## **Use case template**

1.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-001: User Registration | | |
| Primary actor | Unauthenticated user | Secondary actors | Admin |
| Description | Allows a new user to create an account in the application. | | |
| Trigger | User wants to use the application. | | |
| Preconditions | PRE-1: User has internet access  PRE-2: User doesn't already have an account in the system. | | |
| Postconditions | POST-1: A new account is created in the system  POST-2: User is assigned the User role  POST-3: User is authenticated in the system | | |
| Normal flow | 1. User accesses the registration page  2. System displays the registration form  3. User completes personal data (Username, Password)  4. User submits the form  5. System validates the entered data  6. System creates the new account  7. System authenticates the user  8. System redirects the user to the login form | | |
| Alternative flows | 5. Validation fails   1. System displays appropriate error messages 2. User corrects the errors 3. Continue step 4 of normal flow | | |
| Exceptions | Username is already in use  System notifies the user  User make another username | | |

2.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-002: User Authentication | | |
| Primary actor | Unauthenticated user | Secondary actors | - |
| Description | Allows an existing user to authenticate in the application | | |
| Trigger | User wants to access the application's functionalities | | |
| Preconditions | PRE-1: User has an account in the system | | |
| Postconditions | POST-1: User is authenticated in the system  POST-2: System generates a JWT token for the session | | |
| Normal flow | 1. User accesses the authentication page  2. System displays the authentication form  3. User enters username and password  4. User submits the form  5. System validates the credentials  6. System generates a JWT token  7. System redirects the user to the main page | | |
| Alternative flows | 5. Validation fails   1. Credentials are incorrect 2. System displays an error message 3. User re-enters credentials 4. Continue with step 4 of the normal flow | | |
| Exceptions | User account not exist  System notify the user about the access issue | | |
|  |  | | |

3.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-003: Update Profile | | |
| Primary actor | Authenticated user | Secondary actors | - |
| Description | Allows an existing user to update profile image and details about him | | |
| Trigger | User wants to change information about him | | |
| Preconditions | PRE-1: User has an account in the system | | |
| Postconditions | POST-1: User have new details about him | | |
| Normal flow | 1. User accesses the profile page  2. System displays the informations form  3. User enters new image or last name or first name (all of it or combinations)  4. User submits the form  5. System validates the informations  6. System change old informations  7. System show new profile | | |
| Alternative flows | 5. Validation fails   1. Informations are incorrect 2. System displays an error message 3. User re-enters informations 4. Continue with step 4 of the normal flow | | |
| Exceptions | User account not exist  Profile image should be an link from the internet  System notify the user about the access issue | | |

4.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-003: Logout | | |
| Primary actor | Authenticated User | Secondary actors | - |
| Description | Allows the user to disconnect from his account | | |
| Trigger | User wants to leave our platform | | |
| Preconditions | PRE-1: User is authenticated | | |
| Postconditions | POST-1: User should login again | | |
| Normal flow | 1. User accesses the "Logout" button / Navbar section  2. User selects the "Logout" option  3. System redirect to welcome page | | |
| Alternative flows | 8. Validation fails - | | |
| Exceptions | - | | |

5.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-004: Team Management -> Add Members to team | | |
| Primary actor | Team Owner | Secondary actors | User |
| Description | Allows the team owner to add other users as members | | |
| Trigger | Owner wants to expand the team | | |
| Preconditions | PRE-1: User is authenticated | | |
| Postconditions | POST-1: Selected users are added as team members with "Pending" status | | |
| Normal flow | 1. Owner accesses the team page  2. Owner selects "Add members" option  3. System displays the user search interface  4. Owner searches for users by username  5. System displays search results  6. Owner selects users to add  7. Owner confirms addition  8. System creates Participant relationships between users and team  9. System displays confirmation of addition | | |
| Alternative flows | 8. Searching fails   1. Owner enters the list of usernames separated by commas 2. User corrects the data 3. Continue with step 4 of the normal flow | | |
| Exceptions | 1. User is already a team member 2. System notifies the owner | | |

