Software Engineering
Software Requirements Specification (SRS) Document

Weather to Wear

2/8/2024

v0.01

By: [Luke Leong, Jamin Bucur, Arturo D.]

**[My words and actions will reflect Academic Integrity.
I will not cheat or lie or steal in academic matters.
I will promote integrity in the UNCG community.
Luke Leong, Jamin Bucur, Arturo D 02/19/2024]**

Table of Contents

1. Introduction

1.1. Purpose

The primary goal for Weather to Wear is to provide weather and tempature based clothing recomendations. Secondary goals include being a social hub, where you can find and purchase new clothes. It will help non-morning people to worry about one less thing, by providing an accurate recomendation for clothes in a few clicks.

1.2. Document Conventions

The purpose of this Software Requirements Document (SRD) is to describe Weather to Wear's goals, implementation, and features. This document should cover what each user of the application can expect, and how it will be accomplished. In it we will outline the systems and features provided, give an idea of what users can expect to see and use, and detail the scope of the project.

1.3. Definitions, Acronyms, and Abbreviations

Java	A programming language originally developed by James Gosling at Sun Microsystems. We will be using this language to build the Restaurant Manager.
.HTML	Hypertext Markup Language. This is the code that will be used to structure and design the web application and its content.
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

This SRS document has multiple stakeholders; customers, advertisers, admins, and developers. All parties will want to read Section 2.2, User Class and Characteristics to understand what they will need to do in order to use the app.Customers will want to read through Product Features (Section 2.1) to understand the value of the product and features. Advertisers should read through Section 2.2, and all of Section 1, to be better informed of the product. Admins need to read what the previous two have to understand what content will be on the application. Developers should be familiar with the whole document, in order to understand the features, scope, and planning of the project.

1.5. Project Scope

The goal of this software is to be fast,simple, and accurate. To improve users experience and to reduce headaches using the software, these goals should be kept in mind duing development. Making an application people want to use will build an audience, which keeps users engaged, and allows for advertising to function properly.

The benefits of the project to business

- Reducing stress in the morning for customers, as they do not have to worry about what clothes to wear/bring for the day.
- Increase customer engagement with clothes as a form of social media, with the ability to share outfits for the day.
- By being useful to customers, customer engagement will go up, increasing the viewability of ads for advertisers.

1.6. Technology Challenges

Currently, our project is planning to use a Java program, with the console being used as the output. We are hoping to make this application light weight and efficient. The program will need to use an API to obtain the weather for the user's area. An issue that could occur is inaccurate information. For instance, if the API we choose to use is wrong in tempature, then our app will have no way of knowing. We will have to rely on another service for our own to work. We are planning to incorporate HTML into the project by using ThymeLeaf.

1.7. References- None Currently

2. General Description

2.1. Product Features

Weather to Wear (WW) should have features for customers, administrators, and advertisers. Customers should be able to create accounts, save closet information, join groups, and receive recommendations based on the weather. Admins should be able to feature and ban users/groups. Advertisers can put up ads for clothes they want to sell.

2.2. User Class and Characteristics

Currently our application needs any user to be able to input text through a keyboard. They would also need to have knowledge of the heat rating of their clothing. Prior knowledge of common websites would be useful, but not required.

2.3. Operating Environment

Currently the application is planned to use HTML to display frontend (ThymeLeaf). It is designed with computer use in mind (Windows, Mac, Linux systems) with use of an IDE, mobile devices are not currently planned for support.

2.4. Constraints

The program will only run on a computer capable of running the program on an IDE. The full scope of the project is hopefully realized, however the team has a deadline of a few weeks, which could lead to feature cuts. The program would have a challange scaling, as the current plan is to use a txt document to store the information.

2.5. Assumptions and Dependencies

We will be using the Java Language, with our program being dependent on Spring Web, and RestAPI to connect to external APIs and developed with Netbeans.We will be using OpenWeather API

(https://openweathermap.org/api) to get location's weather, and then return that information to the program. ThymeLeaf will be used to provide the frontend.

