*Vermontaholic*

Requirements Document

Last Revised: September 23, 2020

Team: Alexander Cardosi, Erik, Silas Smith

# **Introduction**

The 251 Club enjoys exploring the different towns in Vermont. These people currently keep track of their visited towns in a paper journal. To better keep track of their visited towns, and to share comments of these towns, they need a web application. They also need a way to prove they visited all 251 towns in Vermont. This web application, called Vermontaholic, will be able to perform all of these use cases.

Vermontaholic is a system that allows visitors to manage their exploration of the towns of Vermont. It gives visitors a way to record information about each visit, in a journal-like manner, and share that information with others. Ultimately, Vermontaholic is intended to support members of the 251 Club of Vermont by connecting to the 251 Club’s backend database of users and recorded comments and photos. However, as a simplification, Vermontaholic will initially be designed for use by anyone and have no explicit connection with the 251 Club. Although the concept of “town bagging” is not unique to Vermont, Vermontaholic as currently imagined focuses on that one state, also as a simplification.

The rest of this document describes the detailed functional and nonfunctional requirements for Vermontaholic.

# **Functional Requirements**

The Vermontaholic system shall only support visitors who are visiting towns in the state of Vermont. Vermontaholic shall not attempt to interact with the existing 251 Club web site. However, see the Expected Extensions non-functional requirement. These limitations are likely to be lifted in the future.

## **Actors**

The following actors (roles) shall be supported:

1. **Visitor**. An individual (or a group, but treated as an individual by the system) who visits a town and wishes to record information about that visit (called the visit’s *artifacts* by this specification). This specification uses “visitor” instead of the more common term “user” to reflect the primary motivation of that actor.

2. **Observer** A random individual on the internet without authentication.

3. **Admin** A user who can remove inappropriate comments and images and ban and suspend users.

## **Use Cases**

|  |  |
| --- | --- |
| **ID:** | **Mark-visited** |
| **Title:** | Mark town as visited |
| **Description:** | A visitor marks a town as visited. They can optionally write a comment using the Write-Comment. |
| **Primary Actor:** | Visitor |
| **Preconditions:** | The Visitor must be authenticated to the system |
| **Postconditions:** | None |
| **Main**  **Success Scenario:** | · Visitor selects the town, default will be the current location determined by GPS  · Visitor selects the date/time default determined by current system time  · Visitor marks town as visited. They can decide here if they want to add a comment (Write-Comment) |
| **Extensions:** | None |
| **Frequency of Use:** | Frequently. |
| **Status:** | Not Implemented |
| **Owner:** | Yellow Team |
| **Priority:** | Very high. This is essential functionality. |

|  |  |
| --- | --- |
| **ID:** | **Write-Comment** |
| **Title:** | Create Comment Artifact |
| **Description:** | A visitor creates a comment related to one town that the visitor has visited. |
| **Primary Actor:** | Visitor |
| **Preconditions:** | The Visitor must be authenticated to the system |
| **Postconditions:** | The comment provided by the visitor has been recorded in the artifact database |
| **Main**  **Success Scenario:** | · Visitor selects the town, default will be the current location determined by GPS  · Visitor selects the date/time default determined by current system time  · Visitor writes/edits the comment  · Visitor explicitly saves comment |
| **Extensions:** | None |
| **Frequency of Use:** | Frequently. |
| **Status:** | Not Implemented |
| **Owner:** | Yellow Team |
| **Priority:** | Very high. This is essential functionality. |

|  |  |
| --- | --- |
| **ID:** | **View-Comments-For-Town** |
| **Title:** | View all of the comments (and pictures) for a specific town |
| **Description:** | A user clicks on a town in VT and sees a list of comments and images for that town from what other users have posted. |
| **Primary Actor:** | Observer |
| **Preconditions:** | None |
| **Postconditions:** | None |
| **Main**  **Success Scenario:** | · Observers sees a list of towns and clicks on one  · Observer sees information posted by users in that town. |
| **Extensions:** | None |
| **Frequency of Use:** | Frequently. |
| **Status:** | Not Implemented |
| **Owner:** | Yellow Team |
| **Priority:** | High. This is essential functionality. |

|  |  |
| --- | --- |
| **ID:** | **View-Visited-Towns** |
| **Title:** | View all the towns you have been to and posted comments/pictures for |
| **Description:** | A visitor views a list of all the towns they have posted comments/pictures about |
| **Primary Actor:** | Visitor |
| **Preconditions:** | The Visitor must be authenticated to the system. |
| **Postconditions:** | None? |
| **Main**  **Success Scenario:** | · Visitor clicks to see list of all the towns they have been too  · Can view all the towns you have been to |
| **Extensions:** | None |
| **Frequency of Use:** | Frequently. |
| **Status:** | Not Implemented |
| **Owner:** | Yellow Team |
| **Priority:** | Very high. This is essential functionality. |

|  |  |
| --- | --- |
| **ID:** | **Completed** |
| **Title:** | List of people who have completed |
| **Description:** | An observer views who has completed the challenge |
| **Primary Actor:** | Observers |
| **Preconditions:** | None |
| **Postconditions:** | None |
| **Main**  **Success Scenario:** | · Observer clicks on link to the page  · You can see the list |
| **Extensions:** | None |
| **Frequency of Use:** | Frequently. |
| **Status:** | Not Implemented |
| **Owner:** | Yellow Team |
| **Priority:** | Very high. This is essential functionality. |

# **Non-Functional Requirements**

## **Platform**

· **Platform-Architecture**. This program

· **Platform-Multiuser**. This program will support multiple users which connects to the same backend

· **Platform-Client**. The front-end client will be a mobile friendly web application with support for Chrome (and Chrome for Android) 85+ , Firefox (and Firefox for Android) 79+, Safari (and iOS Safari) 14+, and Edge 86+.

· **Platform-Backend**. Ubuntu 18.04.5 LTS

## **Performance**

2 cores 4 threads with 2.40GHz needed. 4 GB RAM.

## **Security**

Username and hashed password stored on the server. Posts and pictures are public. They are not encrypted and anyone can access them.

## **User Characteristics**

The users know how to use the computer, and smartphone

## **Scale**

The system must support a minimum of 40 users simultaneously.

## **Documentation**

Comments on code, intended users (of the comments) are only the developers.

## **Data Formats**

Users upload images in either of the three formats:

* JPG
* PNG
* GIF

Their posts will be stored in plaintext on the server.

## **Internationalization**

This program will only need to support English speaking people.

## **Expected Enhancements**

· **Enhancement-251-Backend**. Vermontaholic will likely need to be enhanced to (optionally) interact with the back-end databases of the 251 Club of Vermont. Specifically, it should eventually be possible for a visitor who is a member of the club to use his/her club credentials to log in and then upload artifacts to the club’s server (for presentation on the club web page), and also to share artifacts with, and view artifacts shared by other club members.

· **Enhancement-Multistate**. Vermontaholic will NOT be extended to support states other than Vermont.

## **Date**

The project should be completed by December 11th.