

***Software Engineering  
Software Requirements Specification  
(SRS) Document***

**Spartan Event**

**<https://github.com/Natchapone/CSC340Project/tree/main>**

**02/20/2024**

**Version 1**

**By: Natchapone Janjon, Yanae Powell, Sebastian Duarte**

**The UNCG Academic Integrity Policy is followed for this  
project**

1. Introduction.....	3
1.1. Purpose.....	3
1.2. Document Conventions.....	3
1.3. Definitions, Acronyms, and Abbreviations .....	3
1.4. Intended Audience .....	3
1.5. Project Scope .....	3
1.6. Technology Challenges.....	4
1.7. References.....	4
2. General Description .....	4
2.1. Product Features.....	4
2.2. User Class and Characteristics .....	4
2.3. Operating Environment.....	4
2.4. Constraints .....	4
2.5. Assumptions and Dependencies .....	4
3. Functional Requirements .....	4
3.1. Primary.....	4
3.2. Secondary.....	5
3.3. Use-Case Model.....	5
3.3.1. Use-Case Model Diagram.....	5
3.3.2. Use-Case Model Descriptions.....	5
3.3.2.1. Actor: User (Yanae).....	5
3.3.2.2. Actor: Event Organizer (Natchapone) .....	5
3.3.2.3. Actor: Admin (Sebastian) .....	6
3.3.3. Use-Case Model Scenarios .....	6
3.3.3.1. Actor: User (Yanae).....	6
3.3.3.2. Actor: Evenet Organizer (Natchapone) .....	6
3.3.3.3. Actor: Admin (Sebastian) .....	7
4. Technical Requirements.....	7
4.1. Interface Requirements .....	7
4.1.1. Hardware Interfaces .....	7
4.1.2. Communications Interfaces .....	7
4.1.3. Software Interfaces .....	8
5. Non-Functional Requirements .....	9
5.1. Performance Requirements .....	9
5.2. Safety Requirements .....	9
5.3. Security Requirements .....	9

5.4. Software Quality Attributes .....	9
5.4.1. Availability .....	9
5.4.2. Correctness.....	9
5.4.3. Maintainability .....	9
5.4.4. Reusability .....	9
5.4.5. Portability.....	9
5.5. Process Requirements .....	10
5.5.1. Development Process Used.....	10
5.5.2. Time Constraints .....	10
5.5.3. Cost and Delivery Date .....	10
5.6. Other Requirements .....	10
6. Design Documents .....	11
6.1. Software Architecture .....	11
6.2. High-Level Database Schema.....	12
6.3. Software Design.....	13
6.3.1. State Machine Diagram: Actor Name (Responsible Team Member) .....	13
6.3.2. State Machine Diagram: Actor Name (Responsible Team Member) .....	14
6.3.3. State Machine Diagram: Actor Name (Responsible Team Member) .....	15
6.4. UML Class Diagram .....	16
7. Scenario.....	17
7.1. Brief Written Scenario with Screenshots .....	17

# 1. Introduction

## 1.1. Purpose

The goal of Spartan Event is to provide a helpful tool for event organizers that want to get in touch with UNCG students by allowing them to post their events, read comments, and receive RSVPs.

## 1.2. Document Conventions

The purpose of this Software Requirements Document (SRD) is to describe the client-view and developer-view requirements for Spartan Event. In the client-oriented requirements, the client's perspective of the software will be described. There will be a description of how different users can interact with the software. In the developer-oriented requirements, the developer's perspective of the software will be described. There will be a description of the back-end requirements: data, performance, security, and any other requirements developers should know.

## 1.3. Definitions, Acronyms, and Abbreviations

HTML	Hypertext Markup Language. This is a language used in web development. It will be used to structure and design the website.
CSS	Cascading Style Sheets. Used alongside HTML for web development. It will be used to style the website.
JS	JavaScript. This is a programming language used for web development. It will be used to program the behavior of the website.
API	Application Programming Interface. It will be used to access third party resources needed by the website.
RSVP	Comes from french and translates to "Respond, if you please." It will be used by users to let organizers know they plan to attend an event.

## 1.4. Intended Audience

The first three sections of the document are intended for all readers. The other sections are intended for the developers and project managers.

Stakeholders: N/A

Developers: Natchapone Janjon, Yanae Powell, Sebastian Duarte.

Project Managers: Natchapone Janjon, Yanae Powell, Sebastian Duarte.

Users: Event Organizers, Users, and Admins.

## 1.5. Project Scope

The goal of the software is to allow event organizers to promote their events to UNCG students. This aligns with the overall business goals of student events because by making these events known, resources are less likely to be wasted and event profits increase.

The benefits of the software to business include:

- Simplifying the process of promoting events for event organizers, allowing them to spend more time and resources on the event itself.
- Give students a single, easy-to-use platform to keep on top of events they can attend, making it more likely for them to attend.
- Ensuring that civil communication is available so that students can express their thought on events, and event organizers can get feedback that can help improve their events.

### **1.6. Technology Challenges**

- The software requires Functional Web Browser.
- The software requires an internet connection.

### **1.7. References**

Non-Applicable.

## **2. General Description**

### **2.1. Product Features**

The features for the admin include registering, logging in and out, managing users, and moderate reactions (including comments and any interactions with a posted event). Features for event organizer include registering, logging in and out, creating and deleting events, and editing events. Features for the user include RSVPing for events and interacting with event (leaving comments, asking questions, etc).

### **2.2. User Class and Characteristics**

Our web application does not require any prior knowledge aside from knowing how to operate a computer and knowing how to operate a web browser.

### **2.3. Operating Environment**

Our web application is designed to operate on the web on many different devices.

### **2.4. Constraints**

To prevent a cluttered user page, we made it so that the map would need to be toggled to be viewed.  
To handle authorization, we used express-session.

### **2.5. Assumptions and Dependencies**

The webpage depends on node to run as well as these node dependencies: express, multer, sqlite, sqlite3, method-override, express-session, ejs, and better-sqlite3. The webpage also uses the Google Maps API to show maps of the location of the events.