3. Functional Requirements

3.1. Primary

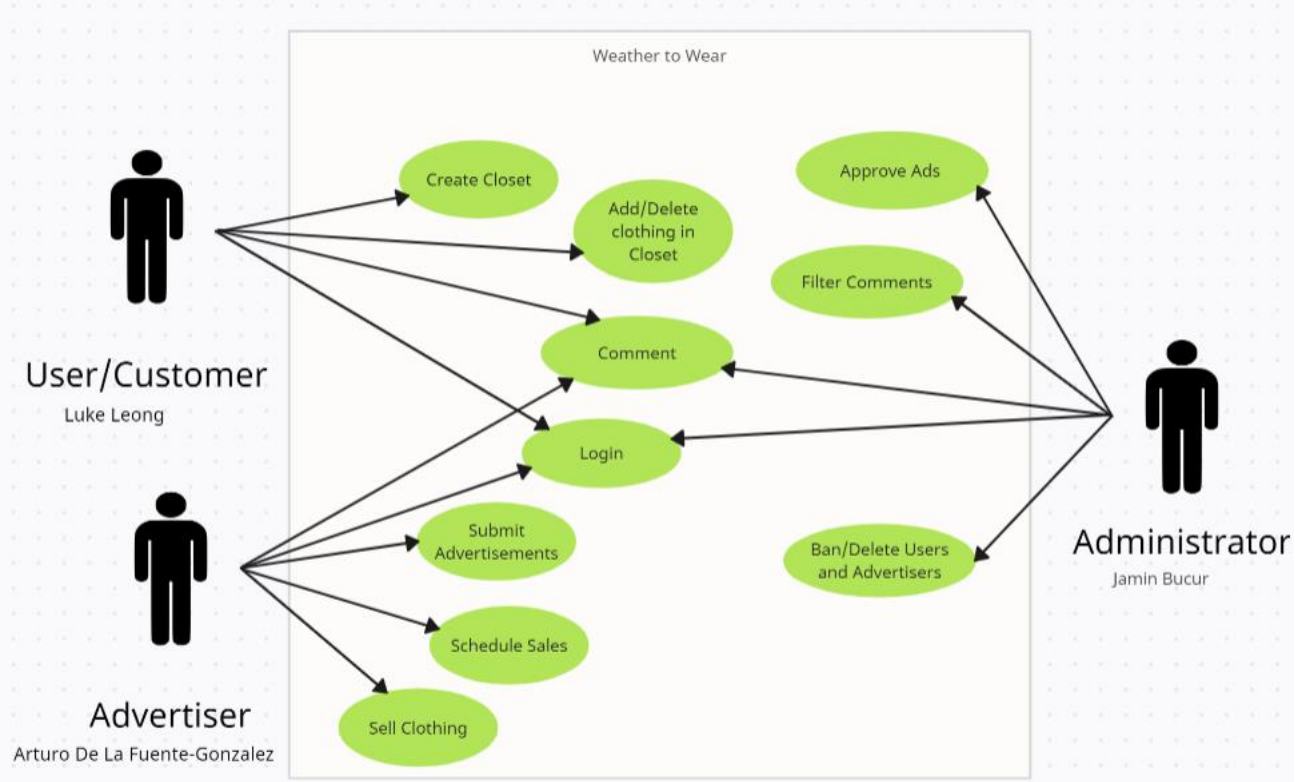
- FR0 (R): The system will allow users to create virtual closets for storing clothing items. Each closet will have a unique name provided by the user upon creation.
- FR1 (R): Users can add new clothing items to their virtual closets or delete existing ones as needed. The system will provide intuitive interfaces for managing closet contents.
- FR2 (R): Users must be able to log in to their accounts securely using authentication mechanisms. Additionally, users can comment on various items within the platform, enhancing user engagement and interaction.
- FR3 (R): Advertisers will be able to list clothing items for sale on the platform, including detailed descriptions, pricing information, and images. The system will support multiple listings per advertiser.
- FR4 (R): Advertisers can schedule sales or promotions for their listed clothing items, specifying sale durations and discount rates. The system will apply discounts automatically during the scheduled period.
- FR5 (R): Advertisers can submit advertisements to be displayed on the platform to promote their clothing items or brand. The system will review and approve submitted ads based on compliance with platform guidelines.
- FR6 (R): Administrators will have the authority to review and approve advertisements submitted by advertisers. They can also manage and moderate comments across the platform, ensuring compliance with community standards.

3.2. Secondary

- Password protection for information only accessible to employees, managers, and each individual table.
- Authorization so that comment editing, approval of ads, and banning is only accessable to administrators. Also, so advertisers are the only one allowed to schedule a sale.

