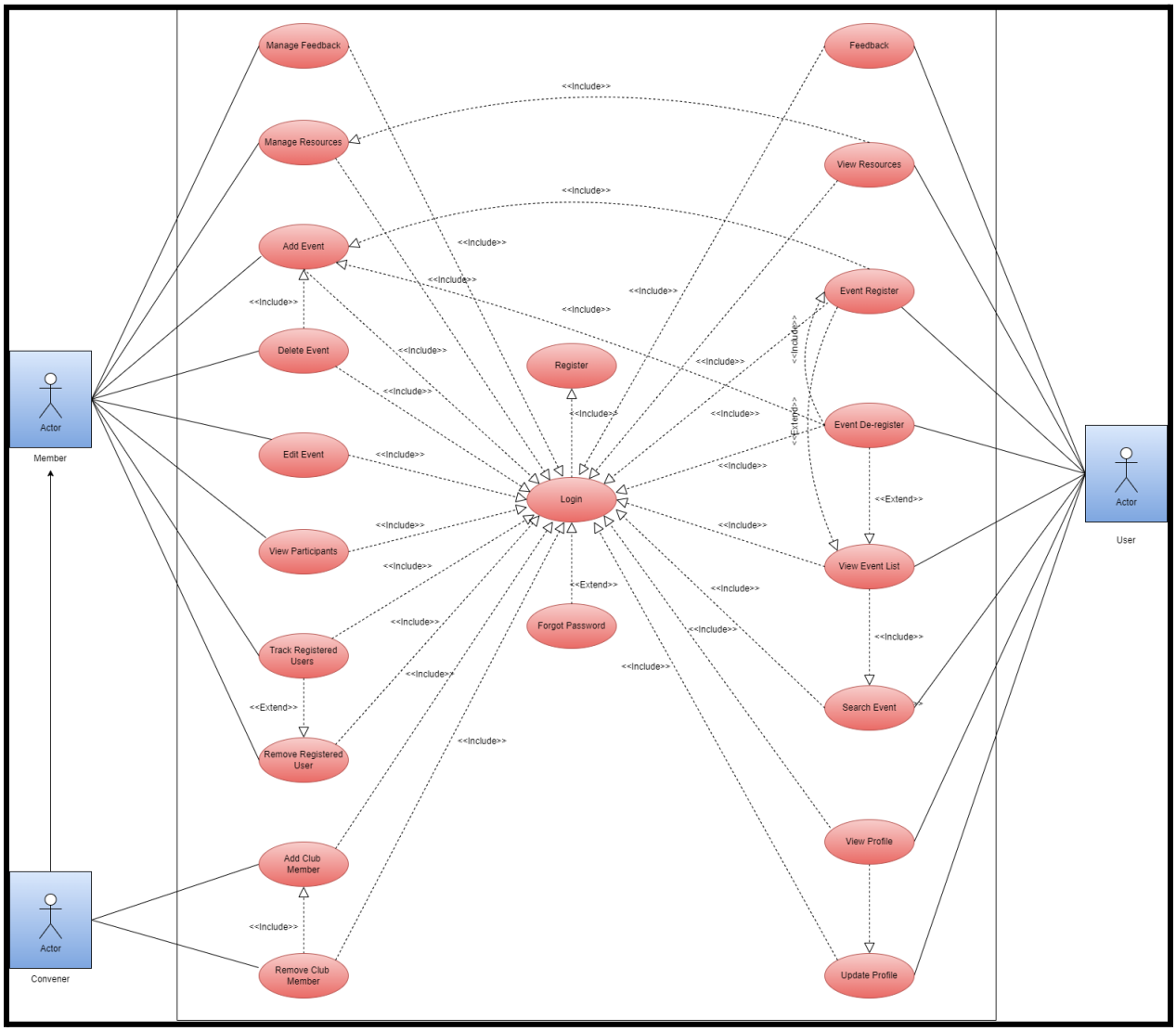
IT-314 Software Engineering

Group: 27 College Programming Club

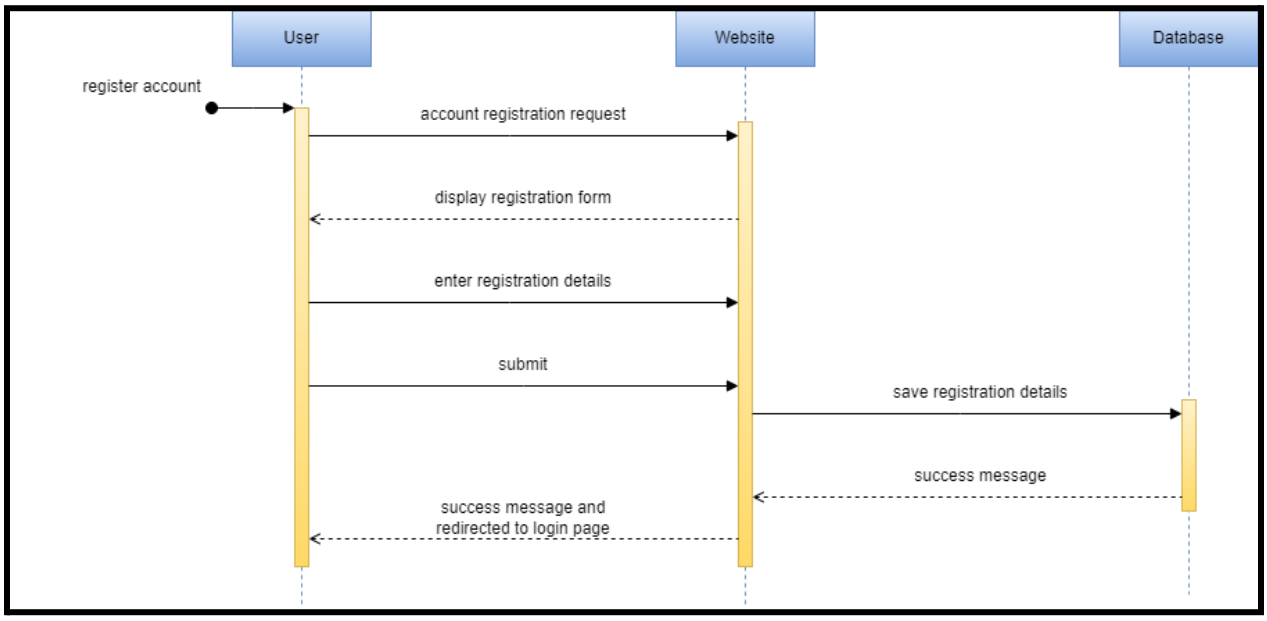
IDs	Name
202001241	RIKI ANILBHAI PARIKH
202001256	TUSHAR DHAR
202001265	**SANGANI KISHAN PARESHKUMAR**
202001267	BHATT MEET DHAVALBHAI
202001268	MANSARA FENIL LALITBHAI
202001272	ARCHIT SINGH
202001276	OM CHALODIYA
202001426	VADODIA HARSHIT JAGDISHBHAI
202001433	MODIYA DEEP ASHOKBHAI
202001445	PATEL KRUNAL MAHESHBHAI