## **3. Functional Requirements**

### **3.1. Primary**

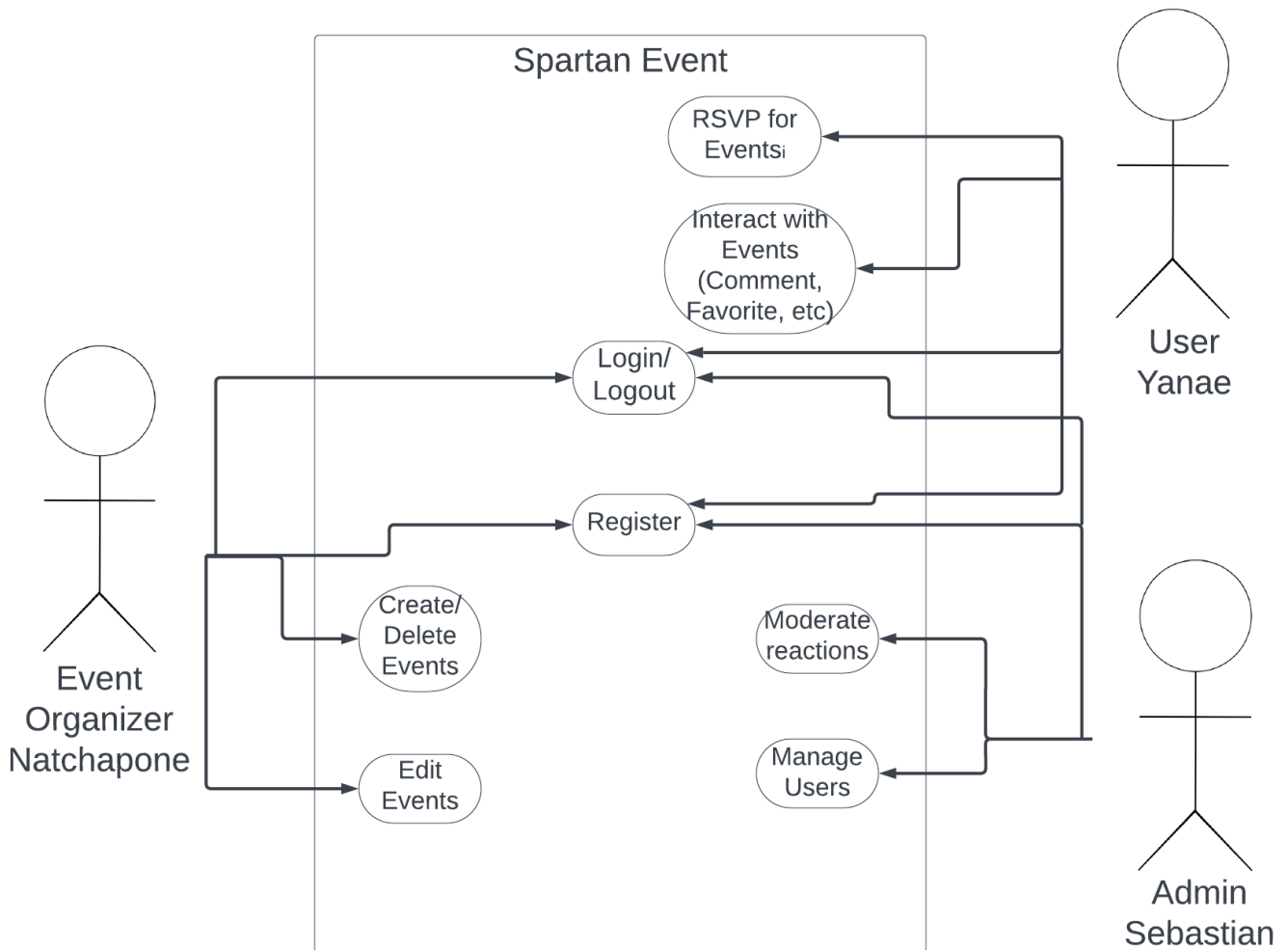
- FR0: The system will allow the user to view posted all events or by category. Events will have related information such date, time, and other description.
- FR1: The system will allow certain users to post events and manage them as well as interact with other users viewing the events.
- FR2: The system will allow certain users to oversee and make adjustments to content regarding the events as well as managing other users and enforcing system rules.

### 3.2. Secondary

- Sign-in/registration system that will categorize users, allowing different types of users certain actions based on their type such as view, create, interact with, manage, or delete events.
- Tagging system that will allow users to filter events based on certain characteristics.

### 3.3. Use-Case Model

#### 3.3.1. Use-Case Model Diagram



#### 3.3.2. Use-Case Model Descriptions

##### 3.3.2.1. Actor: User (Yanae)

- **RSVP for Events:** A user can RSVP and un-RSVP for events so that organizers can gauge the number of people coming to the event.
- **Interact with Events:** A user can interact with events by commenting, favoriting, etc. So that the organizers have a line of communication with students.

##### 3.3.2.2. Actor: Event Organizer (Natchapone)

- **Post Events:** The event organizer can create a post events by listing an event and related information for user sign up.
- **Manage Events:** The event organizer keeps the event updated, in the case of changes for events or on-going/repeating events.

### 3.3.2.3. Actor: Admin (Sebastian)

- **Moderate Reactions:** An admin can delete comments, delete events, flag organizers from events, flag users from comments, and unflag.
- **Manage Users:** An admin can delete, and flag users and organizers.

## 3.3.3. Use-Case Model Scenarios

### 3.3.3.1. Actor: User (Yanae)

- **Use-Case Name:** RSVP for Events
  - **Initial Assumption:** A user can RSVP for events.
  - **Normal:** The user RSVP for an event.
  - **What Can Go Wrong:** The user RSVPs for an event that they have already RSVP'd to. The system should not accept the second RSVP and give an error message to the user.
  - **Other Activities:** A user can un-RSVP for events.
  - **System State on Completion:** An RSVP is created and it is both visible to the user and the event organizer.
- **Use-Case Name:** Interact with Events
  - **Initial Assumption:** A user can leave comments on events.
  - **Normal:** The user leaves a comment on an event.
  - **What Can Go Wrong:** The user leaves a comment on an event that has been deleted. The system should check the status of the event before receiving the comment.
  - **Other Activities:** A user can favorite an event.
  - **System State on Completion:** A comment is left in the comment section of an event and it is visible to all three actors.

### 3.3.3.2. Actor: Event Organizer (Natchapone)

- **Use-Case Name:** Post Events
  - **Initial Assumption:** The Event Organizer has the ability to create and post events.
  - **Normal:** The Event Organizer lists details (date, time, description) about the event that will be displayed when posted and can put tags on the event to specify the category of the event.
  - **What Can Go Wrong:** n/a
  - **Other Activities:** Events can be deleted by its Event Organizers
  - **System State on Completion:** Events are created and posted. Events can be viewed by users and lists related info.
- **Use-Case Name:** Manage Event
  - **Initial Assumption:** Event Organizer keeps the event info updated
  - **Normal:** Event Organizer selects and edit related info to keep the event up to date as well as answer any comments on the event.
  - **What Can Go Wrong:** n/a
  - **Other Activities:** Event Organizer interacts with users and make changes to events accordingly if need be.
  - **System State on Completion:** Events can be edited and Event Organizer can interact with users via responses to comments.

