Users and Characteristics

2.4.1. Regular Users:

- **Characteristics:** Tech enthusiasts, developers, students, professionals.
- **Usage Frequency:** Regularly visit the platform for networking, learning, and participating in coding challenges.
- **Technical Expertise:** Varied, from beginners to advanced programmers.
- **Educational Level:** High school, college, university, or self-taught.
- **Experience:** Diverse levels of experience in coding and tech-related activities.

2.4.2. Administrators:

- **Characteristics:** Platform moderators, administrators, or site managers.
- **Usage Frequency:** Regularly manage user accounts, challenges, tournaments, and system settings.
- **Technical Expertise:** Proficient in using administrative tools and managing online platforms.
- **Security or Privilege Levels:** Have elevated privileges for managing system functionalities.
- **Experience:** Experienced in managing online communities or platforms.

The most important users for this product are the regular users who actively participate in coding challenges, engage with the community, and contribute to the platform's content. Administrators are also crucial for maintaining the platform, ensuring its smooth operation, and addressing any issues that may arise. Their roles are essential for the overall success and sustainability of Hackathon Heroes.

2.5. Operating Environment

Hackathon Heroes will operate in a web-based environment, accessible through standard web browsers on desktop and mobile devices. The system will be compatible with the following operating systems:

- Windows (Windows 7 and above)
- macOS (OS X 10.11 and above)
- Linux distributions
- Mobile operating systems (iOS, Android)

The minimum platform requirements for Hackathon Heroes include:

- Web Browser: Google Chrome, Mozilla Firefox, Safari, Microsoft Edge
- Internet Connection: Broadband connection recommended for optimal performance.
- Screen Resolution: Minimum 1280x768 resolution for desktop browsers.
- Hardware: Standard desktop or laptop computer, or mobile device with sufficient processing power and memory to support web browsing activities.

3. Specific Requirements

3.1. Functional Requirements

3.1.1**. User Authentication:**

- Allow users to create accounts with unique usernames and passwords.
- Provide a secure login mechanism for users to access their accounts.
- Implement password recovery/reset functionality for user account security.

3.1.2**. Posting Updates:**

- Enable users to create new posts with text, images, and other multimedia content.
- Allow users to edit or delete their own posts.
- Provide options for users to like, comment on, and share posts.

3.1.3. **Participating in Challenges:**

- Display a list of available coding challenges and tournaments.
- Allow users to join challenges and competitions.
- Provide a submission mechanism for users to submit their code solutions.

3.1.4. **Managing Profiles:**

- Allow users to edit their profile information, including username, profile picture, and bio.
- Provide settings for users to manage privacy and notification preferences.
- Display user profiles with information such as joined challenges, posts, and achievements.

3.1.5. **Viewing Newsfeed:**

- Display a personalized newsfeed with updates from users the current user follows.
- Implement algorithms to prioritize and organize newsfeed content based on user interactions.
- Allow users to filter newsfeed content based on preferences or categories.

3.1.6. **Accessing Tournaments:**

- Display a list of upcoming and ongoing coding tournaments and competitions.
- Provide detailed information about each tournament, including rules, prizes, and participants.
- Allow users to register for tournaments and track their progress and results.

3.1.7**. Admin Functionalities:**

- Provide administrators with tools to manage user accounts, including user registration and account suspension.
- Allow administrators to create, edit, and delete coding challenges and tournaments.
- Implement moderation features to monitor and address inappropriate content or behavior.

3.2. Behaviour Requirements

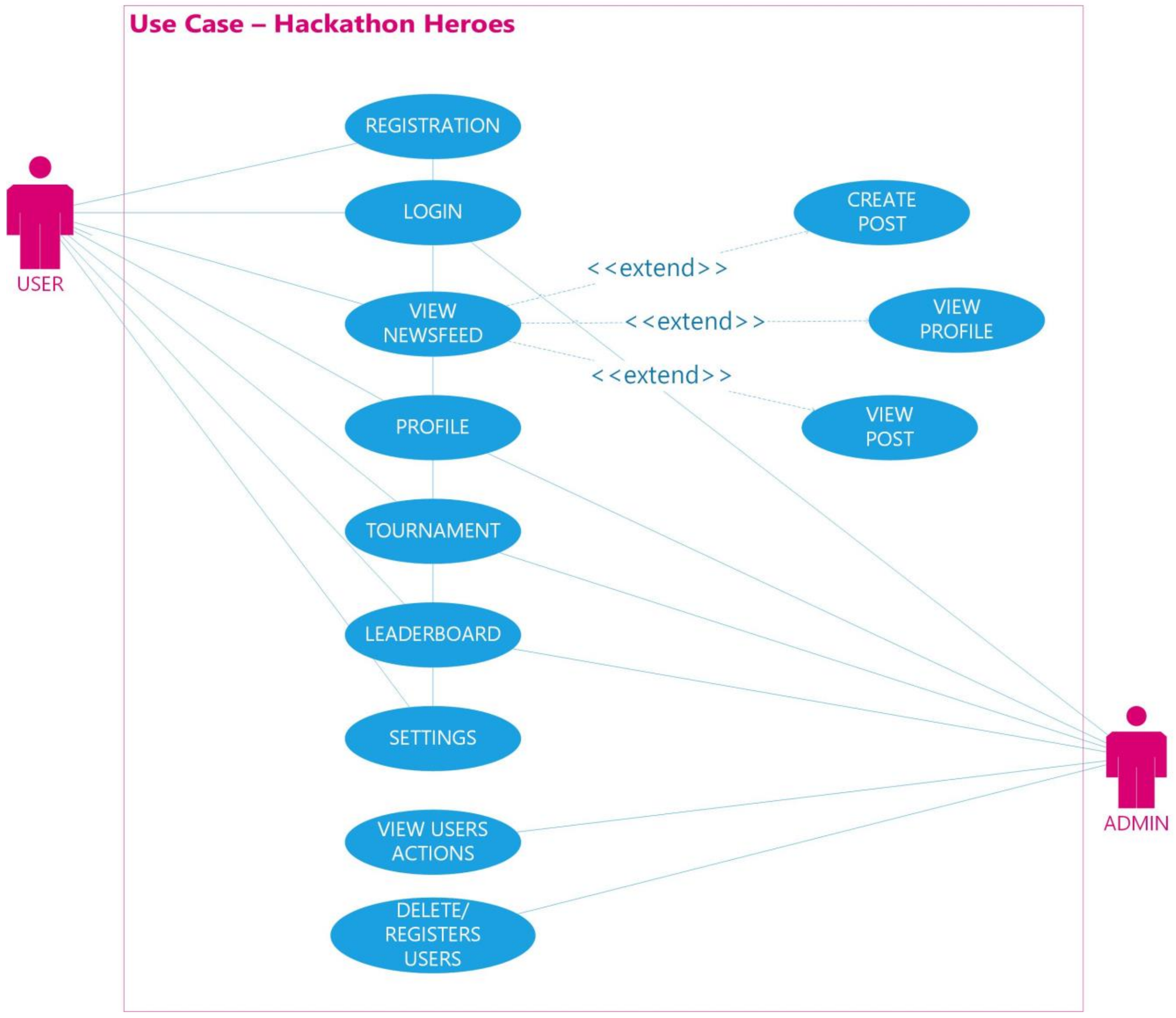


Figure 3.2: Behaviour View

Table 1: User registration

Use Case ID	001
Use Case Name	User Registration
Goal	To create a new account on a platform.
Actors	User

Pre-conditions	User is not registered on this website.
Post-conditions	User's account is created successfully.
Flow of events	● User navigates to the registration page. ● User fills in the required details such as username, email, and password. ● User submits the registration form. ● System validates the input data. ● If validation is successful, the system creates a new user account. ● User receives a confirmation message.

Table 2: User login

Use Case ID	002
Use Case Name	User Login
Goal	To access the platform using existing credentials.
Actors	User
Pre-conditions	User account is created**.**
Post-conditions	User successfully logs into the system.
Flow of events	● User navigates to the login page. ● User enters their username/email password. ● User submits the login form. ● System validates the credentials. ● If credentials are valid, the user is redirected to their dashboard. ● If credentials are invalid, the system displays an error message.

Table 3: Posting Updates

	003

Use Case ID	
Use Case Name	Posting Updates
Goal	To share thoughts, images, or updates with others.
Actors	Logged-in User
Pre-conditions	User is logged into the system
Post-conditions	Post is visible to others on the newsfeed.
Flow of events	● User goes to the "Create Post" section. ● User writes their update or shares an image. ● User can add tags or location if desired. ● User clicks on the "Post" button. ● System verifies the post content. ● If content is valid, post appears on the newsfeed for others to see.

Table 4: Participating in Challenges

Use Case ID	004
Use Case Name	Participating in challenges
Goal	To engage in coding challenges and competitions.
Actors	Logged-in User
Pre-conditions	User is logged into the system
Post-conditions	User's performance is recorded in challenge history.
Flow of events	● User selects a coding challenge from the available list. ● User reads the challenge description and requirements. ● User writes and submits their solution within the specified time. ● System evaluates the solution for correctness and efficiency. ● If solution meets requirements, user earns points and status updates.

Table 5: Managing Profile

Use Case ID	005
Use Case Name	Managing Profile
Goal	To update personal information and settings.
Actors	Logged-in User
Pre-conditions	User is logged into the system
Post-conditions	Profile information is updated.
Flow of events	● User goes to the profile settings section. ● User can update profile picture or cover photo. ● User can accept or reject friend requests. ● User can view their challenge history and performance. ● User saves changes to update their profile.

Table 6: Viewing Newsfeed

Use Case ID	006
Use Case Name	Viewing Newsfeed
Goal	To see updates and posts from other users.
Actors	Logged-in User
Pre-conditions	User is logged into the system
Post-conditions	User can view and interact with newsfeed content.
Flow of events	● User accesses the newsfeed section. ● User scrolls through the feed to see posts from others. ● User can like, comment, or share posts. ● User can also filter posts based on

	tags or keywords.

Table 7: Accessing Tournament

Use Case ID	007
Use Case Name	Accessing Tournament
Goal	To participate in coding tournaments and competitions.
Actors	Logged-in User
Pre-conditions	User is logged into the system
Post-conditions	User successfully registers for selected tournaments.
Flow of events	● User visits the tournaments section. ● User selects a tournament they want to participate in. ● User chooses a subscription plan if required. ● User fills in necessary details for registration. ● User submits the registration form. ● System confirms registration and provides access to the tournament.