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/Customer (Luke Leong)

- **Create Closet:** Users can create a virtual closet where they can organize and store their clothing items.
- **Add/Delete clothing in Closet:** Users can add new clothing items to their virtual closet or remove existing ones as needed.
- **Login & Comment:** Users can log in to their accounts and comment on various items.

3.3.2.2. Actor: Advertiser (Arturo De La Fuente-Gonzalez)

- **Sell Clothing**: Advertisers can list clothing items for sale on the platform, providing details and pricing.
- **Schedule Sales**: Advertisers can schedule specific sales or promotions for their listed clothing items.
- **Submit Advertisements:** Advertisers can submit advertisements to be displayed on the platform to promote their clothing items or brand.
- **Login & Comment:** Advertisers can log in to their administrator accounts and can comment on various items.

3.3.2.3. Actor: Administrator (Jamin Bucur)

- **Approve Ads:** Administrators can review and approve advertisements submitted by advertisers to ensure they comply with platform guidelines.
- **Filter Comments:** Administrators can manage and moderate comments across the platform, ensuring they meet community standards.

- **Ban/Delete Users and Advertisers:** Administrators have the authority to ban or delete users and advertisers who violate platform policies or engage in inappropriate behavior.
- **Login & Comment:** Administrators can log in to their administrator accounts and can comment on various items.

3.3.3. Use-Case Model Scenarios

3.3.3.1. Actor: User/Customer (Luke Leong)

- **Use-Case Name:** Log In
    - **Initial Assumption**: The user attempts to log into their account with a username and password
    - **Normal**: The user inputs their information and is able to log in
    - **What Can Go Wrong:** The system may have an error obtaining the account information
    - **Other Activities**: None
    - **System State on Completion**: The user will be logged into their account, and their current closet will be reflected.

- **Use-Case Name:** Create Closet
    - **Initial Assumption**: The user is logged into their account and wants to organize their clothing items.
    - **Normal**: The user selects the "Create Closet" option from the menu, inputs a name for the closet, and confirms to create it.
    - **What Can Go Wrong:** The system might experience a temporary glitch, preventing the creation of the closet. The user might input invalid characters for the closet name.
    - **Other Activities**: None
    - **System State on Completion**: The user has a newly created virtual closet ready for adding clothing items.
- **Use-Case Name**: Add/Delete Clothing in Closet
    - **Initial Assumption**: The user has already created a closet and wants to manage its contents.
    - **Normal**: The user navigates to their closet, selects the option to add or delete clothing, and follows the prompts to complete the action.
    - **What Can Go Wrong**: The system might fail to save changes due to a technical issue. The user might accidentally delete the wrong item.
    - **Other Activities**: None
    - **System State on Completion**: The user's closet reflects the changes made, with added or deleted clothing items.

3.3.3.2. Actor: Advertiser (Arturo De La Fuente-Gonzalez)

- **Use-Case Name**: Sell Clothing
    - **Initial Assumption**: The advertiser has registered an account on the platform and wants to list clothing items for sale.

- ∉ **Normal**: The advertiser accesses the "Sell Clothing" feature, uploads images and descriptions of the items, sets prices, and confirms the listings.
- ∉ **What Can Go Wrong**: Technical issues may prevent the upload of images or descriptions. The advertiser might input incorrect pricing information.
- ∉ **Other Activities**: None
- ∉ **System State on Completion**: The listed clothing items are available for purchase on the platform.

- **Use-Case Name**: Schedule Sales
  - ∉ **Initial Assumption**: The advertiser wants to plan promotional activities for their listed clothing items.
  - ∉ **Normal**: The advertiser accesses the "Schedule Sales" feature, selects the items for promotion, sets the sale duration and discount rates, and confirms the schedule.
  - ∉ **What Can Go Wrong**: The system might encounter errors in applying the discounts or scheduling the sales. The advertiser might mistakenly set incorrect sale durations.
  - ∉ **Other Activities**: None
  - ∉ **System State on Completion**: The scheduled sales are set to start and end at the specified times with the designated discounts applied.