### 3.3.3.3. Actor: Admin (Sebastian)

- **Use-Case Name:** Moderate Reactions
  - **Initial Assumption:** An admin can delete comments.
  - **Normal:** The admin deletes unpleasant comments.
  - **What Can Go Wrong:** An admin deletes a comment that has already been deleted. The system should detect this and prevent possible loss of data.
  - **Other Activities:** An admin can delete events.
  - **System State on Completion:** The comment is no longer visible to users.
- **Use-Case Name:** Manage Users
  - **Initial Assumption:** An admin can delete and flag users.
  - **Normal:** The admin flags users that have violated terms of service in the past and deletes users.
  - **What Can Go Wrong:** An admin deletes a user that has already been deleted. The system should send an error message in case this happens.
  - **Other Activities:** An admin deletes organizers.
  - **System State on Completion:** A user and all their comments are off the platform and a flag is placed on certain users, only visible to admins.

## 4. Technical Requirements

### 4.1. Interface Requirements

#### 4.1.1. User Interfaces

The web app starts out with a welcome page that displays options to login or register as different types of users. Each of these buttons will take you to a separate page that will provide corresponding functions such as registering as a normal user or an event organizer. Once the user is registered and logged in, they will be able to browse current events at UNCG by scrolling through the page. They can filter events by category or look up specific event by name. All of the aforementioned will be done via the header region of the web app where there will be buttons that will allow the user to access certain actions.

#### 4.1.1. Hardware Interfaces

The web app is accessible on any device with an internet access and the ability to view and interact with web pages including but not limited to desktops and laptop computers, and other devices such as smartphones.

#### 4.1.2. Communications Interfaces

Users must be able to connect to the internet and local database from which the web app utilizes (to be determined). The HTTP must be able to connect to an API which will return relevant data regarding current events at UNCG (to be determined).



#### **4.1.3. Software Interfaces**

We will use Bootstrap to help build the frontend as well as Spring Boot, JPA, and Java/JavaScript to build the backend and frontend to backend functionality.

## 5. Non-Functional Requirements

### 5.1. Performance Requirements

- NFR0 (R): The local copy of the user database will consume less than 20 MB of memory.
- NFR1 (R): The local copy of the event and comments database will consume less than 20 MB of memory.
- NFR2 (R): The system (including the local copies of the databases) will consume less than 80 MB of memory.
- NFR3 (R): The novice user will be able to register for an account in less than 5 minutes.
- NFR4 (R): The expert user will be able to register for an account in less than 2 minutes.
- NFR5 (R): The novice user will be able to login in less than 3 minutes.
- NFR6 (R): The expert user will be able to login in less than 1 minute.
- NFR7 (R): The novice user will be able to RSVP and comment on events in less than 5 minutes.
- NFR8 (R): The expert user will be able to RSVP and comment on events in less than 2 minutes.
- NFR9 (R): The novice event organizer will be able to create, delete, or edit events in less than 10 minutes.
- NFR10 (R): The expert event organizer will be able to create, delete, or edit events in less than 5 minutes.
- NFR11 (R): The novice admin will be able to moderate reactions and manage users in less than 15 minutes.
- NFR12 (R): The expert admin will be able to moderate reactions and manage user in less than 10 minutes.

### 5.2. Safety Requirements

- NFR13 (R): The system will prevent banned emails from signing up again.

### 5.3. Security Requirements

- NFR14 (R): The system will not ask for any sensitive information from the user and will delete any comment or event that requests it.

### 5.4. Software Quality Attributes

#### 5.4.1. Availability

- NFR15 (R): The system will be available 24/7.

#### 5.4.2. Correctness

- NFR16 (R): Admins will overview everything to prevent false information from being spread and will delete events that don't meet certain criteria.

#### 5.4.3. Maintainability

- NFR17 (R): The system will be updated as issues arise to ensure the best user experience.

#### 5.4.4. Reusability

- NFR18 (R): The software can be reused indefinitely.

#### 5.4.5. Portability

- NFR19 (R): The system will be available as long as the user has an internet connection.

## **5.5. Process Requirements**

### **5.5.1. Development Process Used**

Incremental Development Model

### **5.5.2. Time Constraints**

Prototype: February 27, 2024.

Design Document: April 02, 2024.

Code Review: April 16, 2024.

Penultimate version: April 25, 2024.