Table 8: Admin Functionalities

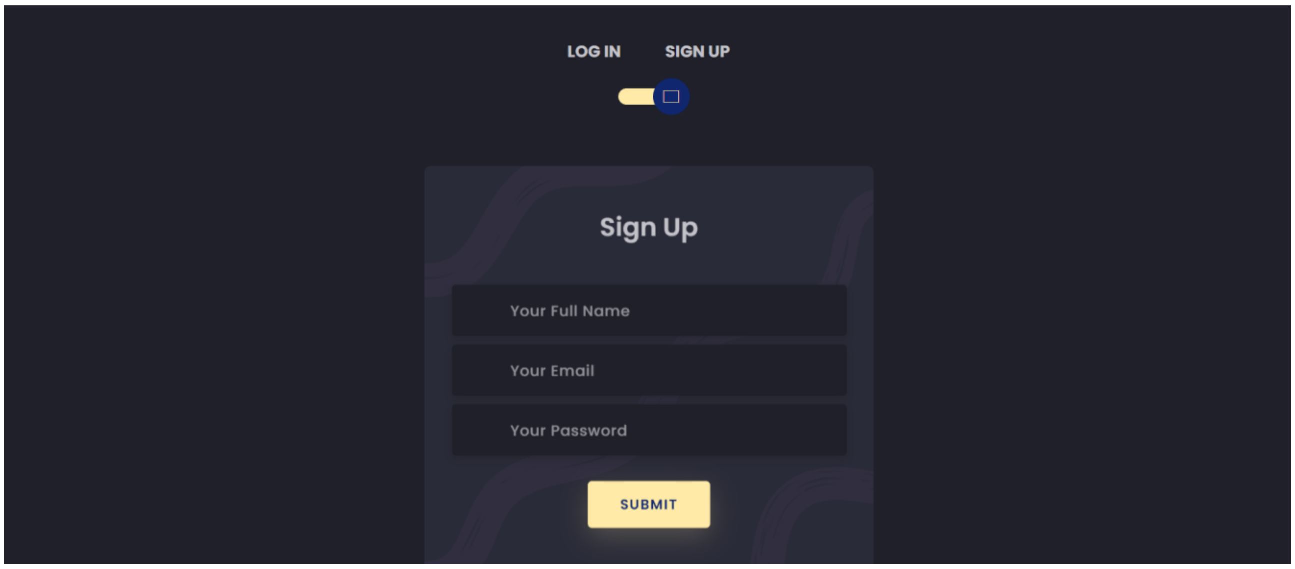
Use Case ID	008

Use Case Name	Admin Functionalities
Goal	To manage users, posts, and challenges on the platform.
Actors	Logged-in User
Pre-conditions	User is logged into the system
Post-conditions	Changes made by admin are reflected in the system.
Flow of events	● Admin accesses the admin panel. ● Admin can view user accounts and activity. ● Admin can moderate posts by editing, deleting, or hiding them. ● Admin can create or modify coding challenges and tournaments. ● Admin can review and approve user submissions and performances.

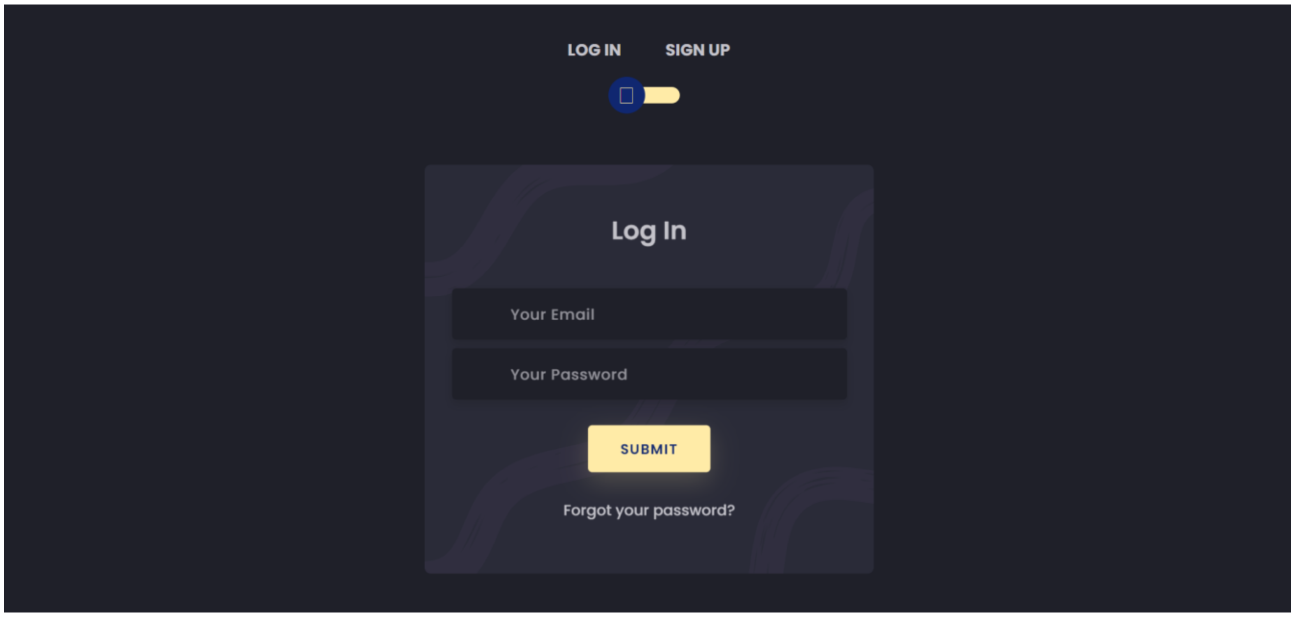
3.3. External Interface Requirements

3.3.1. User Interfaces

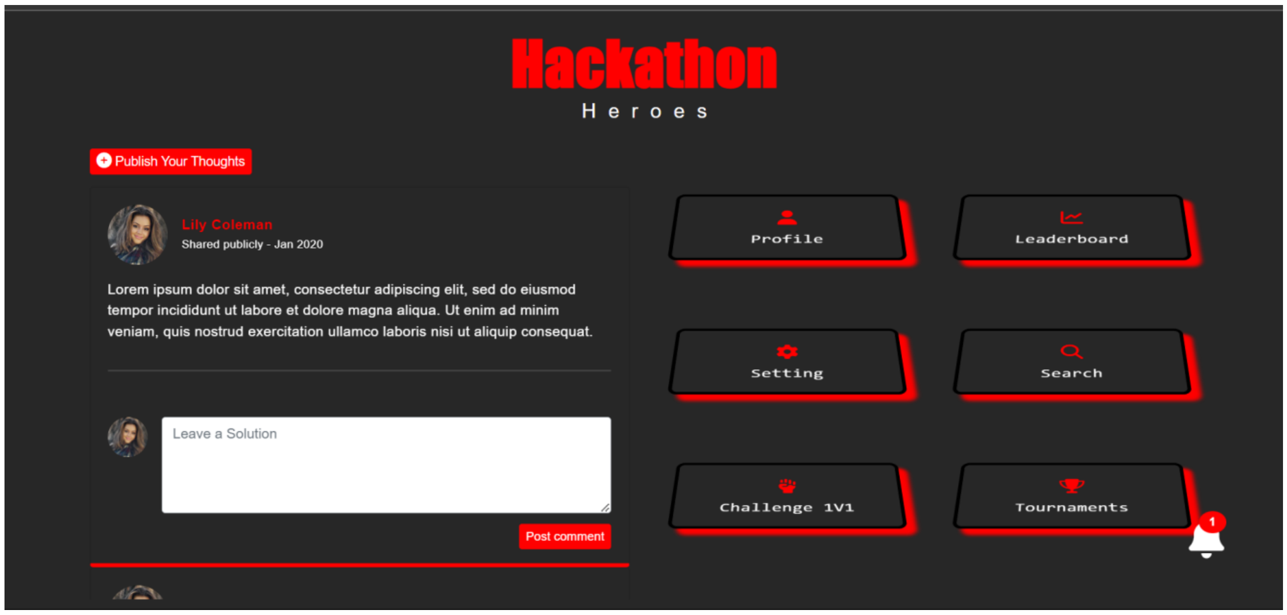
REGISTERATION PAGE



LOGIN PAGE



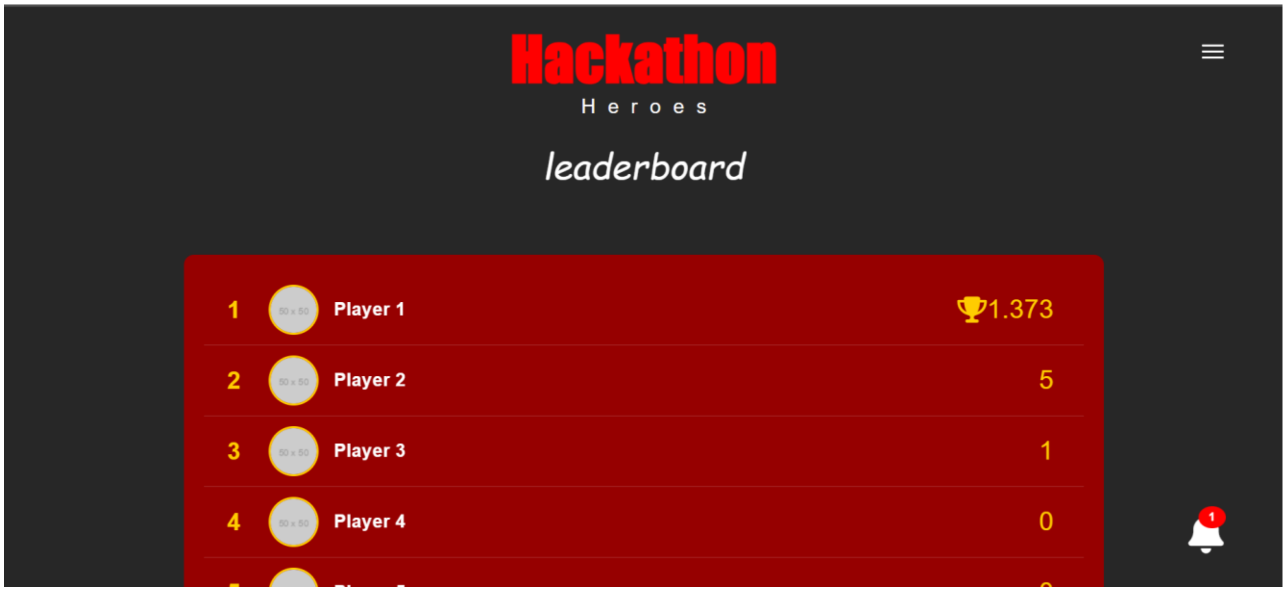
HOME PAGE



Lily Coleman
Shared publicly - Jan 2020

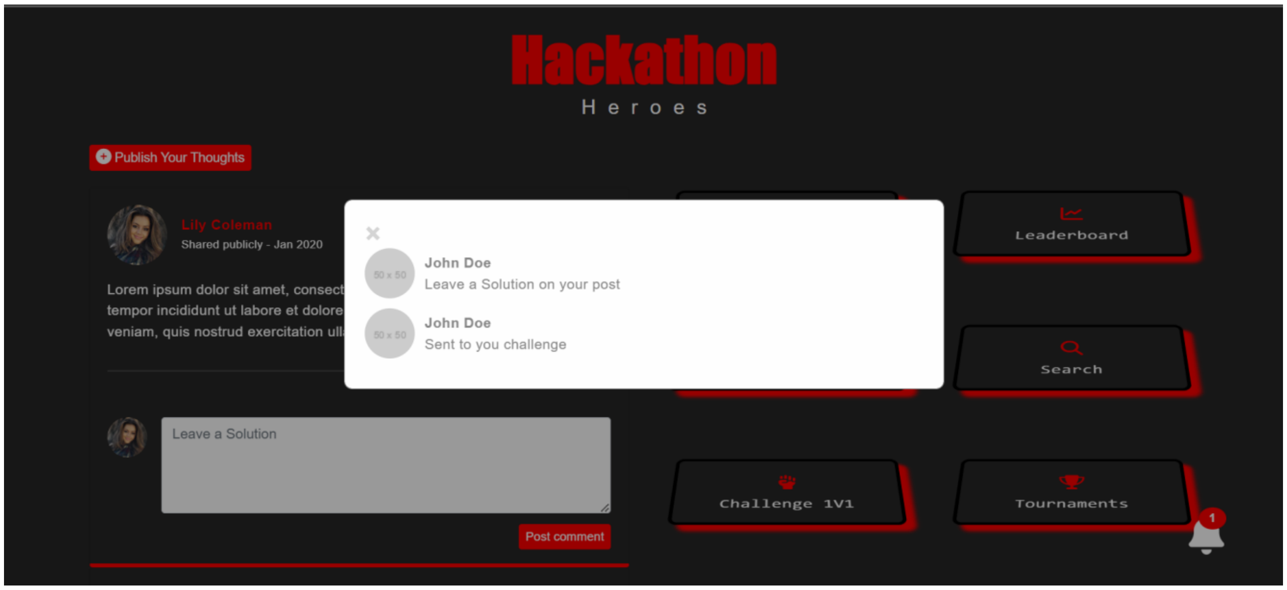
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip consequat.

