Software Engineering
Software Requirements Specification (SRS) Document

WildFinder

2/20/2024

1.0

By: Jacob Greene, Christian Wilson, & Andrew Nice

[Our words and actions will reflect Academic Integrity.
We will not cheat or lie or steal in academic matters.
We will promote integrity in the UNCG community.]

Table of Contents

1. Introduction

1.1. Purpose

The goal of wildFinder is for hikers to meet each other while helping them discover new trails in their county to explore. In order to achieve this goal, WildFinder allows users to communicate with other hikers to hike together, allows experienced hikers to lead groups through trails, allows users to rate trails, and shows updates at trails to help users decide where to explore.

1.2. Document Conventions

The purpose of this Software Requirements Document (SRD) is to describe the general-user, operators, and admin requirement for the WildFinder system. Client-oriented requirements describe the system from the client's perspective. These requirements include a description of the different types of users served by the system. Developer-oriented requirements describe the system from a software developer's perspective. These requirements include a detailed description of functional, data, performance, and other important requirements.

1.3. Definitions, Acronyms, and Abbreviations

- Java
    - A programming language originally developed by James Gosling at Sun Microsystems.
- MySQL
    - Open-source relational database management system.
- HTML
    - Hypertext Markup Language. This is the code that will be used to structure and design the web application and its content.
- CSS
    - Placeholder
- SpringBoot
    - An open-source Java-based framework used to create a micro Service. This will be used to create and run our application.
- MVC
    - Model-View-Controller. This is the architectural pattern that will be used to implement our system.
- Spring Web
    - Will be used to build our web application by using Spring MVC. This is one of the dependencies of our system.
- Thymeleaf

    - A modern server-side Java template engine for our web environment. This is one of the dependencies of our system.
- NetBeans
    - An integrated development environment (IDE) for Java. This is where our system will be created.
- API
    - Application Programming Interface. This will be used to implement a function within the software where the current date and time is displayed on the homepage.

1.4. Intended Audience

The stakeholder(UNCG): 1-5
Project Manager(Ms. Ntini)
Project Developer (Christian Wilson): 3.3.2.1, 3.3.3.1
Project Developer (Andrew Nice): 3.3.2.2, 3.3.3.2
Project Developer (Jacob Greene): 3.3.2.3, 3.3.3.3

1.5. Project Scope

The goal of this software is to provide a platform where casual hikers can find other people to invite and create group hikes within their area, allow hikers who know their local trails to meet new hikers who are interested in having a mentor, provide lists of the parks/trails available to a user, as well as allow the local park/trail operators to provide updates and add new parks/trails once they are open to the public.

Benefits of the project include:

- Increased accessibility for the general public to enjoy the beauty of nature while bringing more engagement and therefore more use and knowledge to local parks/trails.
- Increasing safety and proper use of parks/trails through group activities and mentorship.
- Increasing the likelihood of more visitors to parks/trails.
- Allow park/trail operators to post announcements to potential visitors.

1.6. Technology Challenges

1.7. References

2. General Description

2.1. Product Features

For users, the product features include the ability to create and manage an account. This will let them find new trails as well as meet new people through chat features(1 on 1 chat and group chat). Users can also rate the trails that they have hiked and apply to become a guide for trails. In order to become a guide they must submit a form that will be reviewed by a park operator. Park operators will be able to create an account but must apply to do so through a form that must be approved by a web admin. Upon approval and account creation park operators can upload and update trails, as well as organize guided trail tours. Park operators have access to chat features as well. The web admin will be able to access code, generate status reports, and moderate accounts.

2.2. User Class and Characteristics

Our website application does not expect our users to have knowledge of the parks unless they apply to become guides for the parks. Our website application assumes those who have been confirmed as park administrators have knowledge of the parks they create and have the authority to provide updates for those parks.

2.3. Operating Environment

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

2.4. Constraints

2.5. Assumptions and Dependencies

The software will be dependent on Spring Web, Spring Admin Tools, and Thymeleaf in order to create and execute the MVC architecture that will be developed with NetBeans. The application will use the Leaflet API(https://leafletjs.com/) for the interactive map of the Piedmont Region that will show the different counties and lead you to the page of the trails in a specified county. We plan to use a chat api for the chat features but yet to decide on one.