### **5.5.3. Cost and Delivery Date**

Cost: \$0

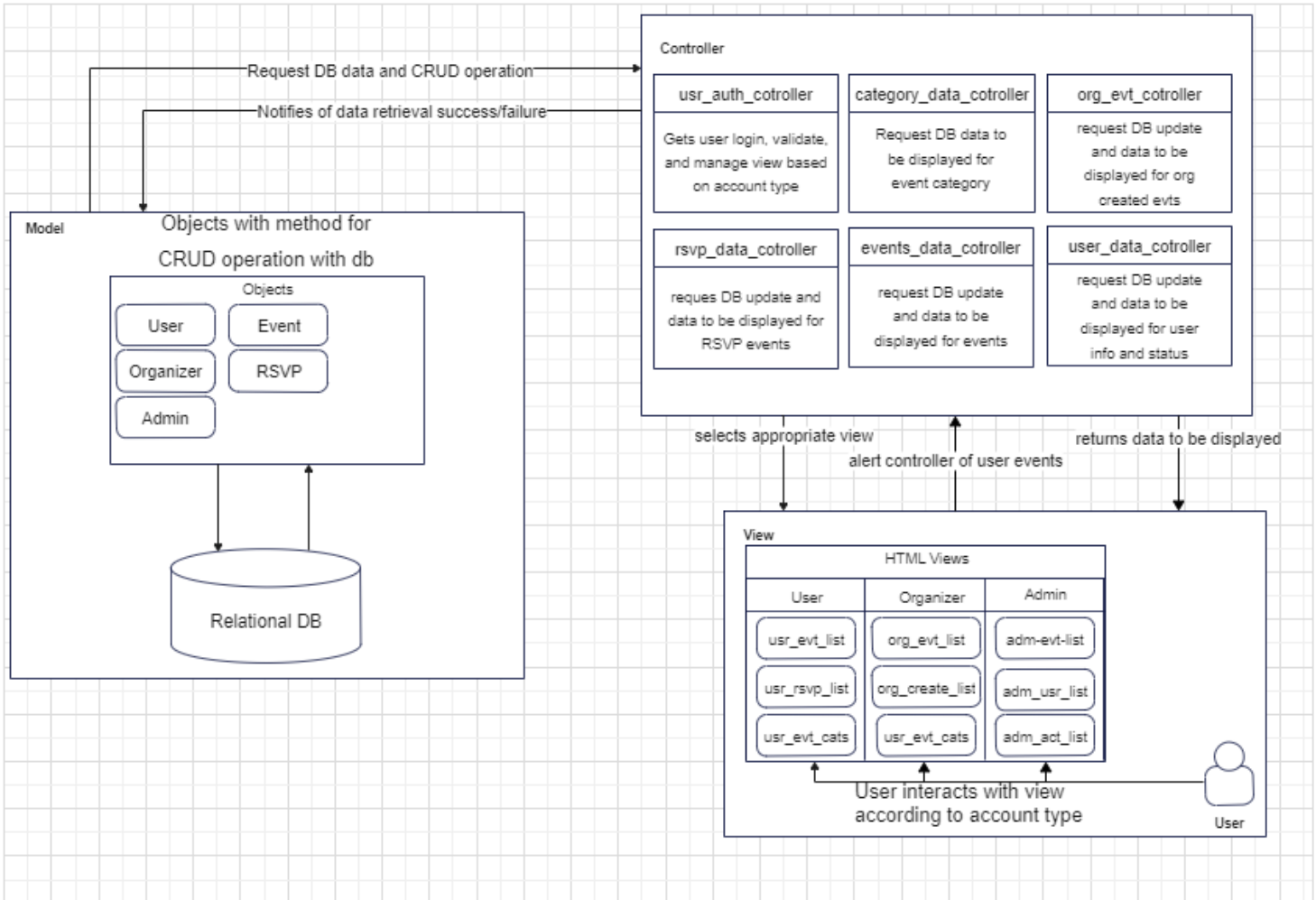
Devilery Date: May 01, 2024.

## **5.6. Other Requirements**

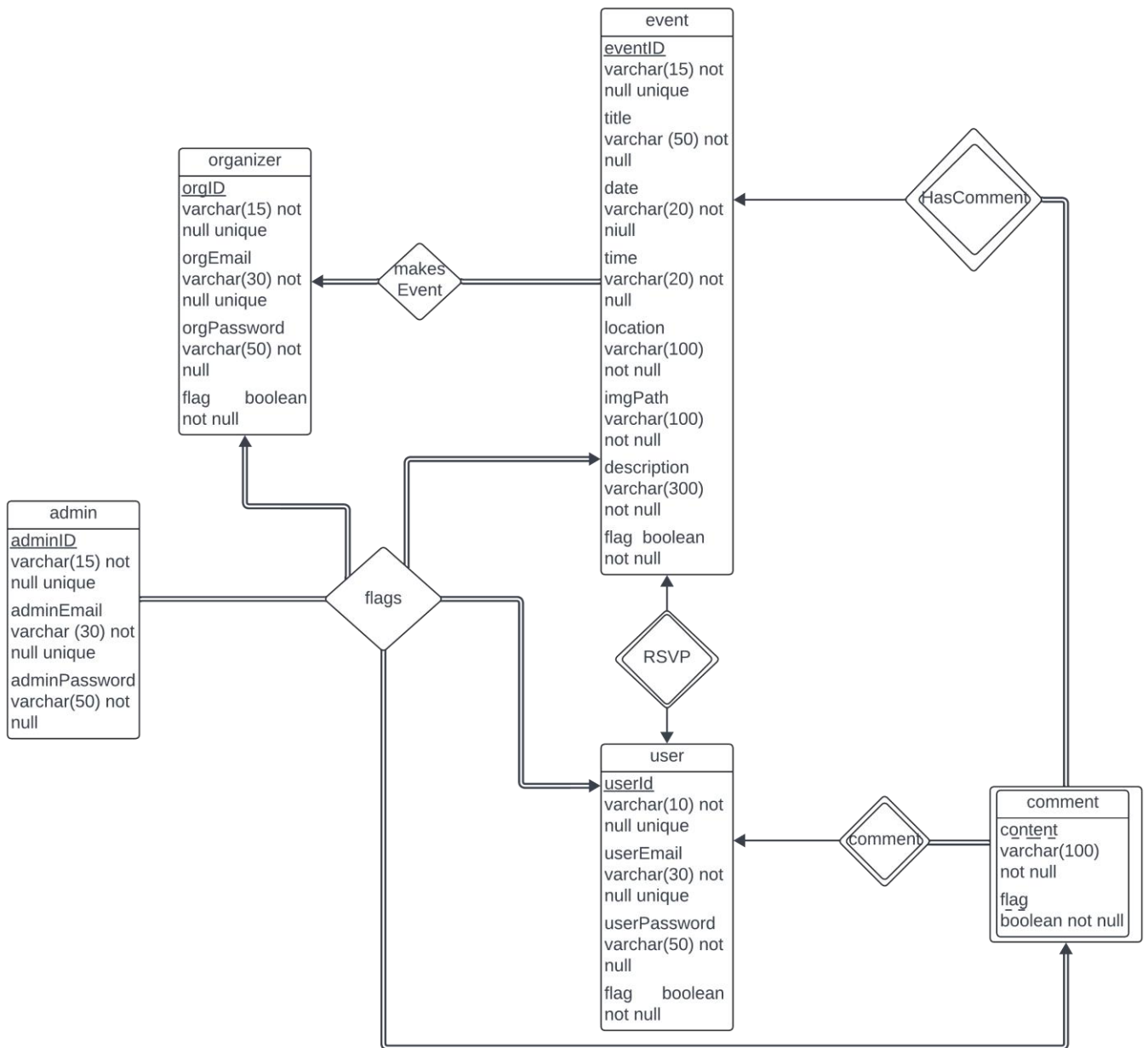
Non-Applicable.