LEADERBPOARD PAGE



1	Player 1	1.373
2	Player 2	5
3	Player 3	1
4	Player 4	0

NOTIFICATIONS

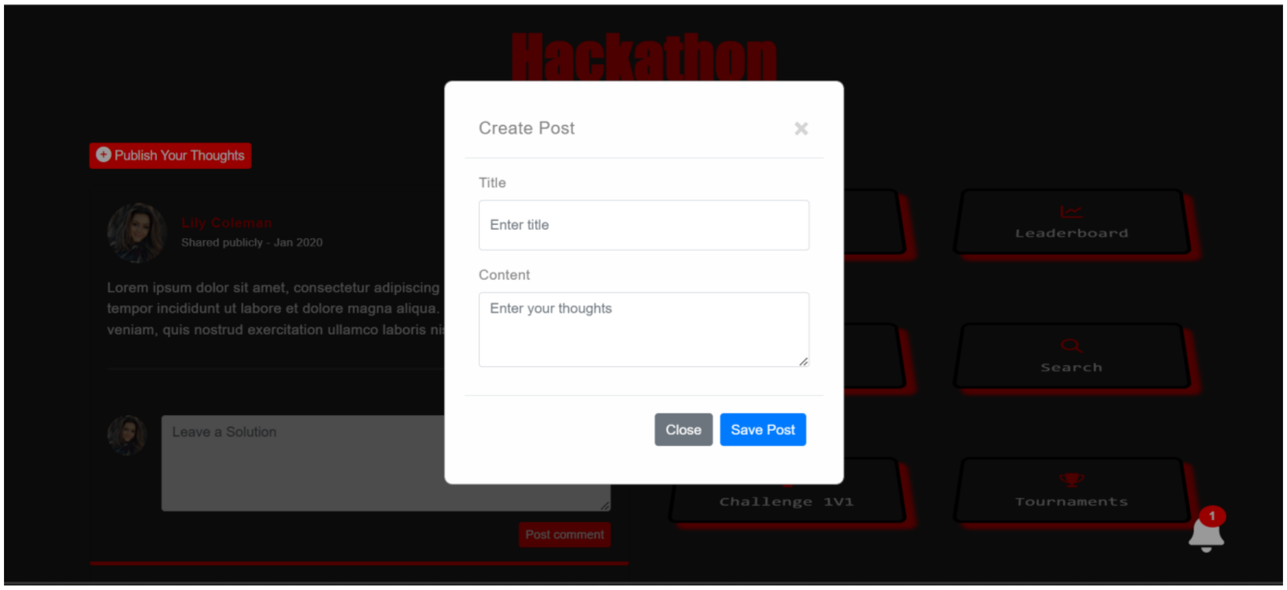


Lorem ipsum dolor sit amet, consect tempor incididunt ut labore et dolore veniam, quis nostrud exercitation ull

John Doe
Leave a Solution on your post

John Doe
Sent to you challenge

POST



Lorem ipsum dolor sit amet, consectetur adipiscing
tempor incididunt ut labore et dolore magna aliqua.
veniam, quis nostrud exercitation ullamco laboris ni

Create Post

Title

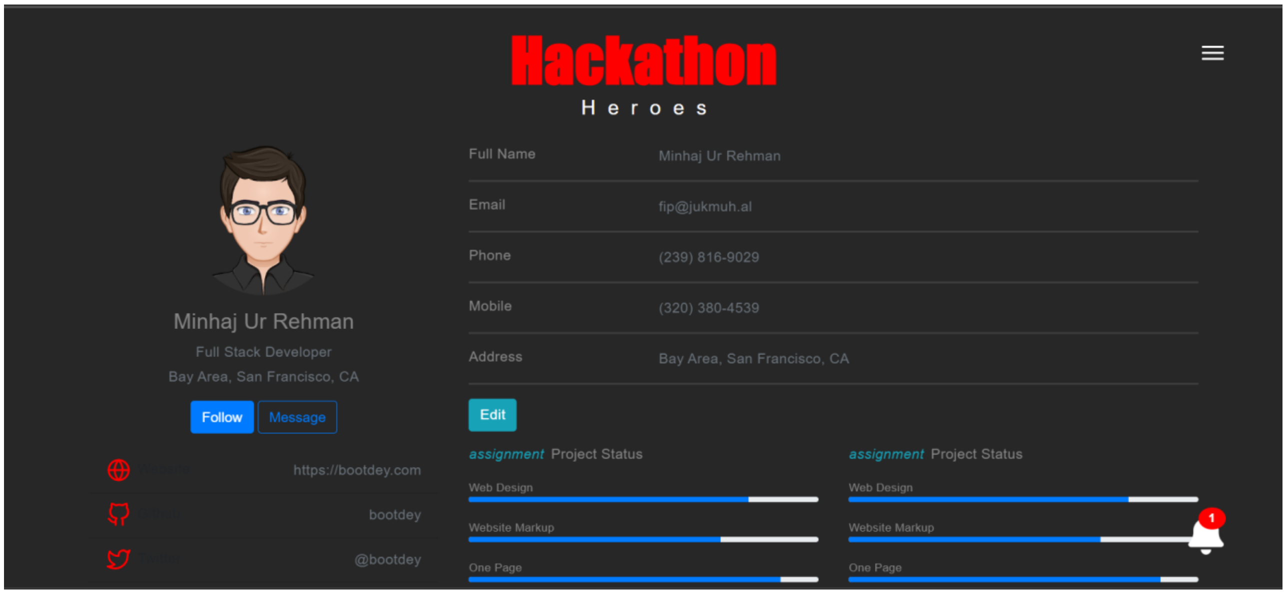
Enter title

Content

Enter your thoughts

Close Save Post

PROFILE PAGE



Minhaj Ur Rehman
Full Stack Developer
Bay Area, San Francisco, CA

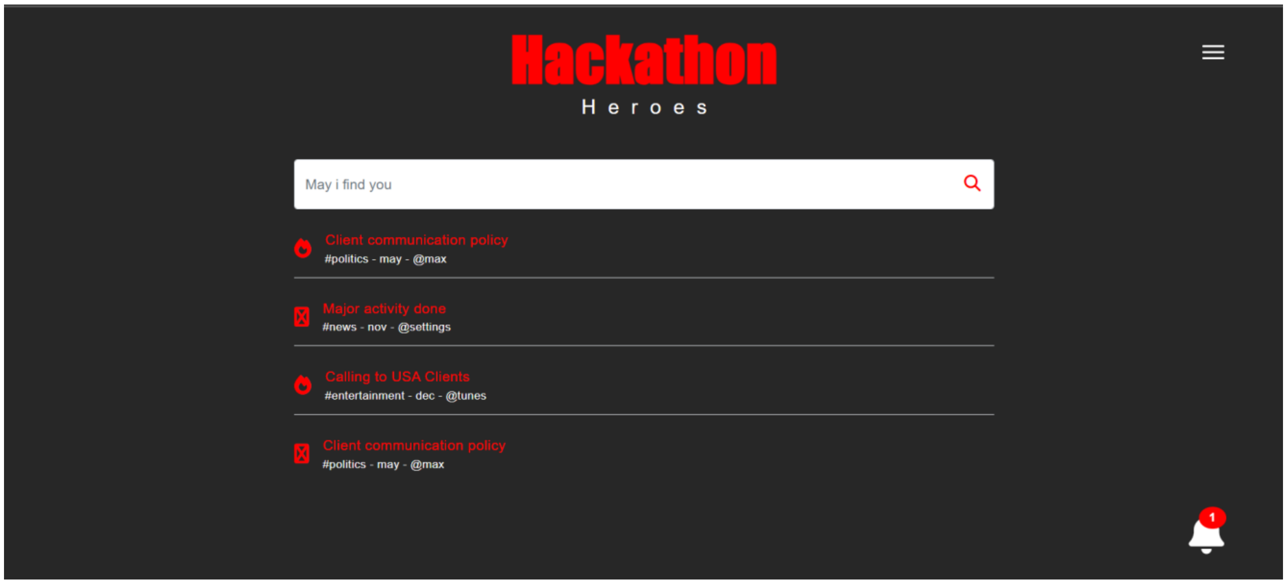
https://bootdey.com
bootdey
@bootdey

Full Name	Minhaj Ur Rehman
Email	fip@jukmuh.al
Phone	(239) 816-9029
Mobile	(320) 380-4539
Address	Bay Area, San Francisco, CA

assignment Project Status
Web Design
Website Markup
One Page

assignment Project Status
Web Design
Website Markup
One Page

SEARCH PAGE



Hackathon
Heroes

May i find you

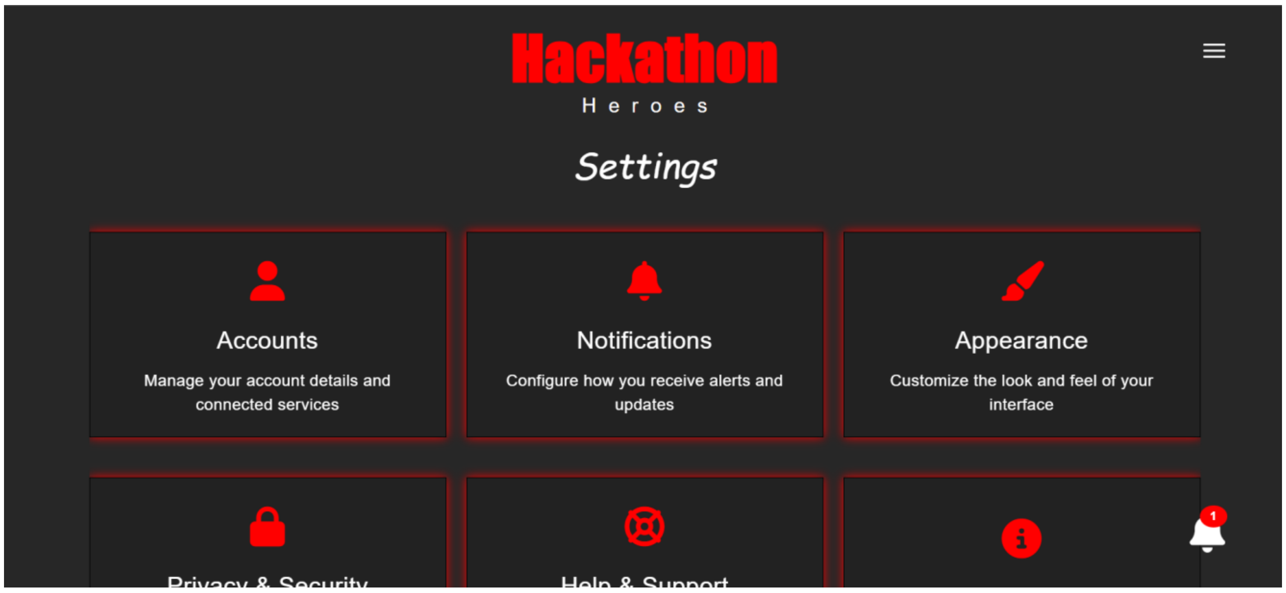
Client communication policy
#politics - may - @max