3. Functional Requirements

3.1. Primary

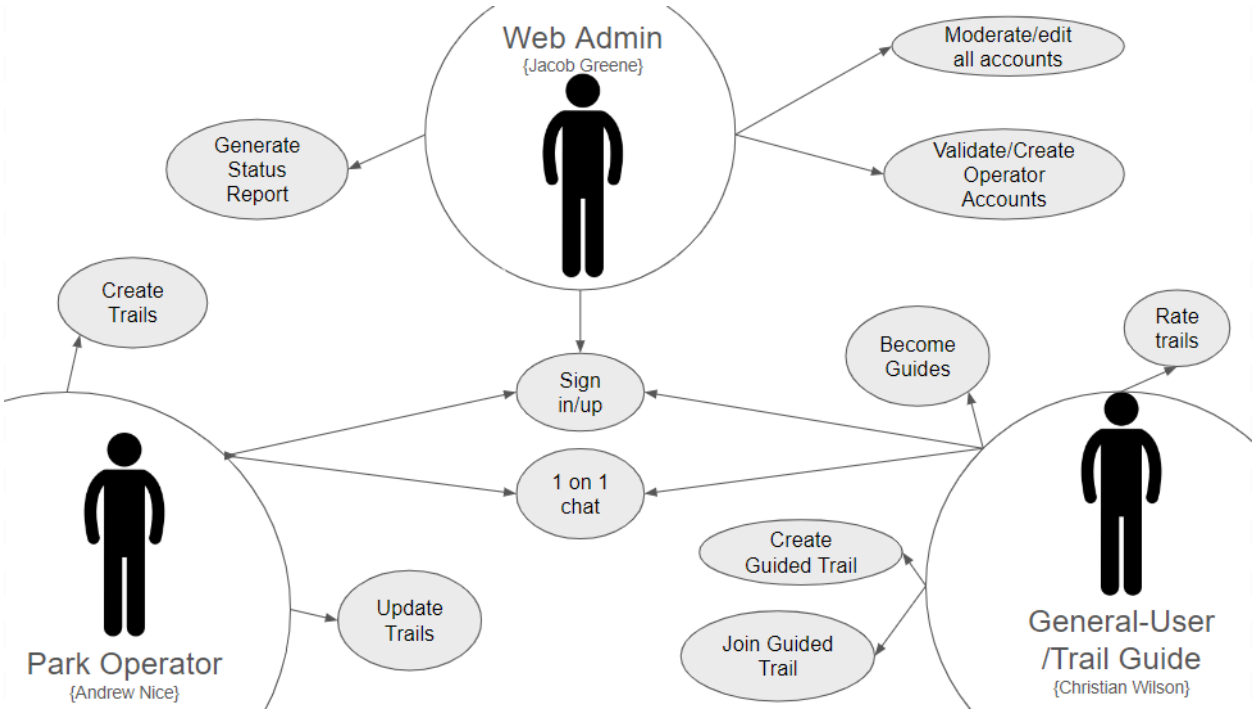
- FR0: The system will allow general users to look up local trails based on a county selection in the piedmont area of North Carolina
- FR1: The system will allow general users to meet and chat with other general users and trail guides for group hiking
- FR2: The system will allow park operators to upload new trails and post descriptions and updates on those trails
- FR3: The system will allow trail guides (elevated general users) to create guided hikes at specific trails.

3.2. Secondary

- Password protection for all users.
- Ability to display information looked up.

3.3. Use-Case Model

3.3.1. Use-Case Model Diagram



3.3.2. Use-Case Model Descriptions

3.3.2.1.

- Actor: General-user or Trail Guide (Christian Wilson):
    - Sign In/Up: General Users can create an account and sign into said account.
    - 1 on 1 chat: General users can chat one on one with other general users, trail guides, or park operators.
    - Group chat: General Users can chat in groups for setting up group trails with other General Users, Trail Guides, and Park Operators.
    - Join Guided Trail: The General Users can sign up for time slots set by Park Operators for Guided Trails which contain at least one Guide.
    - Rate Trails: The General Users can rate trails they have hiked.
    - Become Guides: The General Users can sign up to become guides for a specific park's trails, which shows that a user is competent enough to safely lead others through a trail in a specific park.
    - Create Guided Hike: The Guides can allocate time slots to lead General Users on Guided hikes through the park's trail/trails.

3.3.2.2.

- Actor: Park Operator (Andrew Nice)
    - Sign In/Up: A Park Operator can create an account by filling out a form and can sign into that account.
    - 1 on 1 chat: The Park Operators can chat one on one with General Users, Trail Guides, or other Park Operators.
    - Create Trail: The Park Operators can create new trails.
    - Update Trails: The Park Operators can update trails and change status of trails to show availability.

3.3.2.3.