## 6. Design Documents

### 6.1. Software Architecture

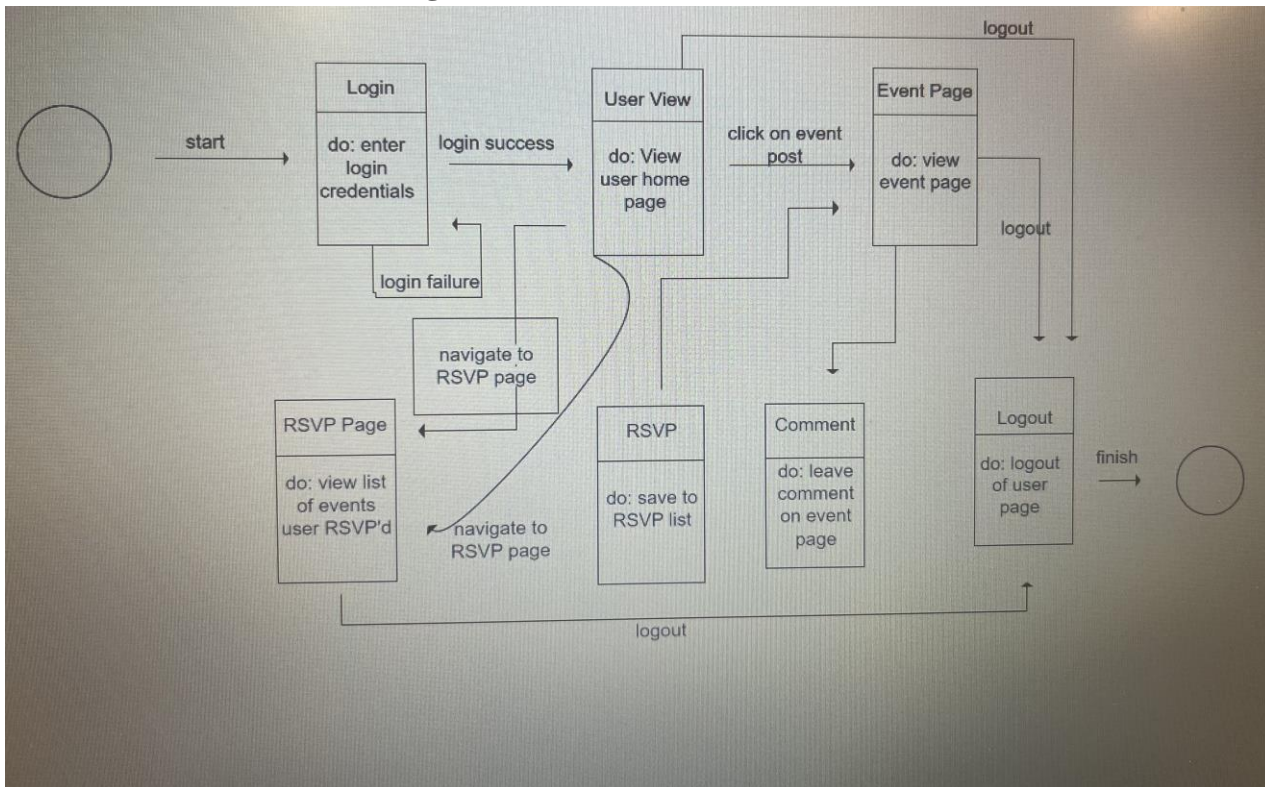


## 6.2. High-Level Database Schema

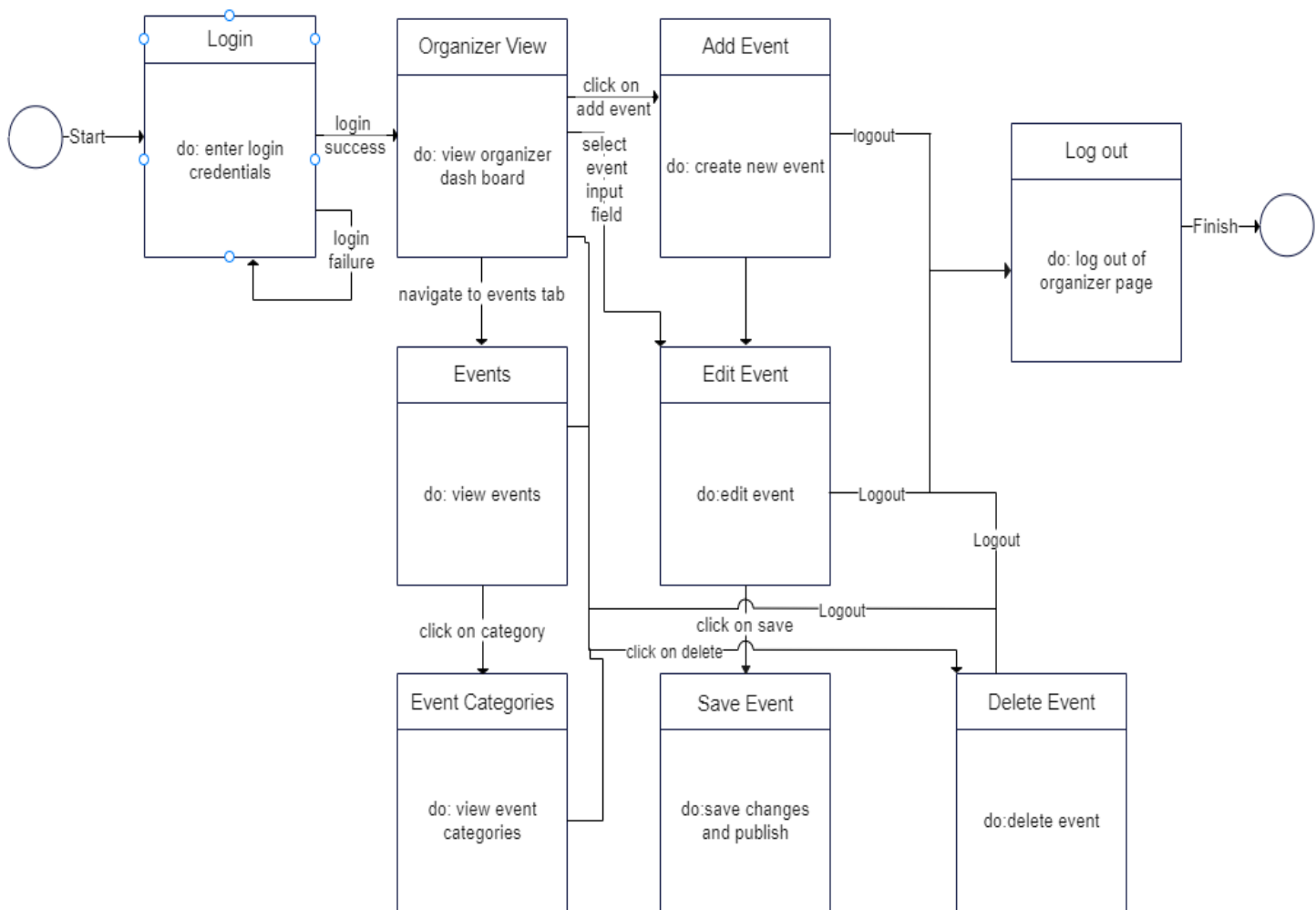


## 6.3. Software Design

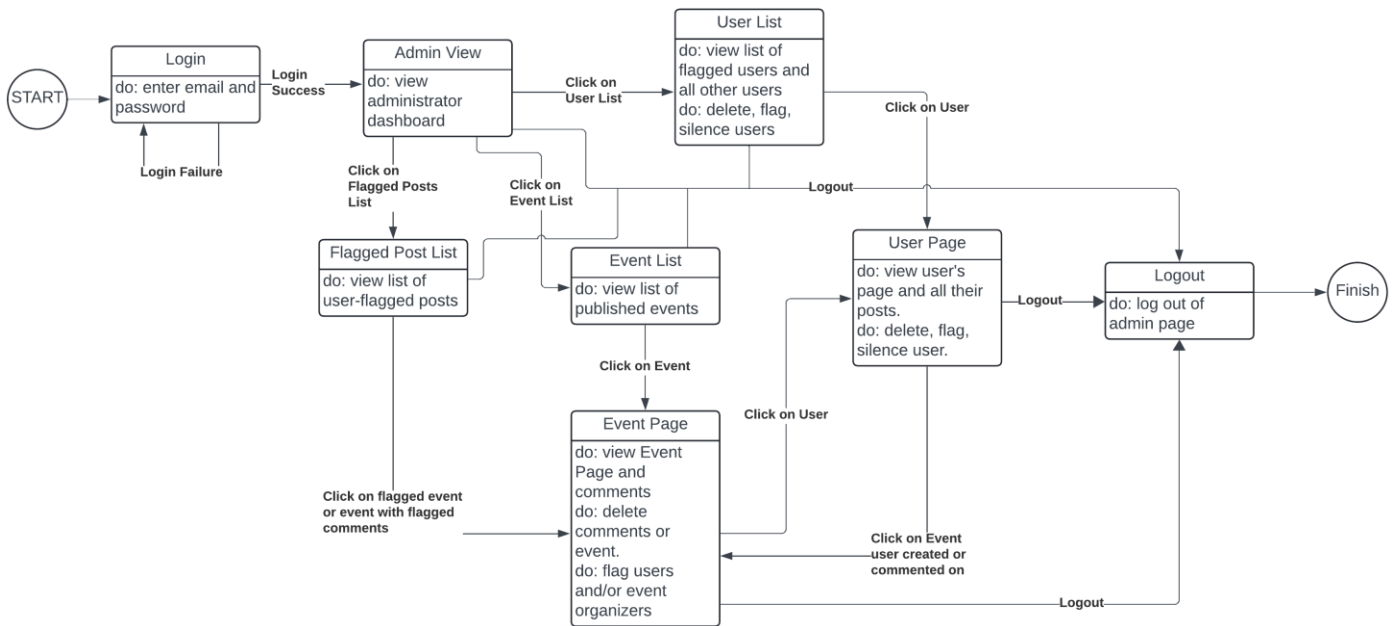
### 6.3.1. State Machine Diagram: User (Yanae Powell)