Major activity done
#news - nov - @settings

Calling to USA Clients
#entertainment - dec - @tunes

Client communication policy
#politics - may - @max

SETTING PAGE



Hackathon
Heroes

Settings

Accounts
Manage your account details and connected services

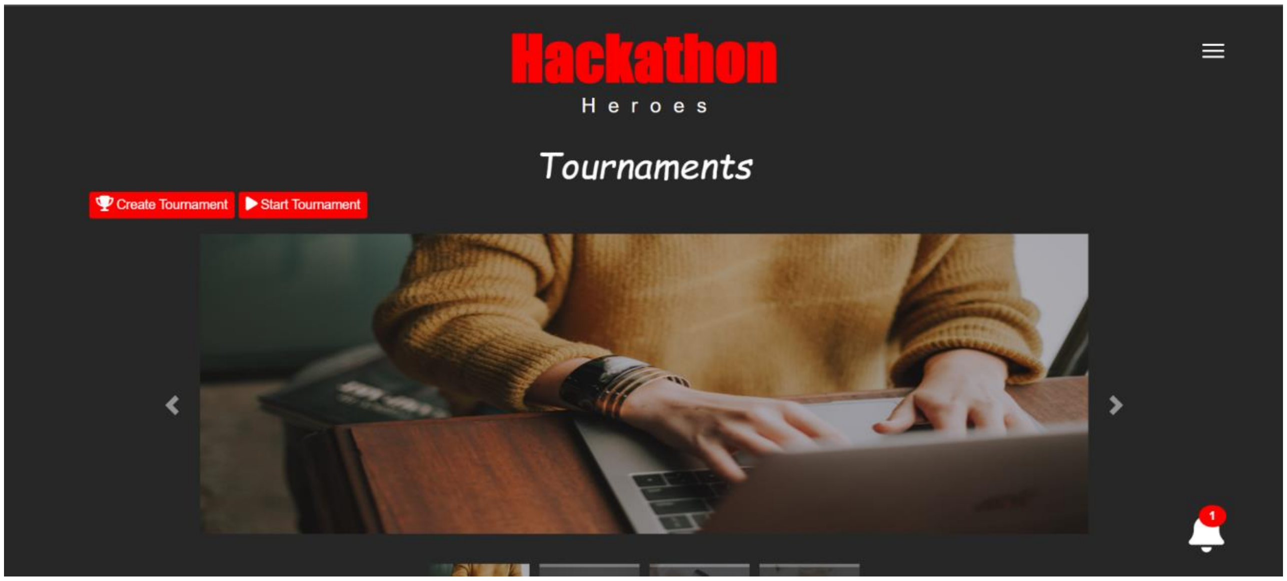
Notifications
Configure how you receive alerts and updates

Appearance
Customize the look and feel of your interface

Privacy & Security

Help & Support

TOURNAMENT PAGE



3.3.2. Hardware Interfaces

The website does not require any hardware component.

3.3.3. Software Interfaces

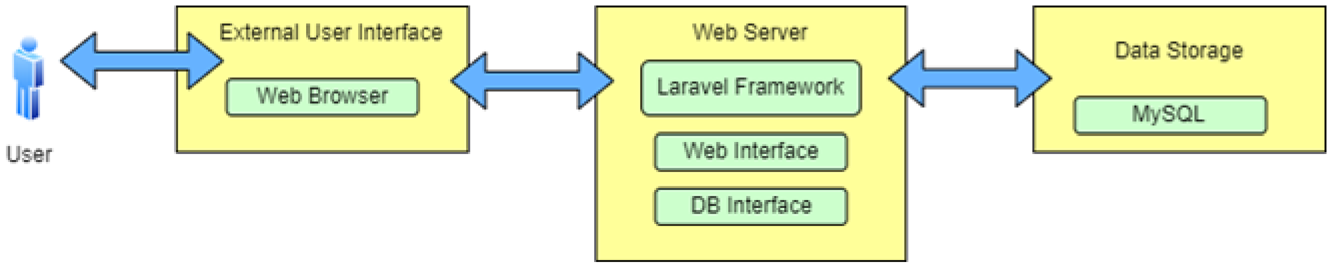


Figure 1.3.3: Software Interfaces View

The software interface of the website should follow the Model-View-Controller (MVC). The interface must be able to connect to a database to store data.

3.3.4. Communications Interfaces

Users will get emails about important things like signing up or changing their passwords. The website will work well on different internet browsers like Google Chrome or Safari. When you use the website, it talks to the main computer (server) using a special language called HTTP. This language helps keep things safe when sending private info, like passwords. When you fill out forms on the website, they'll be easy to understand and tell you if something's wrong. Emails and messages from the website will always look the same, so you know what to expect. And everything will be super safe and fast, making sure your info gets to where it needs to go quickly and securely.

This way, users can trust that their information is safe, and the website works smoothly for everyone.

4. Other Non-functional Requirements

4.1. Performance Requirements

- **Speedy Loading:** Pages should load quickly, so users don't have to wait long.
- **Responsive:** When users click or type something, the website should react right away.
- **Handle Many Users:** Even if lots of people are using the site at once, it should still work smoothly.
- **Efficient Data Use:** The website should use internet data wisely, so it doesn't use up too much.
- **Smooth Graphics:** Images and animations should look nice and move smoothly without any glitches.
- **Stable Connection:** Even if the internet isn't perfect, the website should still work okay without crashing.
- **Easy on Devices:** The website shouldn't drain too much battery or make devices get too hot.
- **Consistent Experience:** No matter if users are on a phone, tablet, or computer, the website should work the same way.

4.2. Safety and Security Requirements

- **Secure Logins:** Making sure only the right people can access the website with usernames and passwords.
- **Protecting Personal Info**: Keeping things like names, emails, and passwords safe from hackers or bad guys.
- **Safe Transactions:** When users buy something or share payment info, it should be encrypted so no one else can see it.

- **Guarding Against Viruses:** Checking files and data for viruses or harmful stuff to keep users' devices safe.
- **Preventing Attacks:** Stopping hackers from breaking into the website and causing trouble or stealing info.
- **Privacy Protection:** Making sure users' private messages, posts, and data aren't shared with anyone they don't want.
- **Regular Updates:** Keeping the website's security features up to date to stay ahead of new threats.
- **Emergency Plans:** Having backup systems in place in case something goes wrong, like a server crashing or an attack happening.

4.3. Software Quality Attributes

- **Reliability:** The website should work smoothly without crashing or causing errors.
- **Usability:** It should be easy for users to figure out how to use the website and find what they need.
- **Maintainability:** Developers should be able to fix bugs and add new features without too much trouble.
- **Scalability:** The website should be able to handle more users and data as it grows without slowing down.
- **Portability:** It should work well on different devices and browsers without any problems.
- **Flexibility:** The website should be able to adapt to changes or updates without breaking anything.
- **Accuracy:** Information on the website should be correct and up to date.
- **Consistency:** The website should look and behave the same way across different pages and sections.

