Use Cases

**Use Case UC1: Register user**

**Primary Actor:** Organizer or runner

**Stakeholders and Interests:**

* + Organizer: Wants to be registered as a user with the proper privileges
  + Runner: Wants accurate, fast entry, and no errors.

**Preconditions:** Organizers or runner is logged on

**Post conditions:** Organizers or Runners are now registered users.

**Summary:** Guest enters website and registers to be either organizer or Runner

**Basic Flow:**

* 1. Organizer/Runner goes to webpage to register a Organizer/Runner account
  2. Organizer/Runner enters in their Name, password, email address, what level or privilege they want(Runner or organizer)
  3. The System then takes this information and creates a user on the database.
  4. The website return the Organizer/Runner a message saying they are register as a Runner or Organizer

**Alternate Flows:**

2a. System detects an invalid or no input for one of the fields:

1. System signals errors and rejects entry.
2. Sends a message to user telling them which input field is invalid and why.

**Use Case UC2: Edit user**

**Primary Actor:** System Admin/Organizer/Runner

**Stakeholders and Interests:**

* + System Admin/Organizer/Runner: Wants accurately edit user account with no errors.

**Preconditions:** System Admin, Organizers or Runner is logged on and is a registered user.

**Post conditions:** System Admin, Organizers or Runners have now edited a/their user account.

**Summary:** System Admin/Organizer/Runner enters website and edits user account information. Organizers and Runners can only edit their own account but System admin can edit any account.

**Basic Flow:**

1. System Admin/Organizer/Runner goes to webpage to edit an Organizer/Runner account
2. System Admin/Organizer/Runner enters in the account fields they want to edit and submits.
3. The System then takes this information and edits the user account on the database.
4. The website return the Organizer/Runner a message saying they have successfully updated the user account information

**Alternate Flows:**

2a. System detects an invalid or no input for one of the fields:

1. System signals errors and rejects entry.
2. Sends a message to user telling them which input field is invalid and why.

2b. System detects they are trying to edit account they do not have privilege to edit

* + 1. System signals errors and rejects entry.
    2. Sends a message to the user telling them they do not have the privilege to edit this user account.

**Use Case UC3: Delete user**

**Primary Actor:** System Admin/Organizer/Runner

**Stakeholders and Interests:**

* + System Admin/Organizer/Runner: Wants accurately delete the correct user account with no errors.

**Preconditions:** System Admin, Organizers or Runner is logged on and is a registered user.

**Post conditions:** System Admin, Organizers or Runners have now deleted the user they wanted to.

**Summary:** System Admin/Organizer/Runner enters website and deletes a user account. Organizers and Runners can only delete their own account but System admin can delete any account.

**Basic Flow:**

1. System Admin/Organizer/Runner goes to webpage to delete an Organizer/Runner account
2. System Admin/Organizer/Runner clicks on the account they want to delete
3. The System then takes this information and deletes the account specified.
4. The website returns the Organizer/Runner a message saying they have successfully deleted the user account.

**Alternate Flows:**

2a. System detects they are trying to delete an account they do not have privilege to delete

1. System signals errors and rejects entry.
2. Sends a message to the user telling them they do not have the privilege to delete this user account.

**Use Case UC4: Submit Race Results**

**Primary Actor:** Organizer

**Stakeholders and Interests:**

* + Organizer: Wants accurate, fast entry, and no errors.

**Preconditions:** Organizer is identified and authenticated.

**Post conditions:** Race results are recorded. Database updated.

**Summary:** Organizer enters in data of the race and this gets stored in the database.

**Basic Flow:**

1. Organizer goes to webpage to create a race to add results to.
2. Organizer enters in the results of each participant as well as age, sex, etc.
3. Organizer submits whole race results and the database adds it to its races
4. Database sends message back to user informing them that everything was added successfully

**Alternate Flows:**

2a. System detects an invalid input for participant:

1. System signals errors and rejects entry.
2. Sends a message to user telling them which input is invalid and why.
3. User then tries to resubmit entry.

3a. System detects an invalid input for race:

1. System signals errors and rejects entry.
2. Sends a message to user telling them which input is invalid and why.
3. User then tries to resubmit entry.

**Use Case UC5: Modify Race Results**

**Primary Actor:** Organizer

**Stakeholders and Interests:**

* + Organizer: Wants accurate, fast modification, and no errors.

**Preconditions:** Organizer is identified and authenticated. There is a race in the database that can be modified.

**Post conditions:** Race results are modified. Database updated.

**Summary:** Organizer modifies the data of an already added race and this gets stored in the database.

**Basic Flow:**

1. Organizer goes to webpage were the race they want to modify is.
2. Organizer clicks on the modify button of the race and is taken to a page were the race information can be modified.
3. Organizer modifies the fields they need to and then submit the newly modified race to the database.
4. Database sends message back to user informing them that everything was added successfully

**Alternate Flows:**

3a. System detects an invalid input for race:

1. System signals errors and rejects entry.
2. Sends a message to user telling them which input is invalid and why.
3. User then tries to resubmit entry.

**Use Case UC6: Search Race Results and runners**

**Primary Actor:** Runner

**Stakeholders and Interests:**

* + Runner: Wants to view accurate race results in a viewable manner

**Preconditions:** Runner is identified and authenticated. There is a race or runner in the database that can be viewed.

**Post conditions:** Runner is able to search for the race results and runners.

**Summary:** Runner wants to find specific race information, like finding race by name and runners in the race by name.

**Basic Flow:**

1. Organizer goes to webpage were the race they want to search.
2. Organizer enters the information into proper field they want to search by. Race name, runner name, etc.
3. Runner submits the key information he wants to search for with the submit button.
4. Database finds the race it need to and then returns the website page for that race with the races information on it.

**Alternate Flows:**

4a. System detects the keyword searched by is not in database:

1. System signals errors and rejects entry.
2. Sends a message to user telling them that keyword is not in the database and to try a different one.

4b. System detects invalid input in the field

1. System signals errors and rejects entry.
2. Sends message to user telling them that the input is invalid and they should change it.

7. When viewing races Runner wants to filter the results by information of runners by age, sex, etc. They toggle a button on the webpage. The toggle then sends a query to the database to only show what is filtered. The database only shows the response of the query filtering the results Runner wants.

8. Runner can also sort the results by age and sex of runners in the race. They change the sort options on the race page view and then submit this. The webpage sends a sort query to the database and the database returns the sorted data accordingly. Both 7 and 8 are altering the race view given by finding a race through the search (use case 6). The view race use case can be seen as a generalized use case from 7 and 8.

