Software Requirements Specification

for

UMaine Connect

Version 0.1

Prepared by Andres Vargas, Ayan Tariq, Nicole Cortez, Noah Brooks, Sean Radel

Maineframe

2/17/2022

Table of Contents

Table of Contents **1**

Revision History **2**

1. Introduction **3**

1.1 Purpose 3
1.2 Document Conventions 3
1.3 Intended Audience and Reading Suggestions 3
1.4 Product Scope 3
1.5 References 3

2. Overall Description **4**

2.1 Product Perspective 4
2.2 Product Functions 4
2.3 User Classes and Characteristics 4
2.4 Operating Environment 4
2.5 Design and Implementation Constraints 5
2.6 User Documentation 5
2.7 Assumptions and Dependencies 5

3. External Interface Requirements **5**

3.1 User Interfaces 5-7
3.2 Hardware Interfaces 7
3.3 Software Interfaces 7
3.4 Communications Interfaces 8

4. System Features **8**

4.1 Seeing events on the map 8
4.2 Adding an event 8
4.3 Creating a profile 9
4.4 Friends list management 9
4.5 Viewing event history 9
4.6 Viewing popular events 10

5. Other Nonfunctional Requirements **10**

5.1 Performance Requirements
5.2 Safety Requirements 10
5.3 Security Requirements 10
5.4 Software Quality Attributes 11
5.5 Business Rules 11

Appendix A: Glossary **11**

Revision History

Name	Date	Reason For Changes	Version
Noah Brooks	2/20	Making initial changes to document	0.1
Andres Vargas	2/20	Making initial changes to document	0.1
Sean Radel	2/20	Making initial changes to document	0.1
Nicole Cortez	2/20	Making initial changes to document	0.1
Ayan Tariq	2/20	Making initial changes to document	0.1

1. Introduction

1.1 Purpose

1.1.1 The UMaine Connect software is a way for users around the UMaine campus to share and locate events in realtime and the future. This document currently outlines all of the information for UMaine Connect version 0.1.

1.2 Document Conventions

1.2.1 This document is currently the first draft of our system requirements specification document and currently does not employ any formatting that indicates significance.

1.3 Intended Audience and Reading Suggestions

1.3.1 This document is intended for users, project managers, developers and documentation writers. It is recommended that everyone first read this introduction. Users are recommended to then read the overall description section. Project managers are recommended to read the entire document from top to bottom how it is laid out. Developers should read the system features and non-functional requirements section. Documentation writers are recommended to read up until the non-functional requirements section.

1.4 Product Scope

1.4.1 The UMaine Connect software is a way for users around the UMaine campus to share and locate events in realtime and the future. The benefit to our software is that it is currently tailored to UMaine, however could easily be rolled out to other campuses. Our current goal is to develop a healthy ecosystem of users around the UMaine campus.

1.5 References

1.5.1 In this version of our system requirements specification document we do not yet have any references.

2. Overall Description

2.1 Product Perspective

2.1.1 We are creating this product in order to improve community involvement and make event planning more efficient.This product is a new self-contained product. This product is not a part of a larger system, but does implement the Google Maps API and Firebase.

2.2 Product Functions

2.2.1

- Implement a map interface
- Create an account
- Personalize an account
- Mark a location for an event
- Set time and description for events
- View attendees for events
- Sign up/attend events
- View Popular events
- Generate Shareable Links for events
- View personal event history
- Add/Remove Friends
- View Friends

2.3 User Classes and Characteristics

2.3.1 Developers are the first class of users. This is the team that is developing the application and responsible for its performance and persistence.
2.3.2 Event Attendees are the users that attend events that are created using our application.
2.3.3 Event Planners are the users that mark points on the map and specify events that are happening. These users provide information on the map to other users that a gathering is happening. All relevant information is provided on the point that is placed on the map.

2.4 Operating Environment

2.4.1 UMaine Connect will be hosted on a cloud hosted Ubuntu 20.04 lts server. UMaine Connect will be accessible over HTTPS on both desktop and mobile systems. UMaine Connect will feature two different user interfaces, one tailored for mobile screens and one tailored to larger desktop screens.

2.5 Design and Implementation Constraints

2.5.1 UMaine Connect is constrained by both Firebase and the Google Maps API. It must operate on a webpage and be written using HTML, CSS, and Javascript. React JS is a library that will be used for UI development. UMaine Connect is constrained by all data collection and privacy laws in the United States. Development must be done using a modified form of agile methodologies. The project is limited in time as a final presentation must be completed by May 1st, therefore we must be ready to release a final product by late April. Development will be performed according to the standards and guidelines set by IEEE. It is important to consider the fact that we need to encrypt our user's data and secure our databases.