5. Design Description

5.1.Composite Viewpoint

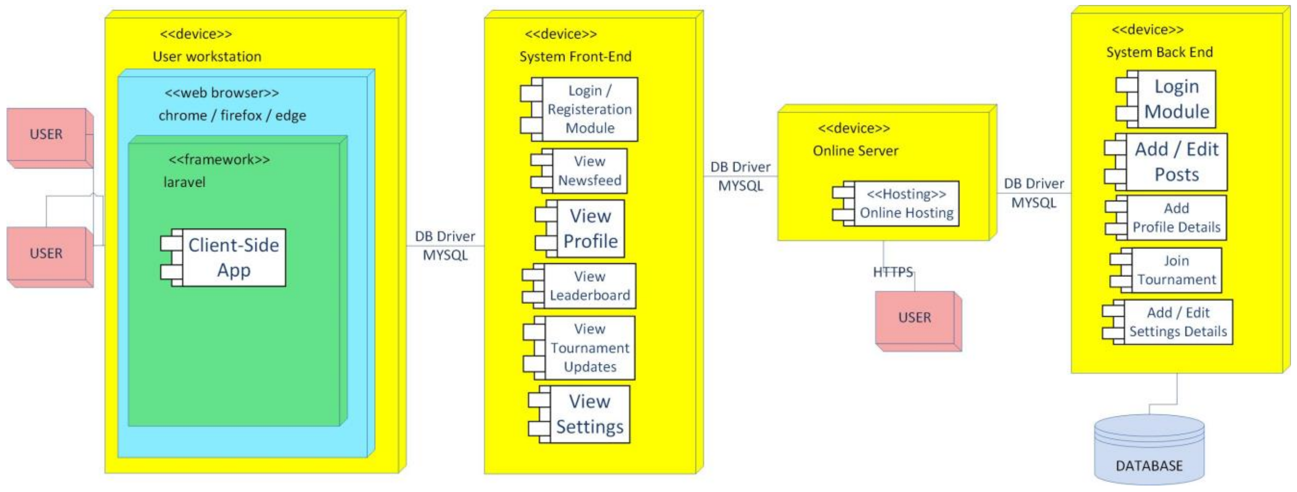


FIGURE 5.1: Composite Viewpoint

5.2.Logical Viewpoint

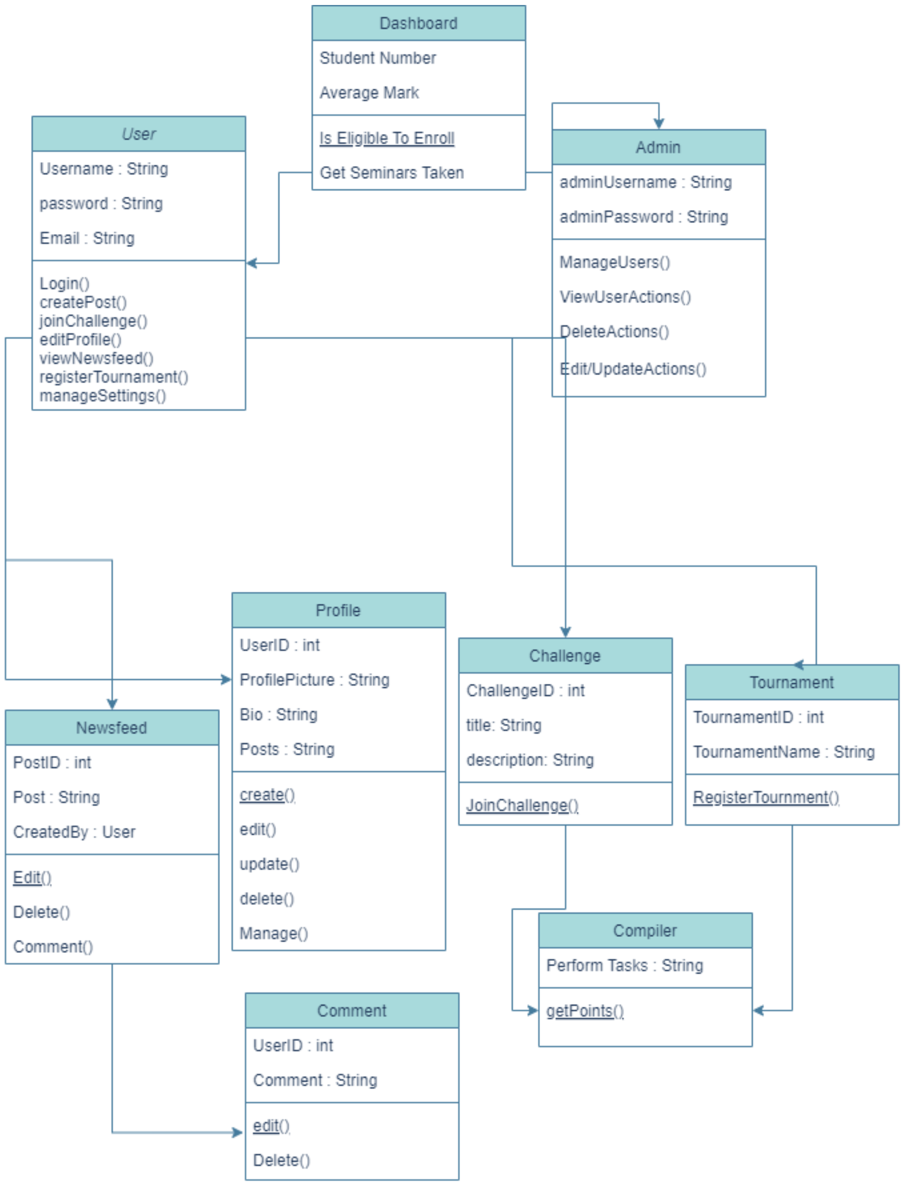


Figure 5.3: Logical View

5.3.Information Viewpoint

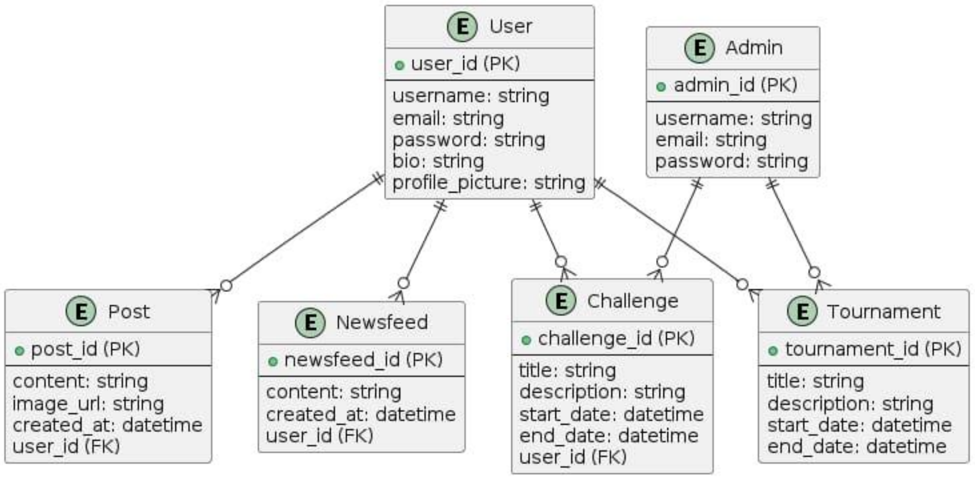
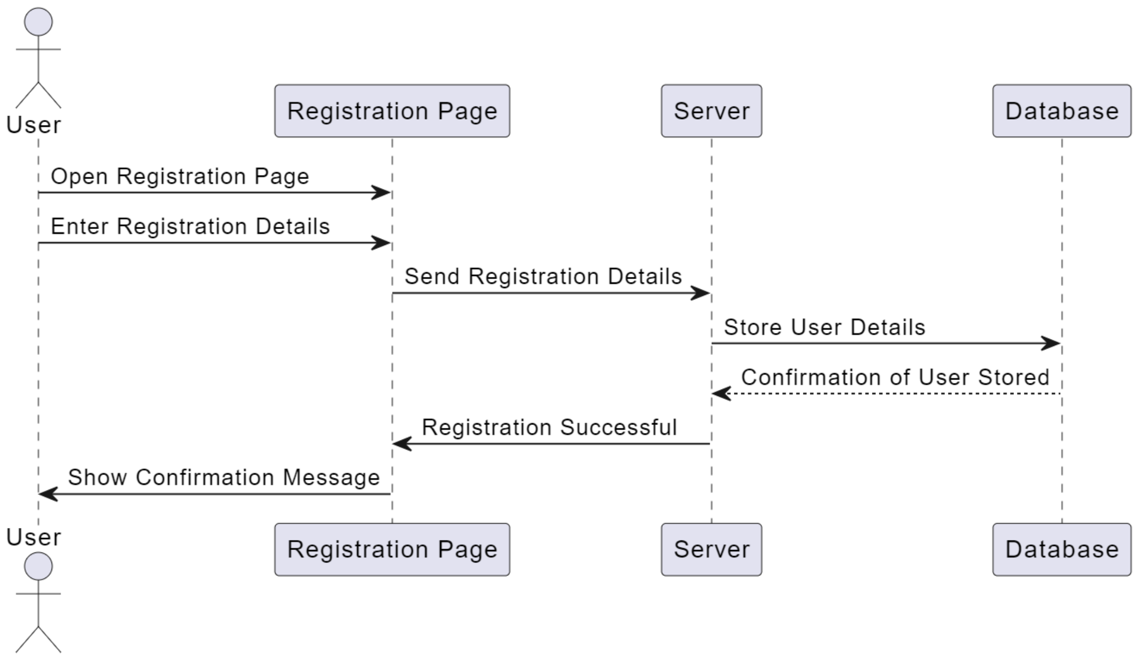


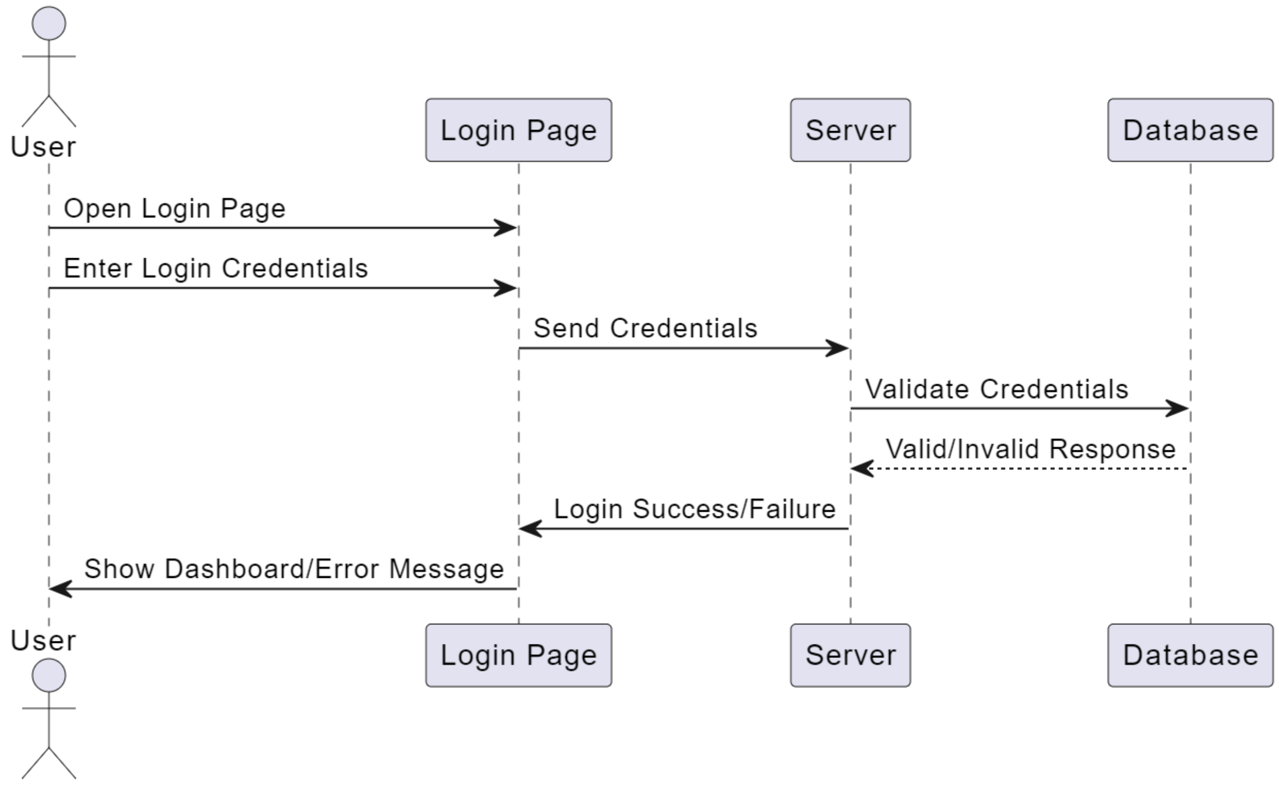
Figure 5.3: Information Viewpoint

5.4. Interaction Viewpoint

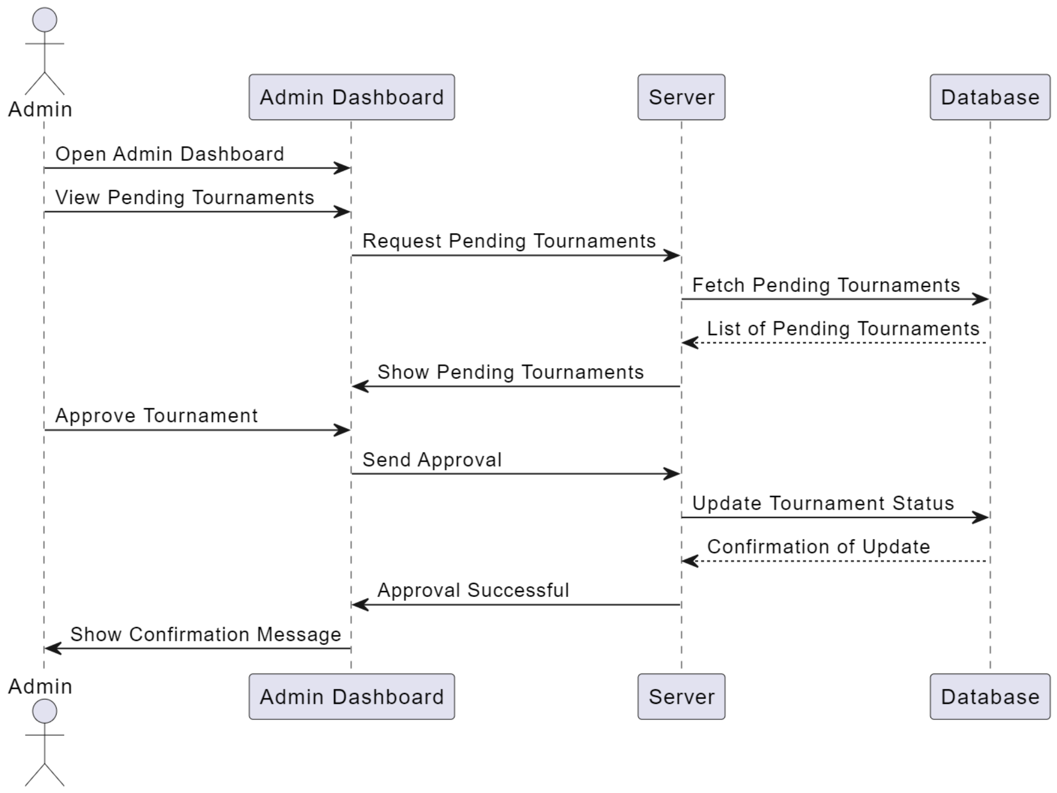
5.4.1.Sequence Diagram For User Registration