Use Case Diagram

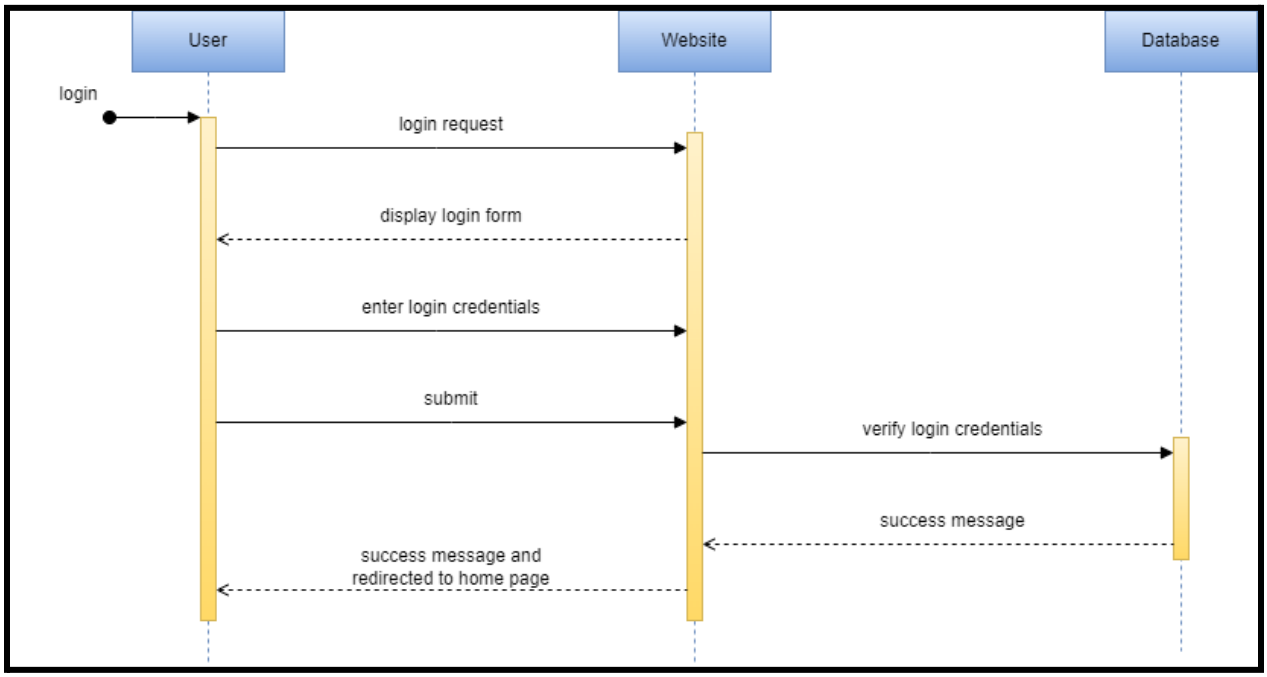


Sequence Diagram

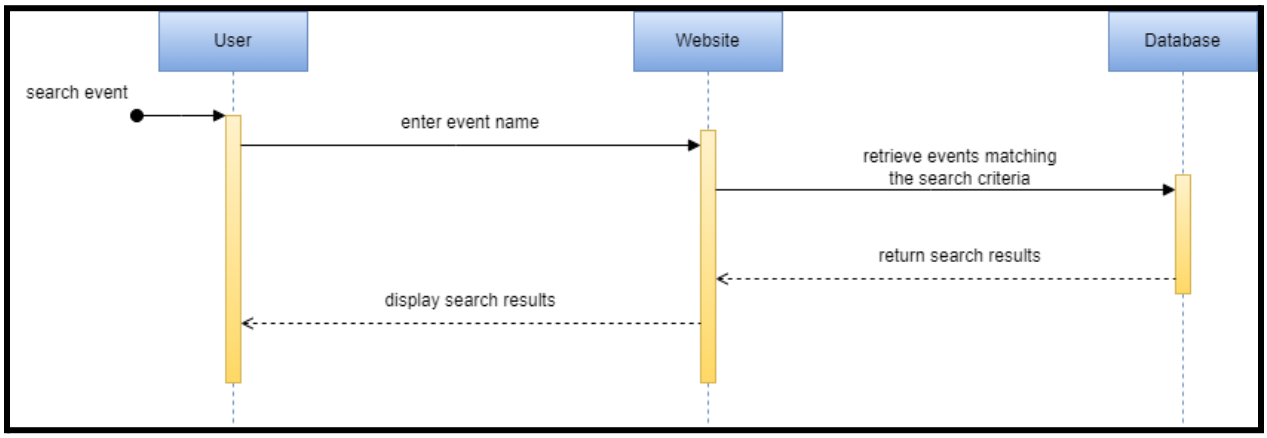
Registration



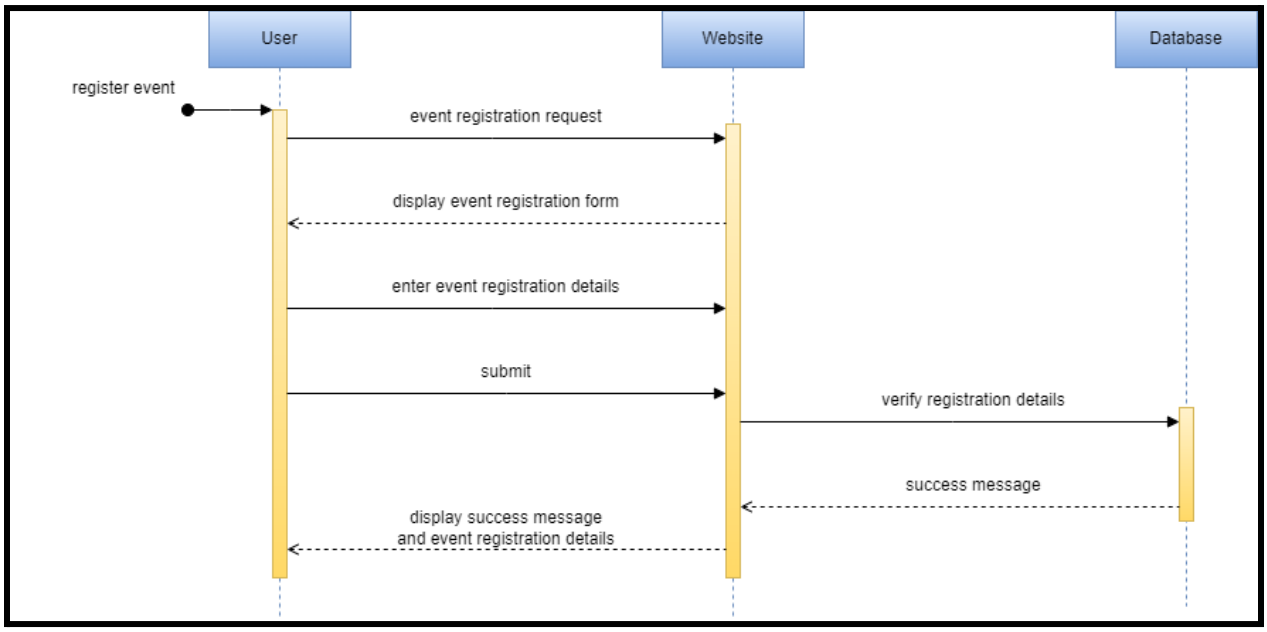
Login



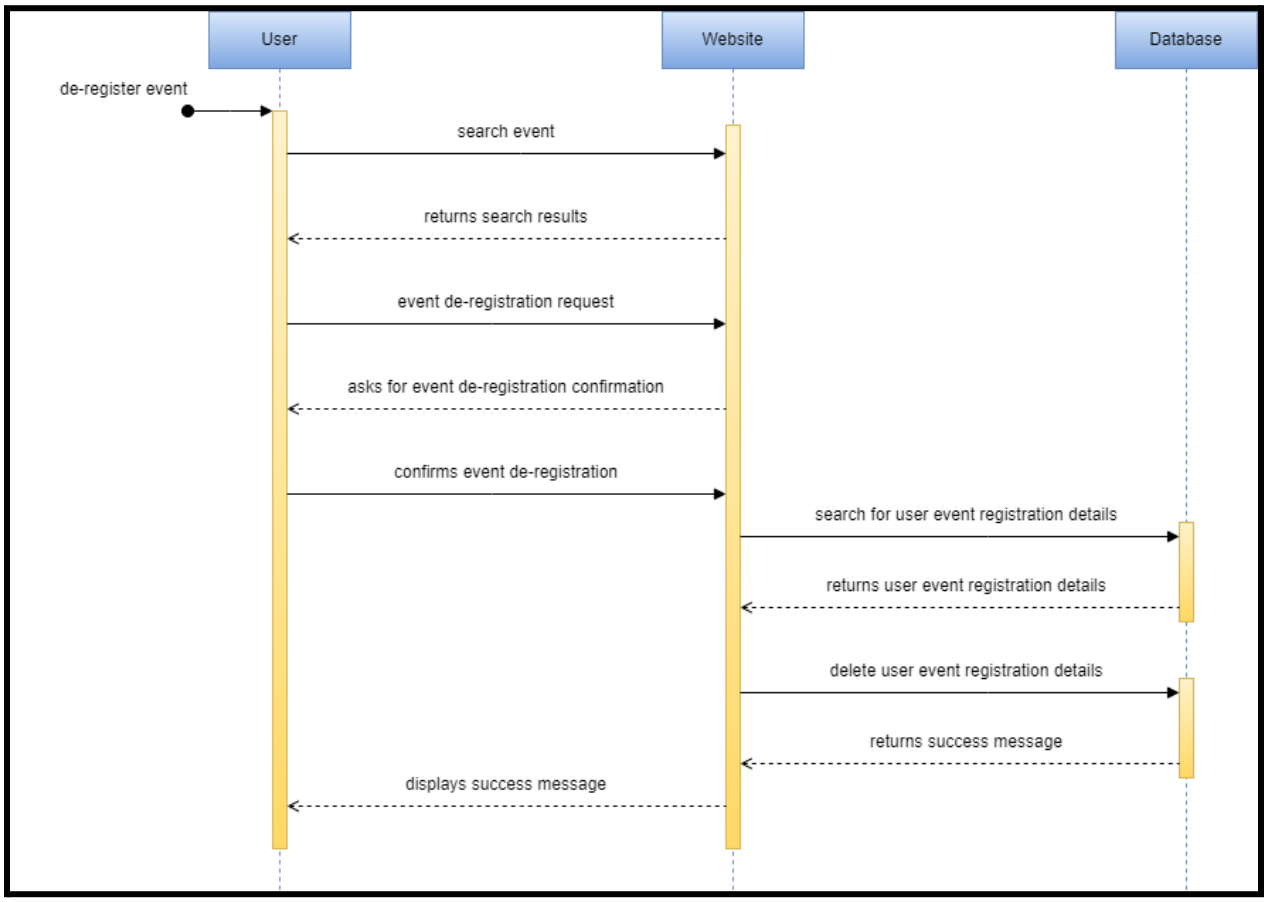
Event Search



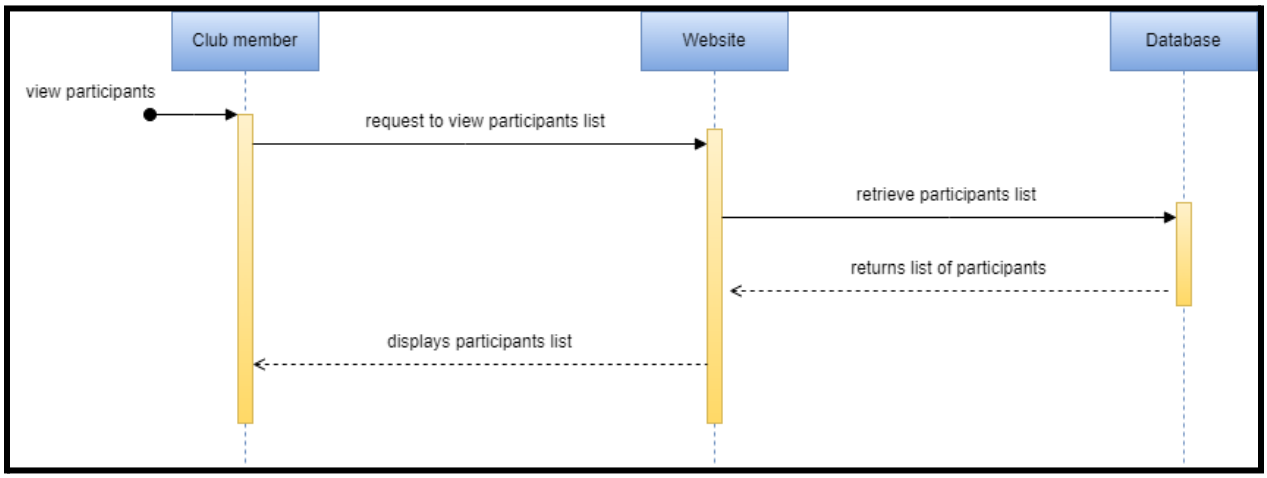
Event Registration



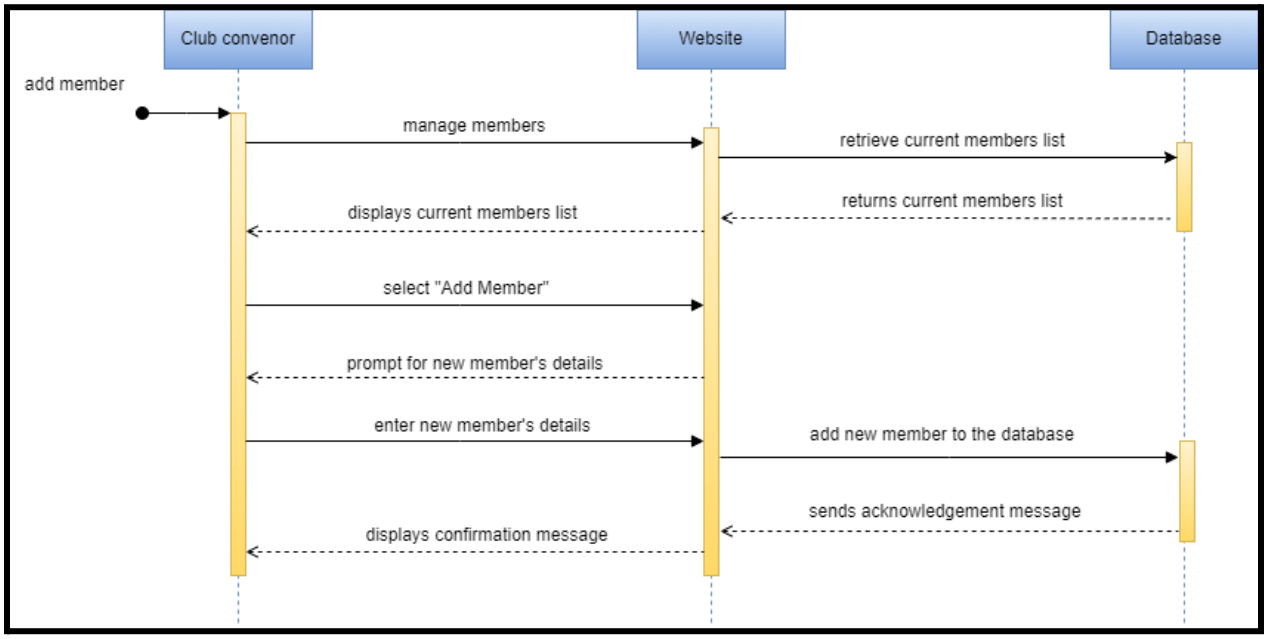
Event De-registration



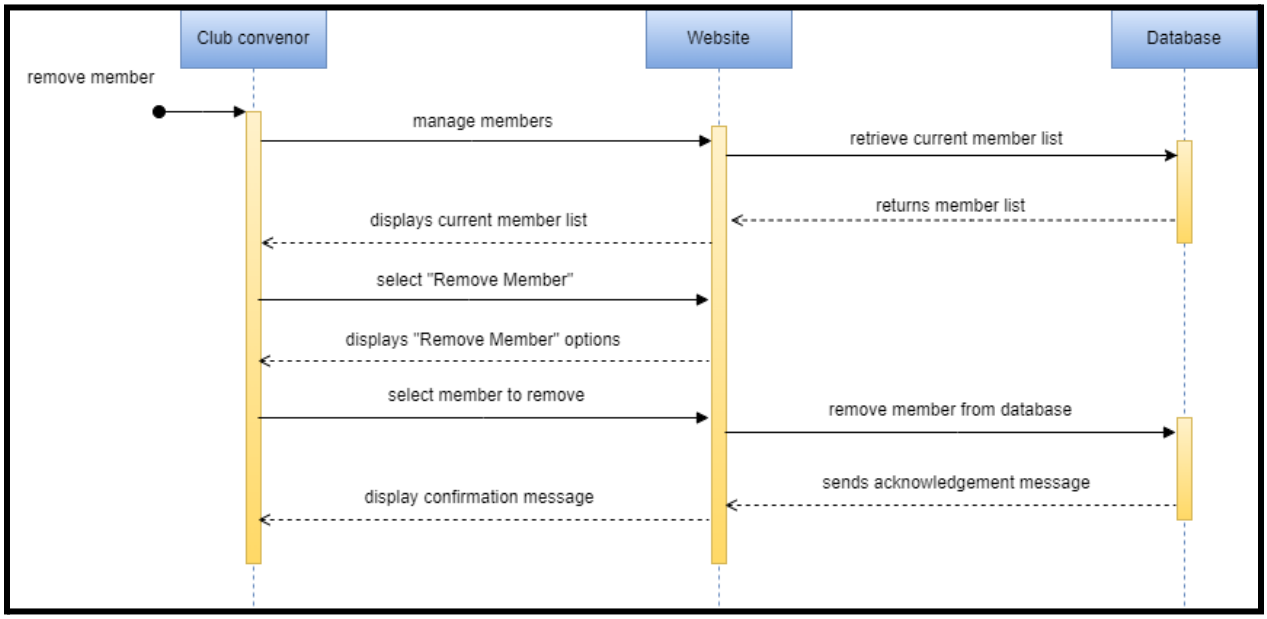
View Participants



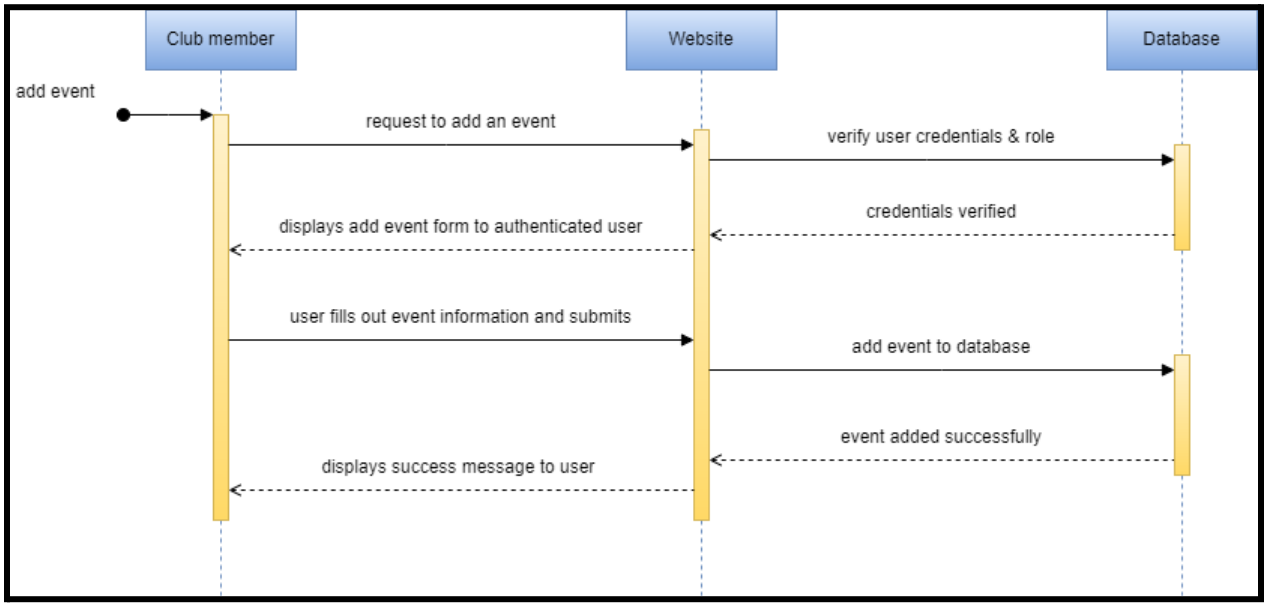
Add Member



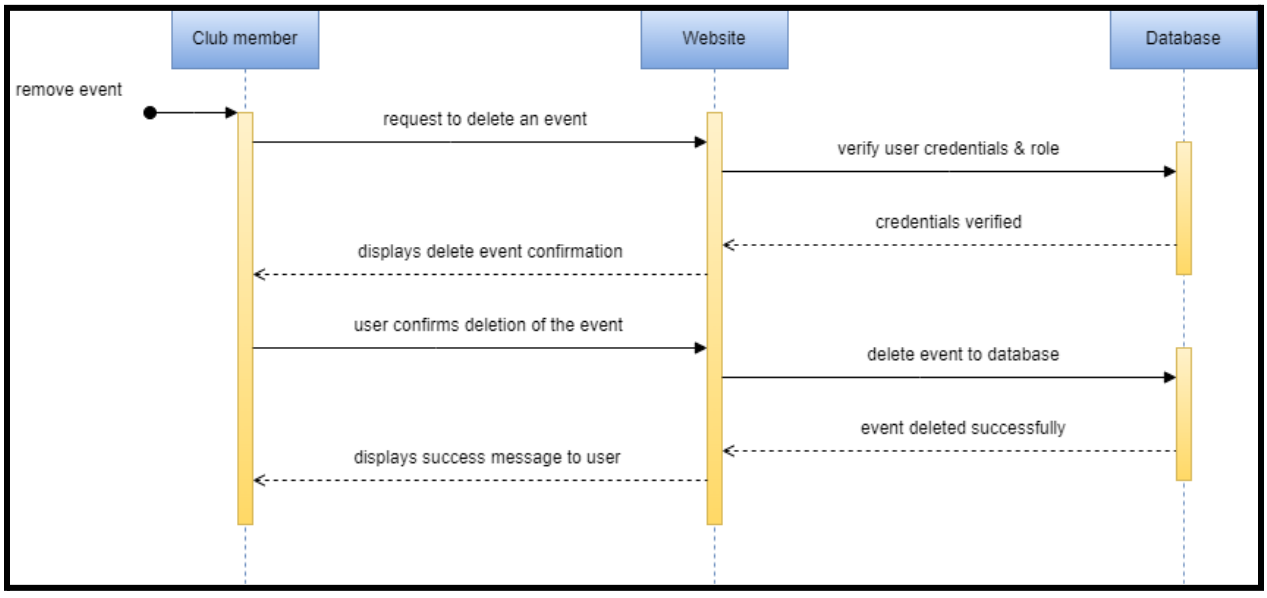
Remove Member



Add Event



Remove Event



Classes - Attributes and Operations

1) Role

- **Attributes:**
  - role_id: a unique identifier for the user role