5.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-005: Create New Competition | | |
| Primary actor | Authenticated user or Admin | Secondary actors | User |
| Description | Allows the creation of a new sports competition | | |
| Trigger | User/Admin wants to organize a competition | | |
| Preconditions | PRE-1: User is authenticated | | |
| Postconditions | POST-1: A new competition is created in the system POST-2: User becomes the creator of the competition | | |
| Normal flow | 1. User accesses the "Competitions" section 2. User selects "Create new competition" 3. System displays the creation form 4. User enters the competition name 5. User adds a description 6. User sets the start and end dates 7. User specifies the maximum number of teams 8. User sets the initial status (e.g., "Open for registration") 9. User confirms creation 10. System validates the data 11. System creates the competition in the database 12. System redirects to the feed (views page) page | | |
| Alternative flows | Validation fails   1. System displays errors 2. User corrects the data 3. Continue with step 9 of the normal flow | | |
| Exceptions | Start date is in the past  System notifies the user about the date error | | |

6.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-006: Register Team in Competition | | |
| Primary actor | Team Owner | Secondary actors | - |
| Description | Allows a team owner to register their team in a competition | | |
| Trigger | Owner wants to participate with their team in a competition | | |
| Preconditions | PRE-1: User is authenticated PRE-2: Competition has "Open for registration" status PRE-3: Current number of teams < maximum number | | |
| Postconditions | POST-1: Team is registered in the competition POST-2: Corresponding relationships are created in the database | | |
| Normal flow | 1. Owner accesses the competition page  2. Owner selects "Register team"  3. System displays the list of teams owned by the user  4. Owner selects the team to register  5. Owner confirms registration  6. System checks eligibility  7. System creates relationships between team and competition  8. System confirms registration | | |
| Alternative flows | Validation fails  1. Team is not eligible  2. Continue with step 6 | | |
| Exceptions | Competition has reached the maximum number of teams | | |

7.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-007: Manage Matches in Competition | | |
| Primary actor | Owner or Admin | Secondary actors | - |
| Description | Allows organizing matches between teams registered in the competition | | |
| Trigger | Creator wants to establish the match schedule | | |
| Preconditions | PRE-1: User is authenticated PRE-2: User is the competition creator or Admin PRE-3: Competition has at least two registered teams | | |
| Postconditions | POST-1: Matches are scheduled and recorded in the system | | |
| Normal flow | 1. Creator accesses the competition page  2. Creator selects "Manage matches"  3. System displays the views interface  4. Creator selects two teams for each match  5. Creator sets the date, time, and location for each match  6. Creator adds a description for each match (optional)  7. Creator confirms scheduling  8. System validates the data  9. System creates Match entities in the database  10. System confirms scheduling | | |
| Alternative flows | 1. Creator selects "Generate matches automatically"  2. System automatically generates matches | | |
| Exceptions | Match date is before the competition start date  System notifies the creator about the date conflict  Creator corrects the date | | |

8.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-009: Create Post | | |
| Primary actor | Authenticated user | Secondary actors | - |
| Description | Allows the user to add a personalized nutritional recipe | | |
| Trigger | User wants to share information | | |
| Preconditions | PRE-1: User is authenticated | | |
| Postconditions | POST-1: Post is created and displayed in the feed | | |
| Normal flow | 1. User accesses the "My Posts" section.  2. User selects "Add new recipe"  3. System displays the recipe form  4. User enters the recipe name  5. User adds a recipe description  6. User adds the ingredients list  7. User adds preparation instructions  9. User confirms recipe creation  10 System validates the data | | |
| Alternative flows | Validations fails  1. System displays errors  2. User corrects the data | | |
| Exceptions | Ingredients list is empty | | |

9.

|  |  |  |  |
| --- | --- | --- | --- |
| ID and name | UC-011:Admin | | |
| Primary actor | Admin | Secondary actors | Users |
| Description | Allows the administrator to manage user accounts | | |
| Trigger | Admin wants to modify user accounts | | |
| Preconditions | PRE-1: User is authenticated PRE-2: User has the Admin role | | |
| Postconditions | POST-1: Changes to accounts are applied | | |
| Normal flow | 1. Admin accesses the Admin section  2. System displays the user/competition/team list  3. Admin can filter or search for users / competition / teams  4. Admin selects a user/competition/teams for editing  5. System displays (pack of 3 – user/comp/teams) details  6. Admin can modify roles  7. Admin can deactivate/reactivate the account  8. Admin confirms changes  9. Admin can delete comps, teams  10. System applies the changes  11. System confirms the modifications | | |
| Alternative flows | Decisions between modify account and delete | | |
| Exceptions | Admins tries to delete their own account | | |