- Actor: Web Admin (Jacob Greene)
    - Sign-in to admin account access admin page: The Web Admin is able to sign into their account.
    - Generate Reports: The Web Admin is able to generate a report on the statistics of the website. Reports could contain new users, returning users, list of trails, list of new trails (day, month year drilldown),
    - Moderate/edit all accounts: The Web Admin can delete accounts found to be problematic.
    - Validate/Create park operator accounts: The Web Admin can approve applications for people to become Park Operators after checking their validity with the people running the trail IRL.

3.3.3. Use-Case Model Scenarios

3.3.3.1. Actor: General User (Christian Wilson)

- **Use-Case Model Name: Sign In/Up**

- Initial Assumption: The General Users have an email and can access the website or have already created an account.
- Normal: A preexisting user will enter their information and login, or a new user will click a sign up option and enter an email and password.
- What Can Go Wrong: An user could lose their password or enter an incorrect email.
- Other Activities: A user could switch to the login page if they accidentally selected to sign up for a new account.
- System State on Completion: The User will have logged in and is redirected to the page to select a county.

- **Use-Case Model Name: 1 on 1 Chat**

- Initial Assumption: The General Users can navigate to the chat function for speaking with a guide or general user
- Normal: A General User of Trail Guide can select a person to hike with and initiate a 1 on 1 chat interaction to plan for the hike.
- What Can Go Wrong: N/A
- Other Activities: A User could switch to chat with another user.
- System State on Completion: A User is now chatting with another user and can terminate the chat when done with the conversation.

- **Use-Case Model Name**: **Join Guided Hikes**

- Initial Assumption: The General Users can navigate to specific trails in a county.
- Normal: After choosing a trail in their county, a User can join a Guided Trail which is shown on the trail's page.
- What Can Go Wrong: Guides could create Guided Hikes and not respond with details or not show up to the hike.
- Other Activities: Users would get information about who is leading the hike to initiate conversation.
- System State on Completion: The User is now attached to the Guided Hike in the system, letting the Gude know that another person intends to join their hike.

- **Use-Case Model Name**: **Rate Trails**

- Initial Assumption: The trail that a User wants to hike exists in the system.
- Normal: A user is able to rate a trail that they have hiked in the past to help others know more about the trail.
- What Can Go Wrong: A user could create a fake review that might negatively affect the trail.
- Other Activities: N/A

- System State on Completion: A new review for the selected trail is attached to the trail

- **Use-Case Model Name**: **Become Guides**
- Initial Assumption: A General User has created an account.
- Normal: A General User fills out a form from the trail page which is sent to the park operator/s attached to the trail.
- What Can Go Wrong: The park operator could refuse to allow a user to become a Guide for that trail.
- Other Activities: N/A
- System State on Completion: A General User (after being accepted by the Park Operator) is elevated to a Guide for that trail.

- **Use-Case Model Name**: **Create Guided Hike**
- Initial Assumption: A User who want to create a Guided Hike has been given the role of a Guide
- Normal: A Trail Guide navigates to the trail where they want to set up a guided hike and clicks an option to start a Guided Hike
- What Can Go Wrong:
- Other Activities: A Trail Guide can close a Guided Hike if they decide that they can not join the hike.
- System State on Completion: A Guided Hike is publicly displayed on the trail's description page where General Users can opt to join.

3.3.3.2. Actor: Park Operator (Andrew Nice)

- **Use-Case Model Name: 1 on 1 chat**
- Initial Assumption: Park operators will be able to engage in a private chat with another user, park operator, or admin.
- Normal: They will be able to receive and send texts to other users. Their chats will be saved in their chat logs.
- What Can Go Wrong: N/A
- Other Activities: Switch to chat with other users.
- System State on Completion: Message is sent to the other user who is able to view and reply to the message they have received.

- **Use-Case Model Name: Sign-in/up**
- Initial Assumption: The park operator will be able to create an account and will see that they must submit a form in order to be approved as a park operator.
- Normal: Park operators will sign up by choosing they'd like to create an account as a park operator. They will sign up by creating an email and password. Upon creation they will have to fill out a google form that will be reviewed by an admin. If approved they

will be sent an email saying so, to which they will be able to log into WildFinder as a park operator.

- What Can Go Wrong: They forget their password and must reset it.
- Other Activities: N/A
- System State on Completion: The park operators account will be created and they'll be able to access the website.

- **Use-Case Model Name**: **Upload Trail**

- Initial Assumption: Park Operators are able to click on a button that lets them create a trail in a county of their choosing.
- Normal: The trail is created by entering the trail name, location, and images. Optionally they can add extra information about the trail.
- What Can Go Wrong: They upload something inappropriate.
- Other Activities: N/A
- System State on Completion: On the county page the new trail is added.