3.3.3.3. Actor: Administrator (Jamin Bucur)

- **Use-Case Name**: Approve Ads
  - ∉ **Initial Assumption**: The administrator is logged into the system and needs to review submitted advertisements.
  - ∉ **Normal**: The administrator accesses the list of pending ads, reviews each one for compliance with guidelines, and approves or rejects them accordingly.
  - ∉ **What Can Go Wrong**: The administrator might overlook policy violations in an advertisement. Technical issues may disrupt the approval process.
  - ∉ **Other Activities**: None
  - ∉ **System State on Completion**: Approved advertisements are ready to be displayed on the platform.
- **Use-Case Name**: Filter Comments
  - ∉ **Initial Assumption**: The administrator wants to ensure a positive and respectful community atmosphere by moderating comments.
  - ∉ **Normal**: The administrator accesses the comment moderation tool and manages comments accordingly.
  - ∉ **What Can Go Wrong**: Might accidently flag legitimate comments as inappropriate. Technical issues could hinder the comment moderation process.
  - ∉ **Other Activities**: None
  - ∉ **System State on Completion**: The platform's comments section reflects the applied moderation actions, with inappropriate content filtered out.

4. Technical Requirements

4.1. Interface Requirements

4.1.1. User Interfaces

Dashboard/Home Screen: This screen will provide access to various features such as creating a closet, managing clothing items, selling clothing, scheduling sales, approving ads, filtering comments, and managing users and advertisers. It will display notifications and alerts relevant to the user's activities.

Closet Management Screen: This screen will allow users to view, add, and delete clothing items in their virtual closet. It will provide options to organize items and add details such as descriptions, tags, and images.

Advertisement Submission Screen: Advertisers will use this screen to submit advertisements. They will input ad details, upload images, and submit for review.

Advertisement Approval Screen: Administrators will review submitted advertisements on this screen. They will have options to approve, reject, or request modifications to ads.

Comment Moderation Screen: Administrators will use this screen to moderate comments. They will view comments, apply filters, and take actions such as approving, deleting, or flagging comments.

4.1.2. Hardware Interfaces

The hardware will run on any hardware device capable of running NetBeans IDE, displaying and interacting with web pages, and having access to the internet.

4.1.3. Communications Interfaces

It must be able to run a localhost server from Spring Boot Web and have the local txt database. The communication protocol, HTTP, must be able to connect to the OpenWeather API and return the weather temperature based on the location set.

4.1.4. Software Interfaces

We will use Spring Boot and ThymeLeaf to help build the frontend, as well as txt files 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

- NFR0(R): The Closet will consume less than 20 MB of memory
- NFR1(R): The system will consume less than 50MB of memory
- NFR2(R): The novice user will be able to add clothes and get resulting fit based on the weather the in less than 5 minutes.
- NFR3(R): The expert user will be able to add clothes and get resulting fit based on the weather the in less than 1 minute.

5.2. Safety Requirements

- NFR0(R): The system's error message should be clear and simple to let the user understand the misuse of the app. and prevent future similar mishaps/confusions.
- NFR1(Opt): The system will implement a timeout session to recongize when a user has been inactive for a long period of time

5.3. Security Requirements

- NFR4(R): The system will only be usable by authorized users.

5.4. Software Quality Attributes

5.4.1. Availability

- Scheduled Maintenance should be initialized during scheduled inactive hours to minimize conflict with user's using the app. .

5.4.2. Correctness

- All actions and processes should result in accurate outcomes to ensure and maintain the reliability and trustworthiness of the system's functionality.

5.4.3. Maintainability

- Code will be documented clearly and follow recommended coding standards/practices to manage facilitations

5.4.4. Reusability

- Common functionalities should be reusuable if it can be used in different parts of the system/code

5.4.5. Portability

- Program should be compatible with NetBeans and any other environments recently stated where it states it'll be deployed

5.5. Process Requirements

5.5.1. Development Process Used

- **Waterfall Model**

1. **Requirement Analysis:**
   Gather and document all requirements outlined in the Software Requirements Document (SRD), including user needs, system features, and business goals.
2. **Design Phase:**
   Create a detailed system architecture based on the outlined requirements, utilizing technologies such as Java, Spring Boot, and Thymeleaf.
3. **Implementation Phase:**
   Develop the backend functionality using Java with the Spring Boot framework, implementing features such as user authentication, closet management, advertisement listing, and comment moderation.
4. **Verification Phase:**
   Perform rigorous testing, including unit testing, integration testing, and system testing, to ensure the application functions as expected and meets the outlined requirements.
5. **Maintenance Phase:**

Provide ongoing support and maintenance, addressing any bugs, performance issues, or feature requests that arise post-deployment.

5.5.2. Time Constraints

- Deadline April 30

5.5.3. Cost and Delivery Date

- Costs: time, blood, sweat, and tears.
- Expected Delivery Date: April 30

5.6. Other Requirements- none

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