  - role_title: the title of the role
  - role_description: the description of the role

- **Operations:**
  - createRole(): allows an authorized user to create a new role
  - editRole(): allows an authorized user to edit an existing role
  - deleteRole(): allows an authorized user to delete an existing role
  - assignRole(): allows an authorized user to assign a role to another user

2) User

- **Attributes:**
  - user_id: a unique identifier for the user
  - user_name: the user's name
  - user_role: the role of the user (e.g. student, club member, faculty, admin)
  - user_email: the email address of the user
  - user_password: the password of the user

- **Operations:**
  - createUser(): allows an authorized user to create a new user account
  - editUser(): allows an authorized user to edit an existing user account
  - deleteUser(): allows an authorized user to delete an existing user account
  - viewProfile(): allows the user to view their own profile information
  - editProfile(): allows the user to edit their profile information
  - viewEvents(): allows the user to view upcoming events
  - registerEvent(): allows the user to register for an upcoming event
  - de-registerEvent(): allows the user to de-register from an upcoming event
  - viewAttendance(): allows the user to view their attendance records
  - viewStandingsList(): allows the user to view the standings list of an event

3) Event

- **Attributes:**
    - event_id: a unique identifier for the event
    - event_name: the name of the event
    - event_description: a description of the event
    - event_date: the date of the event
    - event_time: the time of the event
    - event_location: the location of the event

- **Operations:**
    - createEvent(): allows an authorized user to create a new event
    - searchEvent(): allows the user to search for an event
    - editEvent(): allows an authorized user to edit an existing event
    - deleteEvent(): allows an authorized user to delete an existing event
    - registerEvent(): allows the user to register for an upcoming event
    - de-registerEvent(): allows the user to de-register from an upcoming event
    - viewAttendance(): allows an authorized user to view attendance records for an event
    - viewStandingsList(): allows the user to view the standings list for a particular event

4) Resources

- **Attributes:**
    - resource_id: a unique identifier for the resource
    - resource_name: the name of the resource
    - resource_description: a description of the resource
    - resource_url: the URL where the resource can be accessed
    - resource_type: the type of the resource (e.g. article, video, book)
    - resource_tags: a list of tags associated with the resource (e.g. topic)