- **Use-Case Model Name: Update Trail**

- Initial Assumption: The park operator is able to modify the information on the trail.
- Normal: They'll click on the edit tool by the trail allowing them to edit the information about the trail.
- What Can Go Wrong:
- Other Activities: They can delete the trails.
- System State on Completion: The updated version of the trail will show in the specified counties page.

3.3.3.3. Actor: Web Admin (Jacob Greene)

- **Use-Case Model Name**: **Sign-in to admin account access admin page**

- Initial Assumption: User has credentials to input into username/password fields.
- Normal: Admin enters username/password.
- What Can Go Wrong: Admin can forget username/password. Unwanted users accessing admin accounts.
- Other Activities: N/A
- System State on Completion: User selects admin login from homepage sign-in, enters credentials and logs-in, admin is directed to the admin page that contains "tools".

- **Use-Case Model Name**: **Generate Reports**

- Initial Assumption: Admin account is logged into and currently on page of "tools".

- Normal: Select generate report and report type
- What Can Go Wrong: Admin could not understand how to generate reports.
- Other Activities: N/A
- System State on Completion: Report is printed to the browser window in designated area.

- **Use-Case Model Name**: **Moderate/edit all accounts**
- Initial Assumption: Logged into admin account and currently on page with admin "account tools".
- Normal: Admin enters page of user information with editing authorization, makes changes, saves.
- What Can Go Wrong: Wrong information is edited, changes are not saved, the page is too convoluted and packed with information to find the correct user accounts.
- Other Activities: Filter options?
- System State on Completion: Changes are saved and reflected in user information database.

- **Use-Case Model Name**: **Validate/Create park operator accounts**
- Initial Assumption: Logged into admin account and currently on page with "account tools".
- Normal: Admin receives notice of new application for park operator, admin validates information, admin creates new park operator account and enters all information and completes account set-up.
- What Can Go Wrong: Incorrect park operator information entered, account creation failure, park operators lose account login information.
- Other Activities: N/A
- System State on Completion: New park operator account is created.

4. Technical Requirements

4.1. Interface Requirements

4.1.1. User Interfaces

4.1.2. Hardware Interfaces

The web application will run on any hardware device that has access to the internet, can display web pages, and has the ability to interact with web pages. While this does include things such as smartphones and tablets, the website is designed to work primarily on laptops and pc's so the layout might be different for mobile devices.

4.1.3. Communications Interfaces

It must be able to connect to the internet as well as the local database which will be determined in detail at a later date. The communication protocol HTTP must be able to connect with the Leaflet API to return the map of Piedmont North Carolina.

4.1.4. Software Interfaces

We will use React and Spring Boot ThymeLeaf to help build the frontend, as well as JPA for the backend database functionality. We will also use Spring Boot with Java to connect the frontend to the backend.

5. Non-Functional Requirements

5.1. Performance Requirements

5.2. Safety Requirements

- NFR0(R): 1 on 1 chat should only be seen by those in the chat itself.
- NFR1(R): Passwords should only be able to be accessed to change and recover lost passwords.

5.3. Security Requirements

- NFR2(R): The system will only be usable by authorized users.

5.4. Software Quality Attributes

5.4.1. Availability: Application should be able to be accessed at any time.

5.4.2. Correctness: Details about the Trails should be correct, and false information about said parks should be corrected.

5.4.3. Maintainability: As more trails are added, the website should be able to efficiently retain and maintain all trails.

5.4.4. Reusability: The application should be able to be used multiple times by many people.

5.4.5. Portability: The website should be able to be accessed by any device connected to the internet with an appropriate browser.

5.5. Process Requirements

5.5.1. Development Process Used

Waterfall Model

5.5.2. Time Constraints
Initial Prototype: 2/27/2024
Improved SRS Document: 4/2/2024
Final Prototype: 4/30/2024
Final Documentation: 5/1/2024

5.5.3. Cost and Delivery Date
The cost will most likely be close to $422(rough cost of the class) and the final draft will be submitted by Apr 30, 2024.

5.6. Other Requirements
Continued direction from Project Manager about finishing the project (learning in class).

6. Design Documents
6.1. Software Architecture

6.2. High-Level Database Schema

6.3. Software Design
6.3.1. State Machine Diagram: General User or Trail Guide (Christian Wilson)
6.3.2. State Machine Diagram: Park Operator (Andrew Nice)
6.3.3. State Machine Diagram: Web Admin (Jacob Greene)

6.4. UML Class Diagram

7. Scenario
7.1. Brief Written Scenario with Screenshots