Software Engineering
Software Requirements Specification (SRS) Document

Spartan Event

02/20/2024

Version 1

By: Natchapone Janjon, Yanae Powell, Sebastian Duarte

The UNCG Academic Integrity Policy is followed for this project

Table of Contents

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 nad 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.

Formatted: Centered

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

1.7. References

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

2.5. Assumptions and Dependencies

3. Functional Requirements

3.1. Primary

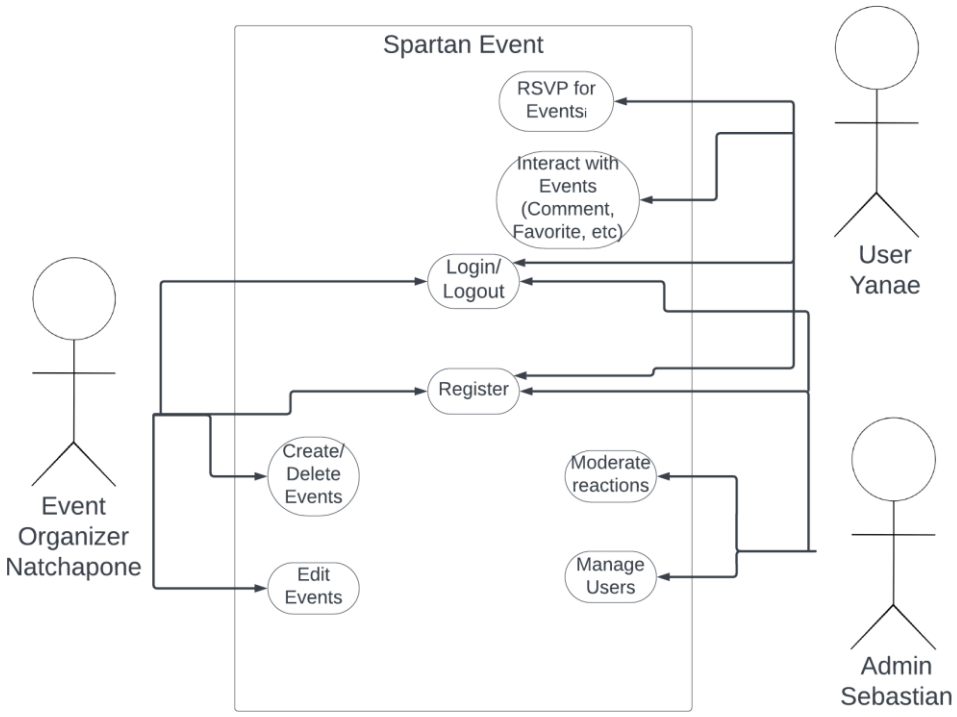
- FR0: The system will allow the user to view posted all events or by category. Events will have related information such date, time, and other description.
- FR1: The system will allow certain users to post events and manage them as well as interact with other users viewing the events.
- FR2: The system will allow certain users to oversee and make adjustments to content regarding the events as well as managing other userers 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 gage 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, give warnings to users, and filter the contents of events and comments.
- **Manage Users:** An admin can delete, silence, and flag users.

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: Evenet 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 reponses to comments.

3.3.3.3. Actor: Admin (Sebastian)

- **Use-Case Name**: Moderate Reactions
    - **Initial Assumption**: An admin can delete comments and give warnings to users.
    - **Normal**: The admin deletes unpleasant comments and gives warnings to users that posted those comments.

- **What Can Go Wrong**: An admin deletes a comment that ahs already been deleted. The system should detect this and prevent possible loss of data.
- **Other Activities**: Admin manages filters of unacceptable language so that comments and events with certain words won't be allowed to publish.
- **System State on Completion**: The comment is no longer visible to users and the user has a warning on their account.

- **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 that have three offenses.
  - **What Can Go Wrong**: An admin deletes a user that has already been deleted. The system should send an error message in the case that this happens.
  - **Other Activities**: An admin can silence users.
  - **System State on Completion**: An 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 display obtions 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 browse current events at UNCG by scrolling through the page. They can filter evernts by category or look up specific event by name. All of the aforementioned wiill be done via the header region of the web app where there will be bottons 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 filter innapropriate language so that it can be used by everyone safely.

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 for 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 to be 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

6. Design Documents

6.1. Software Architecture

6.2. High-Level Database Schema

6.3. Software Design

6.3.1. State Machine Diagram: Actor Name (Responsible Team Member)

6.3.2. State Machine Diagram: Actor Name (Responsible Team Member)

6.3.3. State Machine Diagram: Actor Name (Responsible Team Member)

6.4. UML Class Diagram

7. Scenario

7.1. Brief Written Scenario with Screenshots