User
Registration Page
Server
Database
Open Registration Page
Enter Registration Details
Send Registration Details
Store User Details
Confirmation of User Stored
Registration Successful
Show Confirmation Message

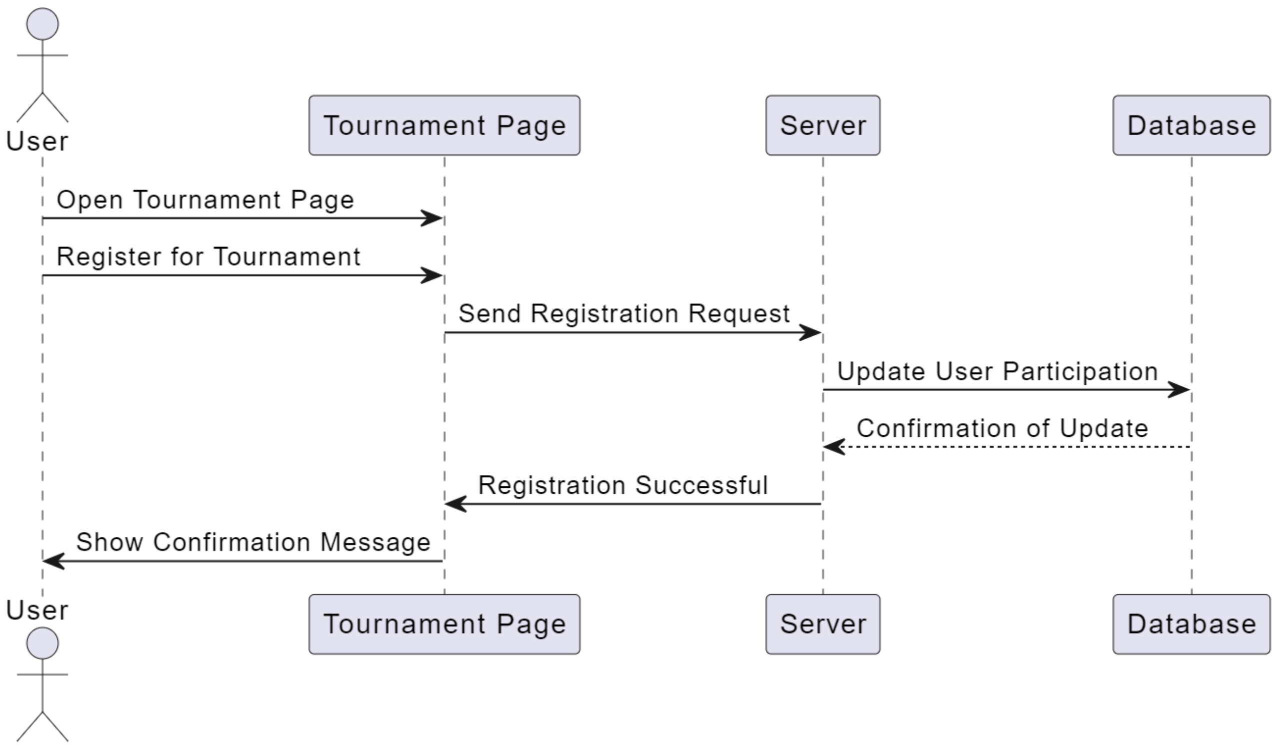
5.4.2.Sequence Diagram For User Login



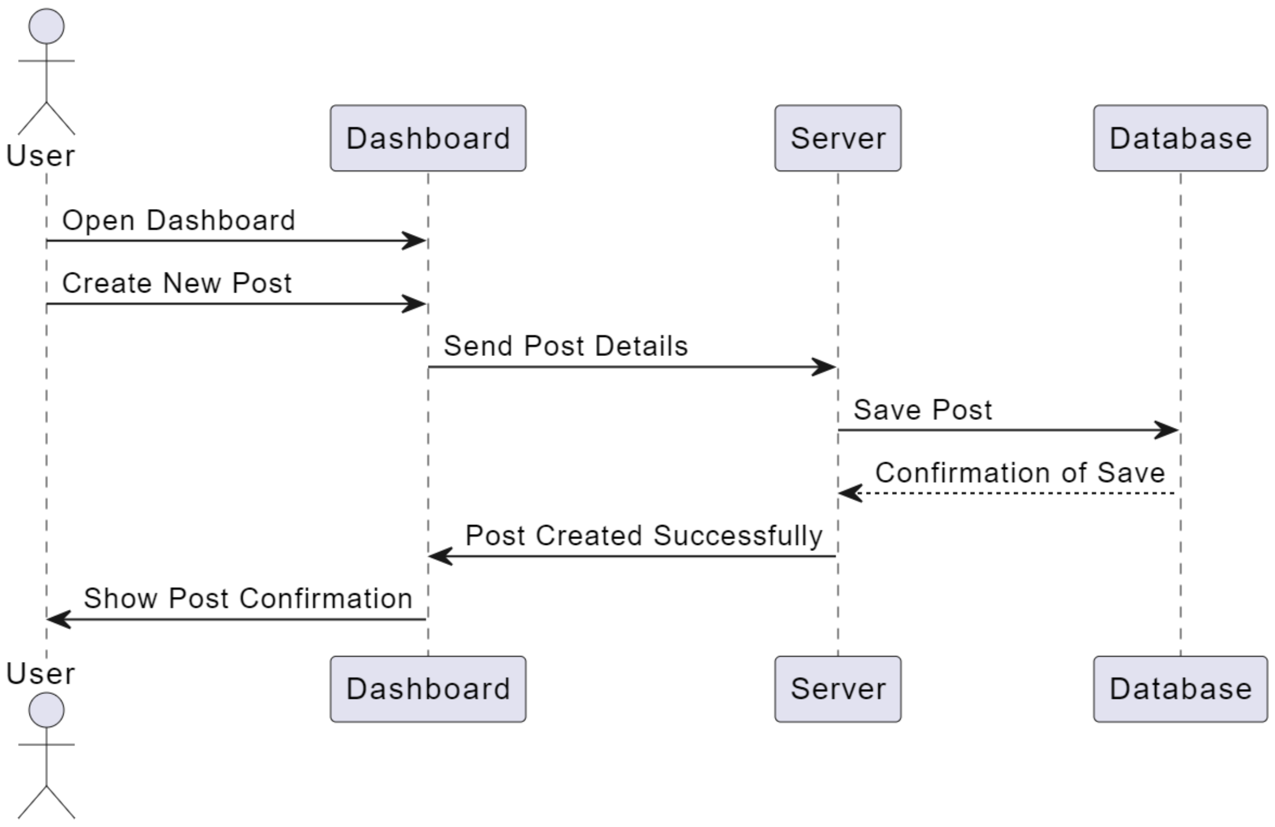
5.4.3.Sequence Diagram For Admin Manage Tournament



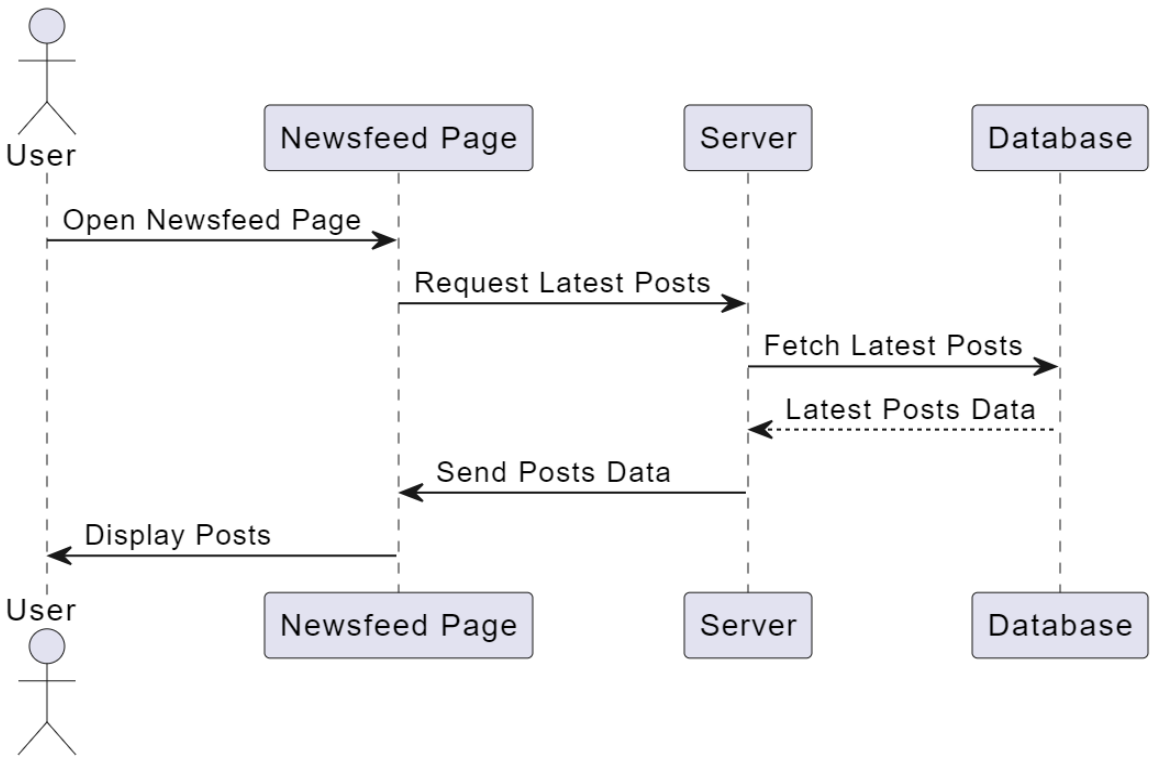
5.4.4.Sequence Diagram For User Participation in a Tournament