### 6.3.2. State Machine Diagram: Event Organizer (Natchapone Janjon)

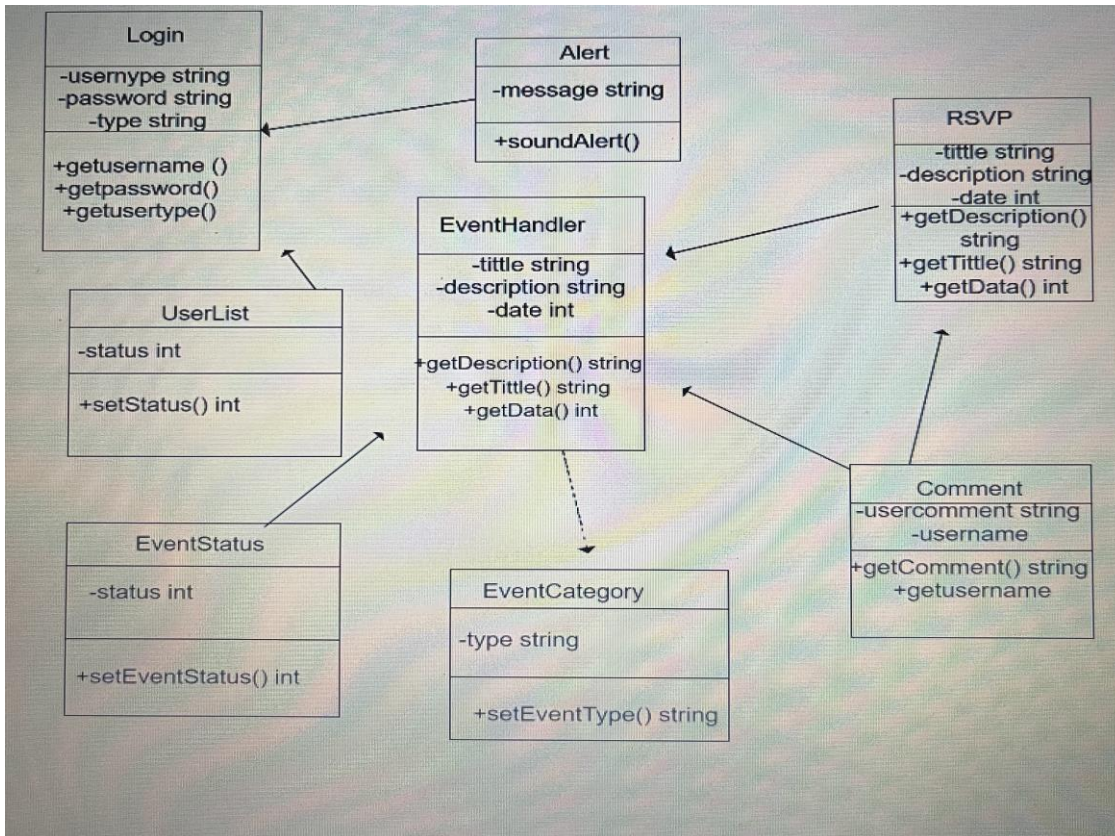


### 6.3.3. State Machine Diagram: Admin (Sebastian)





## 6.4. UML Class