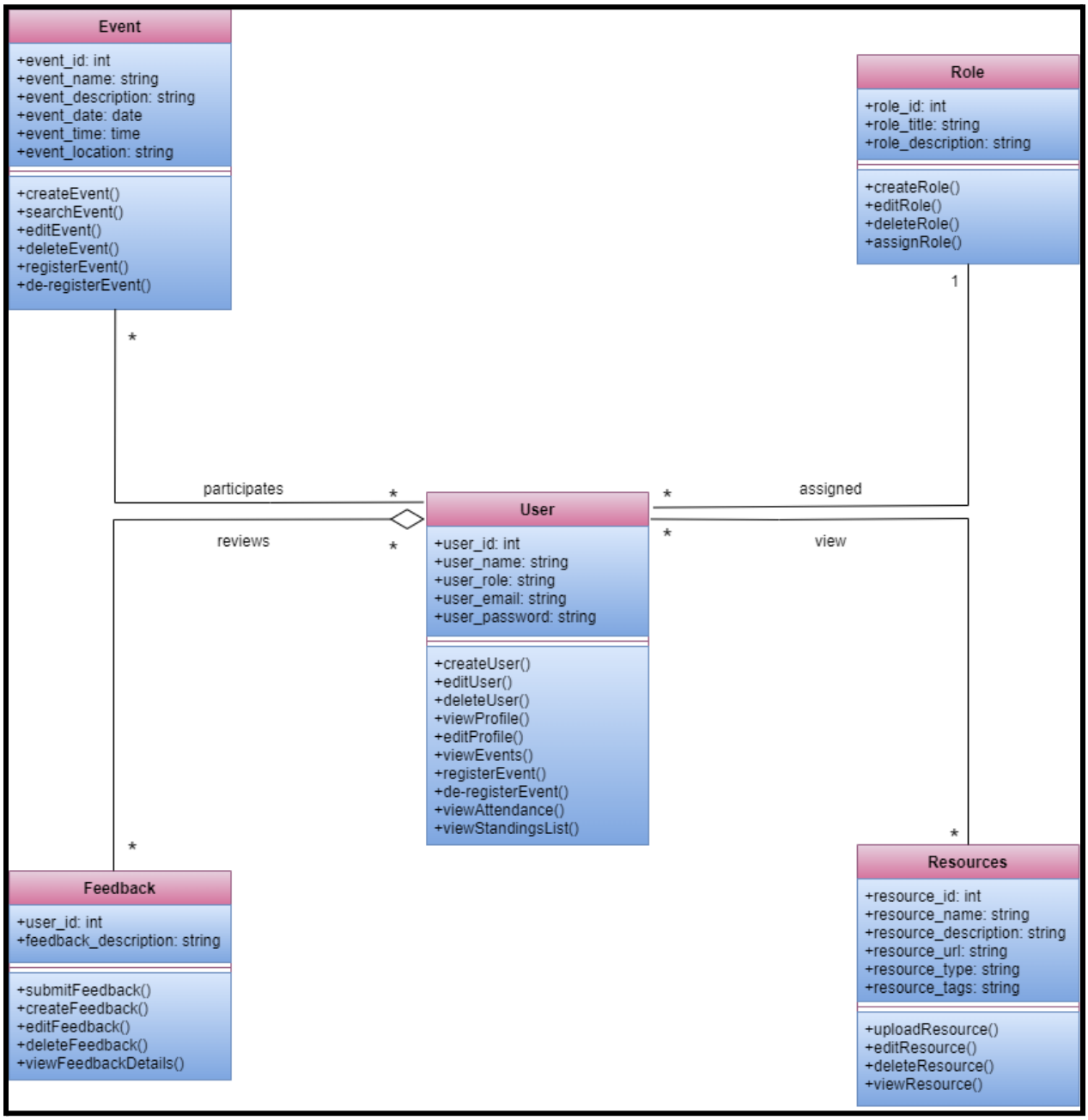
- **Operations:**
    - uploadResource(): allows an authorized user to upload a new resource
    - editResource(): allows an authorized user to edit an existing resource
    - deleteResource(): allows an authorized user to delete an existing resource
    - viewResource(): allows the user to view a specific resource

5) Feedback

- **Attributes:**
  - user_id: the unique identifier of the user
  - feedaback_description: the description of the feedback provided by the user

- **Operations:**
  - submitFeedback(): allows the user to submit the feedback form
  - createFeedback(): allows an authorized user to create a new feedback form
  - editFeedback(): allows an authorized user to edit existing feedback form
  - deleteFeedback(): allows an authorized user to delete existing feedback form
  - viewFeedbackDetails(): allows an authorized user to view the details of a specific feedback

Class Diagram



High-level System Design

We will be using a 3-layer architecture model that consists of a presentation tier, a business logic (application) tier, and a data tier. The presentation tier is a graphical user interface (GUI), the application tier handles logic, and the data tier stores information.

- ## Presentation Layer

    - The presentation layer typically consists of the user interface that users interact with while accessing the website. This includes the design, layout, and styling of the website's pages, as well as any interactive elements such as buttons, forms, menus, images, videos, audios, etc.

    - Some common technologies and tools used to create the presentation layer of a website include HTML, CSS, and JavaScript for the front-end development, and libraries or frameworks such as Bootstrap, React, etc. It is important to ensure that the presentation layer is accessible and responsive, meaning that it can be used on a variety of devices and screen sizes and that it adheres to web accessibility guidelines to ensure that all users can interact with the website.

    - This layer is distributed to a computing device using a web browser or a web-based application and is constructed with ReactJS. Application program interface (API) calls are the primary communication between the presentation tier and the other levels. The user interface should be easy to navigate and aesthetically pleasing. It should include a homepage, events page, and resources page.

- ## Business Logic Layer

    - The business logic layer, also called the application layer, is responsible for managing the business logic and data operations related to the club's activities. It typically includes functionality such as membership management, event registration, and data analytics.

    - The business logic layer acts as an intermediary between the presentation layer (which includes the website's user interface) and the data layer (which includes the database and other data sources). It ensures that data is validated, processed, and stored correctly and that the website functions smoothly and efficiently. We will be implementing the business logic layer using Node.js.

- ## Data Access Layer

    - The data access layer is responsible for managing the communication between the website's business layer and the database. It handles tasks such as retrieving, updating, and deleting data, and ensures that the data is stored and managed efficiently and securely.

    - We will use MongoDb to implement this layer, which should store and manage user account information, event details, and other club-related information.