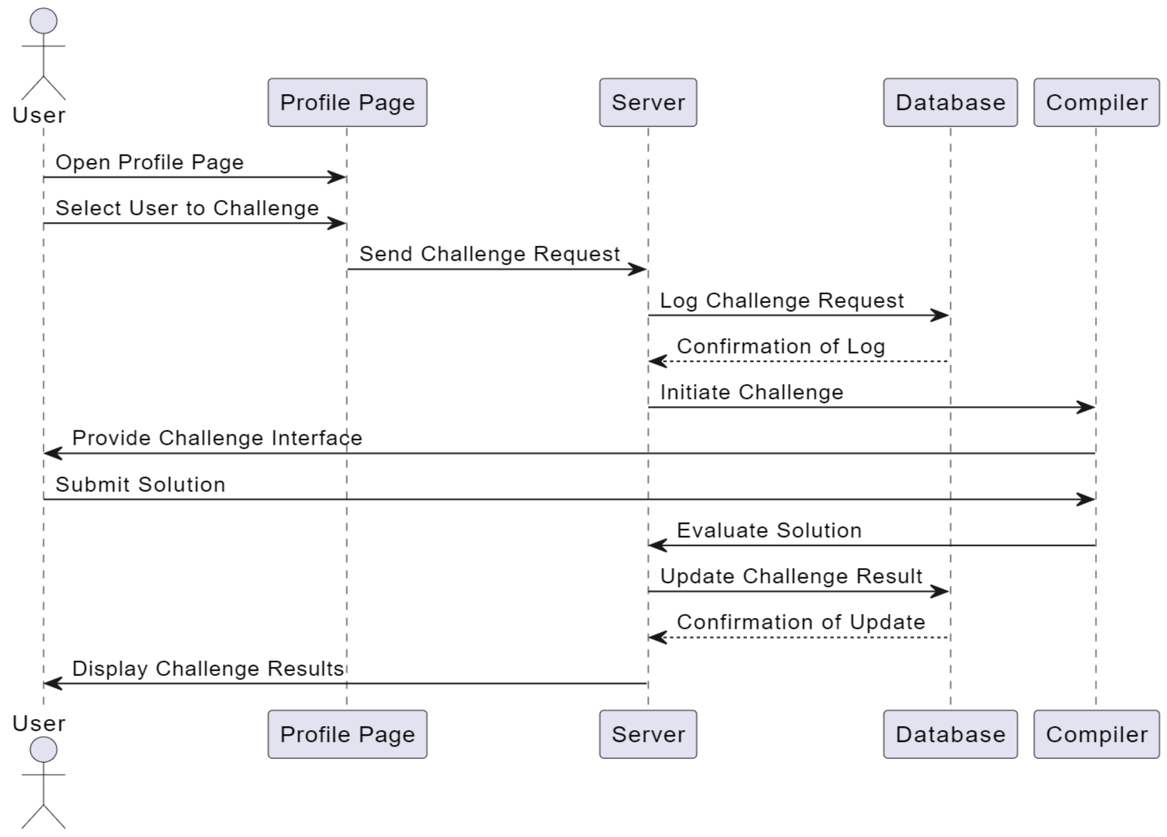
5.4.5.Sequence Diagram For User Posting an Update



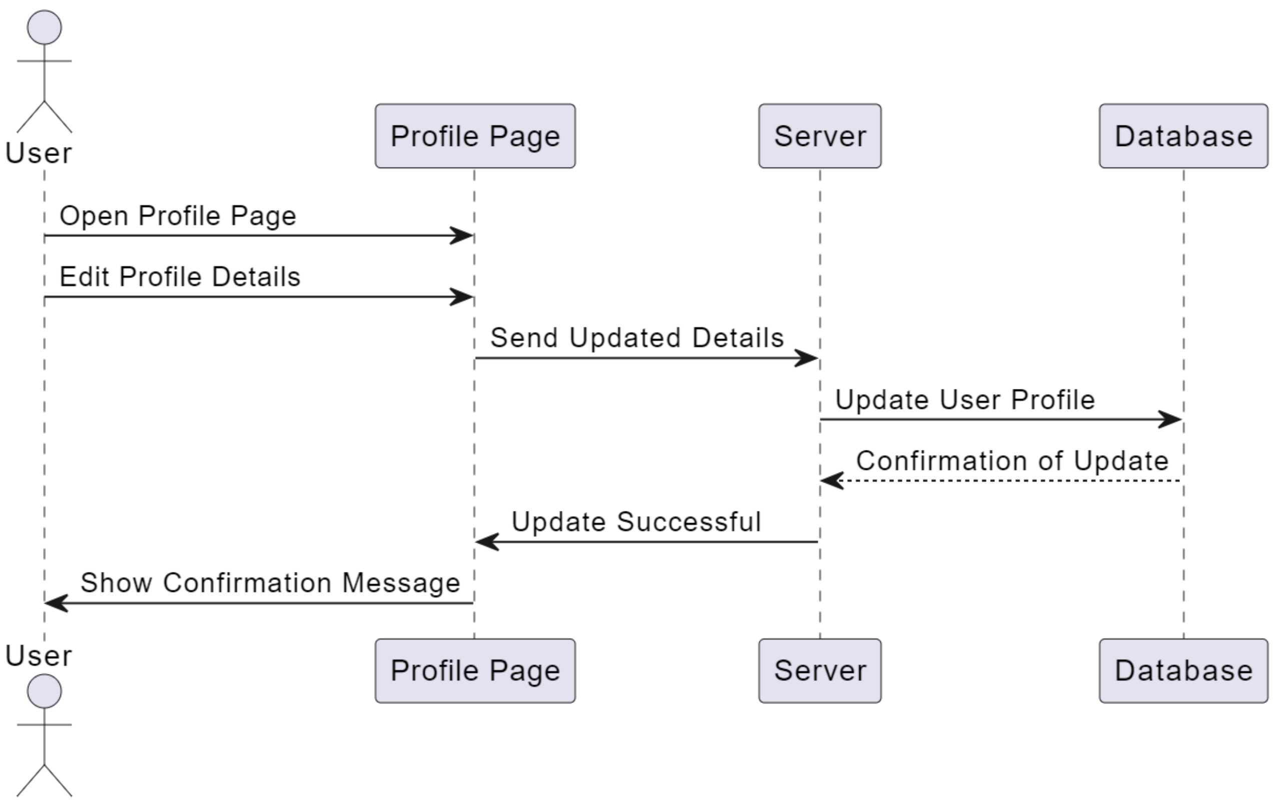
5.4.6.Sequence Diagram For Viewing the Newsfeed



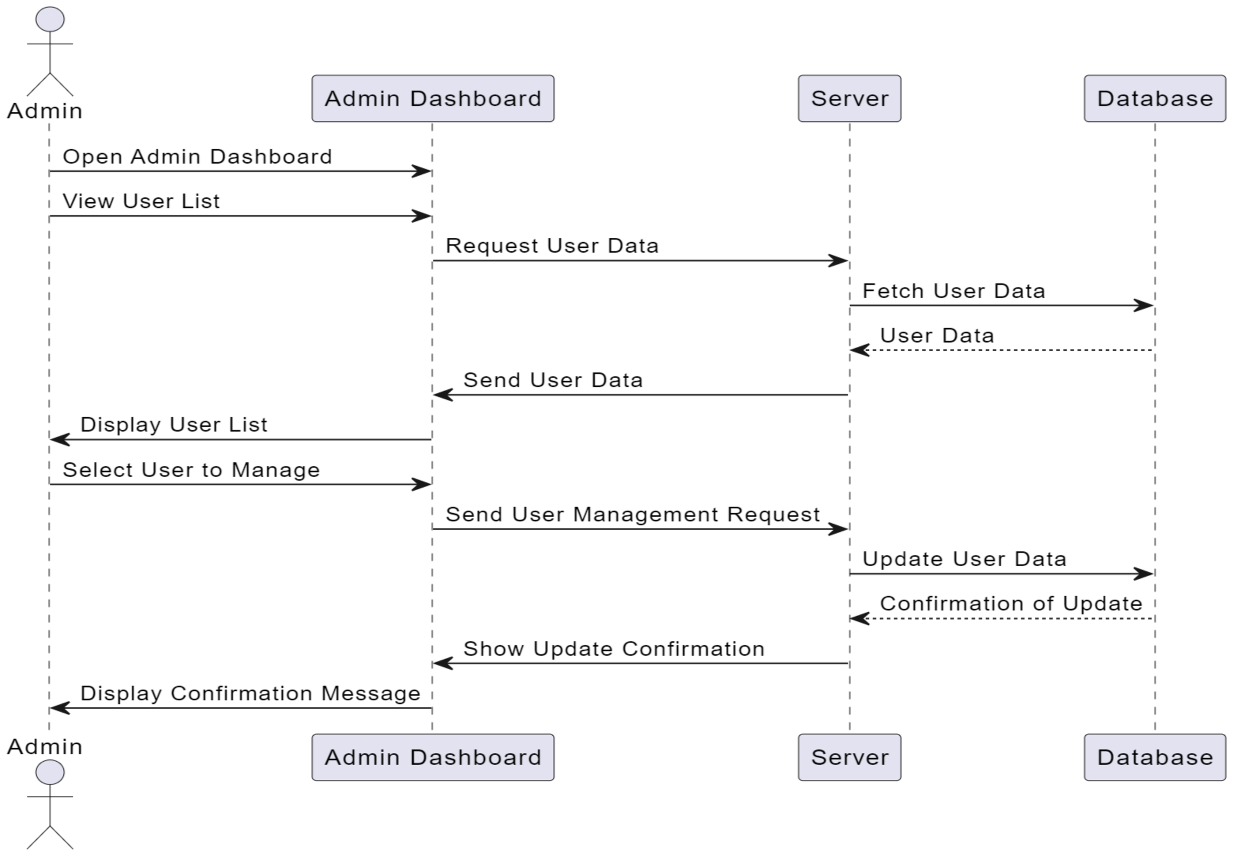
5.4.7. Sequence Diagram For User Initiating a Challenge



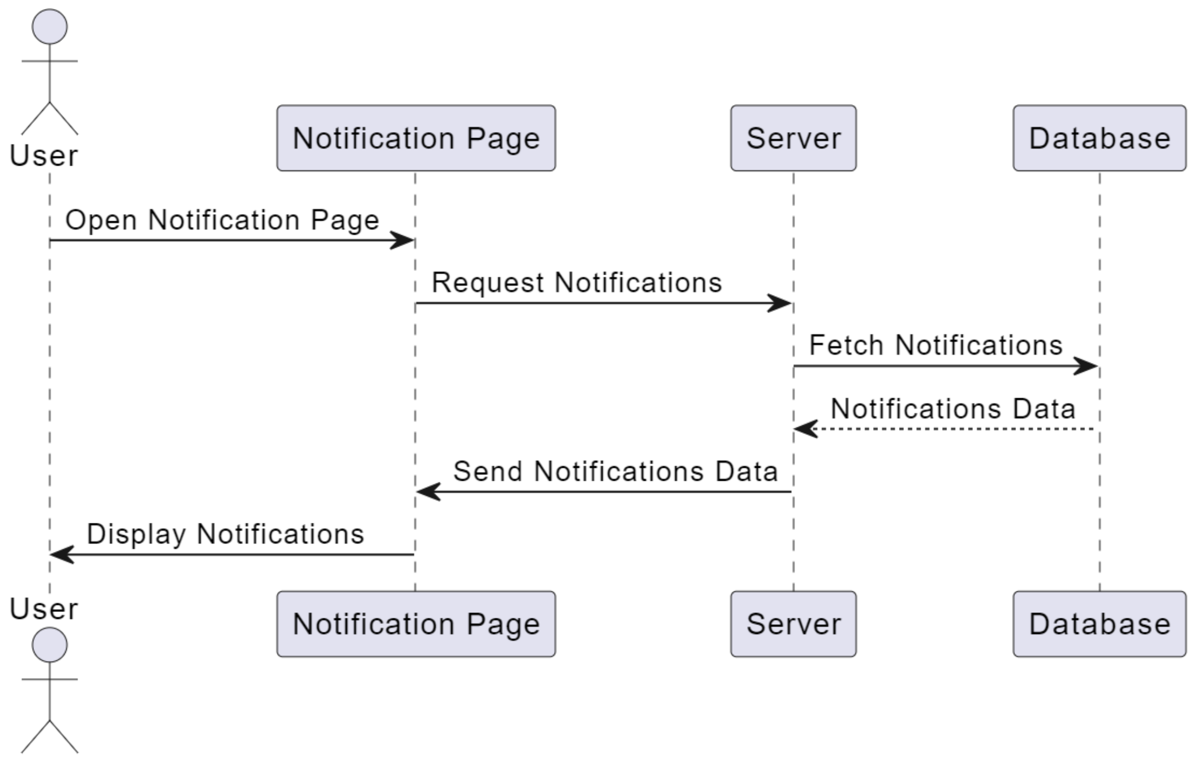
5.4.8. Sequence Diagram For User Managing Their Profile