Diagram

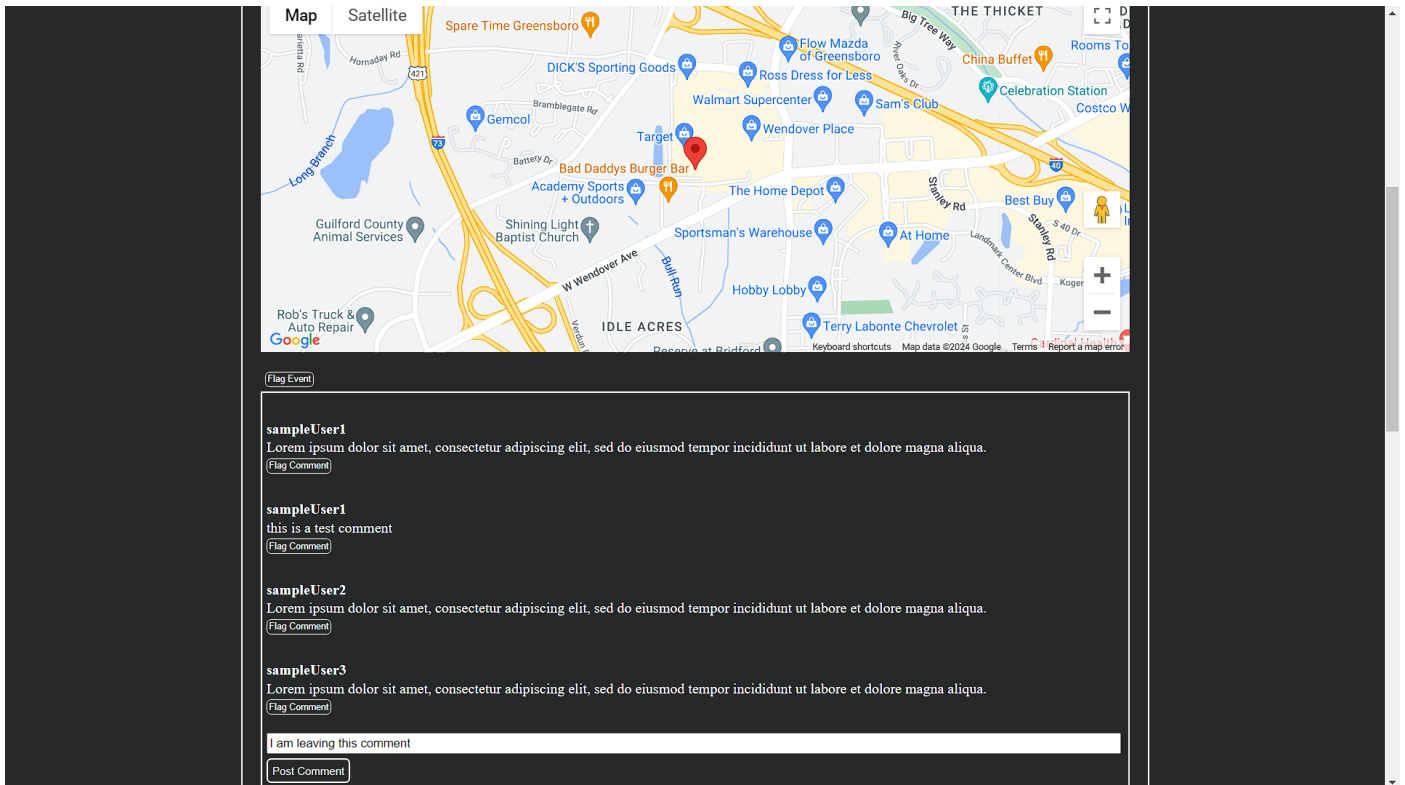


## 7. Scenario

### 7.1. Brief Written Scenario with Screenshots

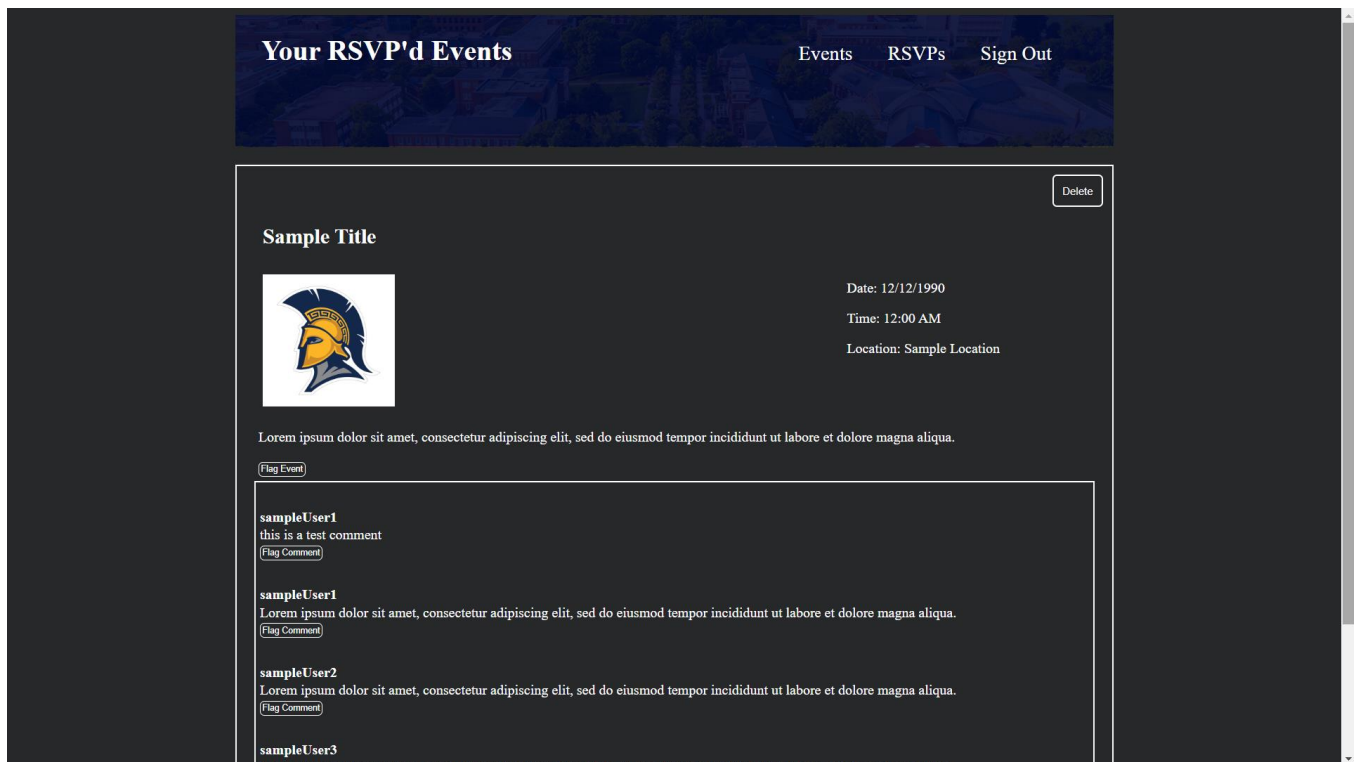
User: Interact with Events use case:

User SampleUser1 logs in, they view their events feed, a map for the event, flag an event, flag a comment, and leave a comment. SampleUser1 exits.



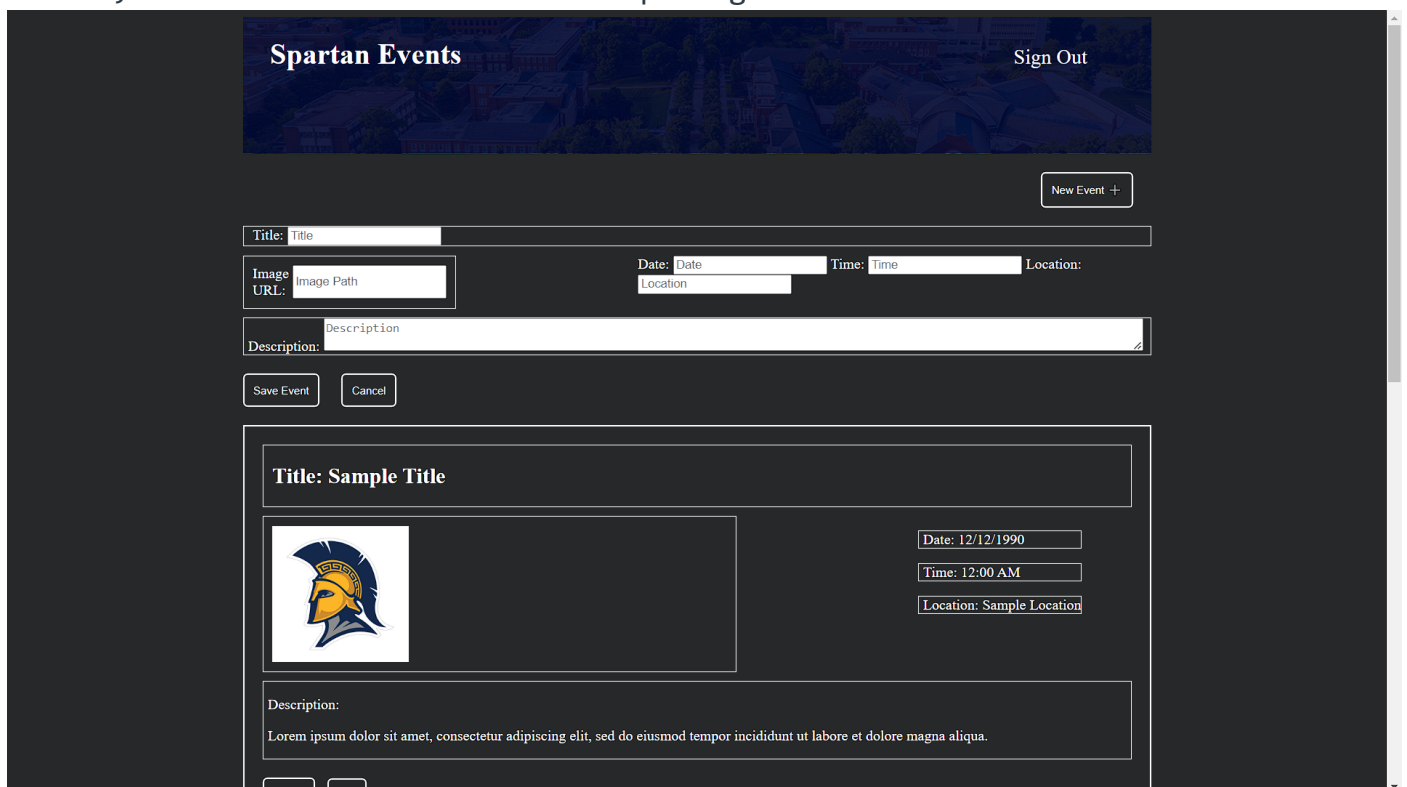
User: RSVP for Events use case:

User SampleUser1 logs in, they view their events feed, RSVP for an event, go to their RSVPs page, view their RSVP, and delete their RSVP. SampleUser1 exits.



Organizer: Create/Delete Events use case:

Organizer sampleOrganizer1 logs in, view their events, presses the new event button, fills in new event form, creates new event, views new event in their events, deletes their new event. sampleOrganizer1 exits.



Organizer: Edit Events use case:

Organizer sampleOrganizer1 logs in, view their events, presses the edit button under one of their events, edits the event, and sees the edited event in their events. sampleOrganizer1 exits.

Title: Sample Title

Image URL: /images/UNCGlogo.jpg

Date: 12/12/1990

Time: 12:00 AM

Location: Sample Location

Description:

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.


Save Event

Cancel

Admin: Moderate Reactions use case:

Admin SampleAdmin1 logs in, view flagged events and flagged comments page, unflags an event, unflags a comment, goes to a flagged event, flags a comment, flags an event, deletes a comment, deletes an event, and returns to flagged events page. SampleAdmin1 exits.

Title: Sample Title



Date: 12/12/1990

Time: 12:00 AM

Location: Sample Location

Description:

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

DELETE

FLAG

Comments:

From: sampleUser1

this is a test comment

DELETE

FLAG

From: sampleUser2

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

DELETE

FLAG

Admin: Manage Users use case:

19

Admin SampleAdmin1 logs in, view flagged events and flagged comments page, goes to flagged Users page, unflags an event organizer, unflags a user, delete a user. SampleAdmin1 exits. Deleted user tries to log in but can't.