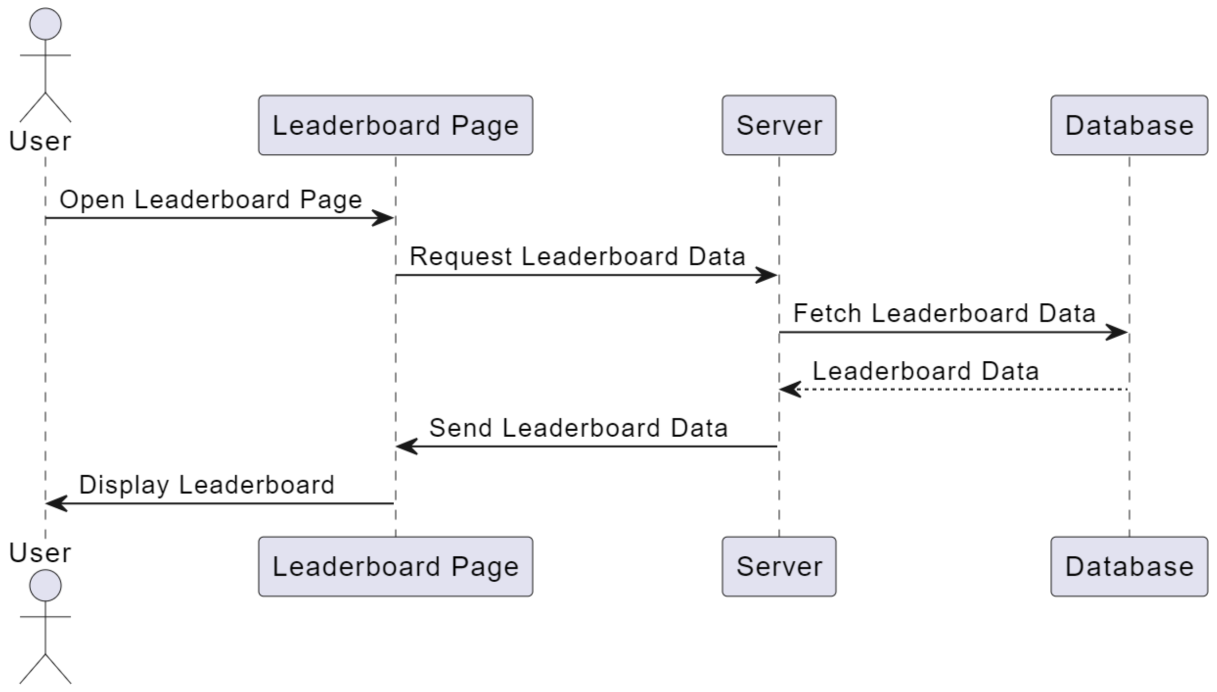
5.4.9. Sequence Diagram For Admin Managing Users



5.4.10. Sequence Diagram For Viewing Notifications



5.4.11. Sequence Diagram For Viewing Leaderboard



5.5.State Dynamics Viewpoint

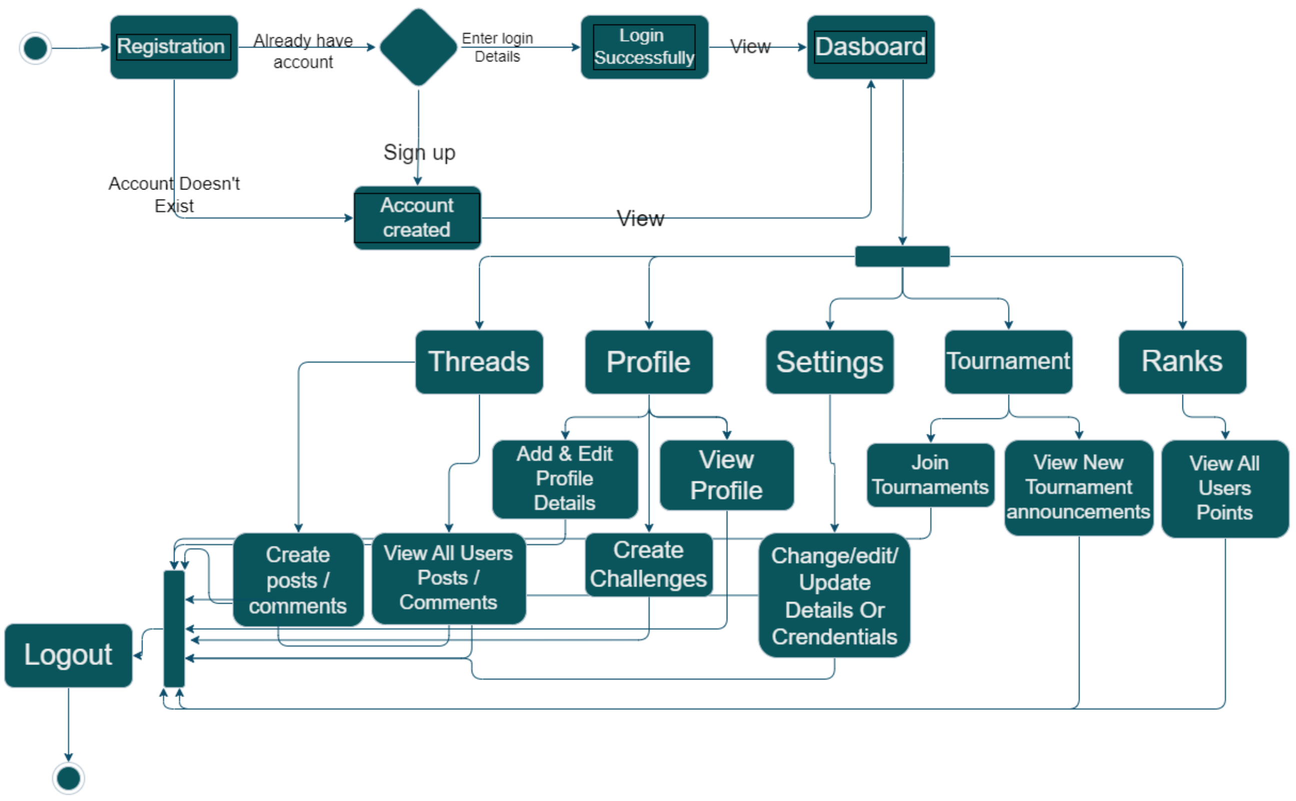


Figure 5.5: State Machine Diagram

5.6.Algorithm Viewpoint

5.6.1. Authentication Module:

```
 function authenticateUser(username, password) {
   if (isValidCredentials(username, password)) {
       return true; // User authenticated successfully
   } else {
       return false; // Invalid credentials
   }
}

function isValidCredentials(username, password) {
   // Fetch user data from database based on username
   userData = getUserData(username);

   // Check if user exists and password matches
   if (userData != null && userData.password == password) {
      return true; // Credentials are valid
   } else {
      return false; // Invalid credentials

  }
}

function getUserData(username) {
   // Query database to fetch user data based on username
   // Return user data if found, otherwise return null
}
```

5.6.2. Posting Module:

```
function createPost(user, content) {
   // Save post to database with user ID and content
   newPost = {
      userId: user.id,
      content: content,
      createdAt: getCurrentDateTime()
   }
   database.savePost(newPost);
}

function editPost(postId, newContent) {
   // Update post content in database
   post = database.getPostById(postId);
   if (post != null) {
      post.content = newContent;
      database.updatePost(post);
   } else {
      print("Post not found!");
   }
}

function deletePost(postId) {
   // Delete post from database
   post = database.getPostById(postId);
   if (post != null) {
      database.deletePost(post);
```

```
    } else {
        print("Post not found!");
    }
}

function getPosts() {
    // Retrieve posts from database
    return database.getAllPosts();
}

// Helper function to get current date and time
function getCurrentDateTime() {
    return now(); // Replace with appropriate date and time function
}
```

5.6.3. Newsfeed Module:

```
function loadNewsfeed() {
    // Retrieve latest posts from database
    posts = fetchLatestPostsFromDatabase()
    return posts
}

function commentOnPost(postId, userId, comment) {
    // Add comment to post in database
    addCommentToPost(postId, userId, comment)
}
```

5.6.4. Challenge Module:

```
function createChallenge(title, description) {
    // Save challenge details to database
    saveChallengeToDatabase(title, description)
}

function joinChallenge(userId, challengeId) {
    // Add user ID to list of participants for the challenge
    addUserIdToChallengeParticipants(userId, challengeId)
}

function getChallenges() {
    // Retrieve list of available challenges from database
    challenges = fetchChallengesFromDatabase()
    return challenges
}
```

5.6.5. Profile Module:

```
function editProfile(userId, newDetails) {
    // Update user profile details in database
    updateUserProfileInDatabase(userId, newDetails)
}

function addFriend(userId, friendId) {
    // Add friend to user's friend list in database
    addFriendToUserInDatabase(userId, friendId)
}

function blockUser(userId, blockedUserId) {
    // Add blocked user to user's block list in database
    addUserToBlockedListInDatabase(userId, blockedUserId)
}
```

5.6.6. Tournament Module:

```
function createTournament(name, description) {
    // Save tournament details to database
    saveTournamentToDatabase(name, description)
}

function registerForTournament(userId, tournamentId) {
    // Add user ID to list of participants for the tournament
    addUserToTournamentParticipants(userId, tournamentId)
}

function getTournaments() {
    // Retrieve list of available tournaments from database
    retrieveTournamentsFromDatabase()
}
```

5.6.7. Settings Module:

```
function updateSettings(userId, newSettings) {
   // Update user settings in database
   saveNewSettingsToDatabase(userId, newSettings)
}

function changePassword(userId, newPassword) {
   // Update user password in database
   saveNewPasswordToDatabase(userId, newPassword)
}

function deactivateAccount(userId) {
   // Set user account status to inactive in database
   setAccountStatusToInactive(userId)
}
```

5.6.8. Leaderboard Module:

```
function getLeaderboard() {
   // Fetch users sorted by the number of challenges won
   leaderboard = fetchLeaderboardFromDatabase()
   return leaderboard
}
```

5.6.9. Admin Module:

```
function reviewComplaint (complaintId, action) {
   // Update the Complaint status and take necessary actions
   updateComplaintStatus(ComlaintId, action)
}
```