2.6 User Documentation

2.6.1 UMaine Connect's front page will feature links to an "About" page that will work to be the central hub for necessary guides as well as troubleshooting techniques related to utilized technologies. This page will list information about the application, and extra information that may not be featured on the landing page.

2.7 Assumptions and Dependencies

2.7.1 UMaine Connect is heavily dependent on Google Maps API and Firebase. Associated costs with these services could constrain development. We are assuming that we will be able to use these services for free for the duration of this project. During the development phase of this project we will rely on Linode to host our web server. We are assuming we will get $100 credit for signing up and that the site will only cost $5 a month to run. We will be using a free domain name from Freenom while we are developing this project.

3. External Interface Requirements

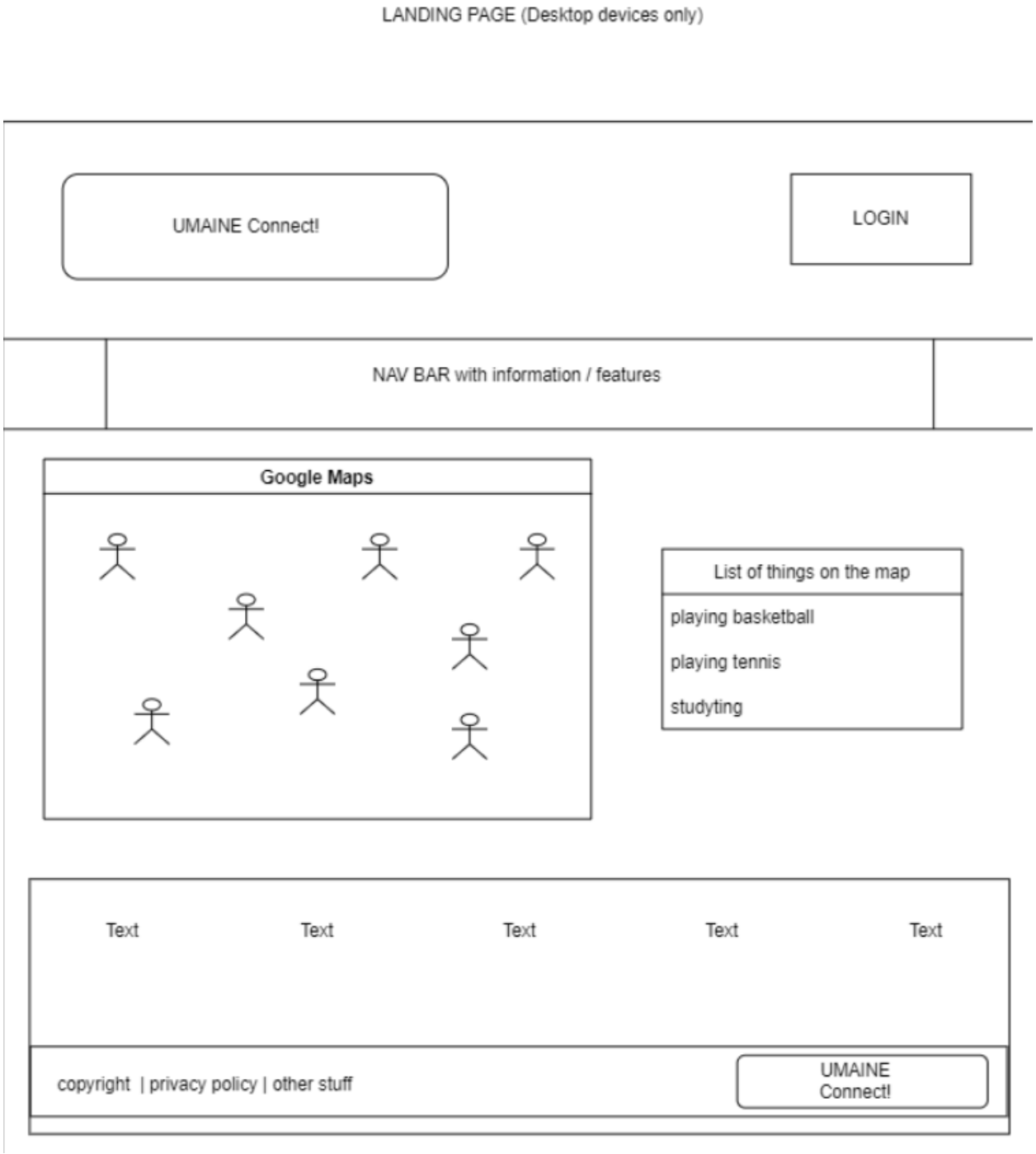
3.1 User Interface

3.1.1 Every page on the UMaine Connect website will feature a navigation bar that has links back to the landing page, as well as to all other relevant pages. Every single page will have links towards the bottom that direct to privacy policy and legality notices. All pages should link to account and login pages. All pages must have a UMaine Connect logo that acts as a link back

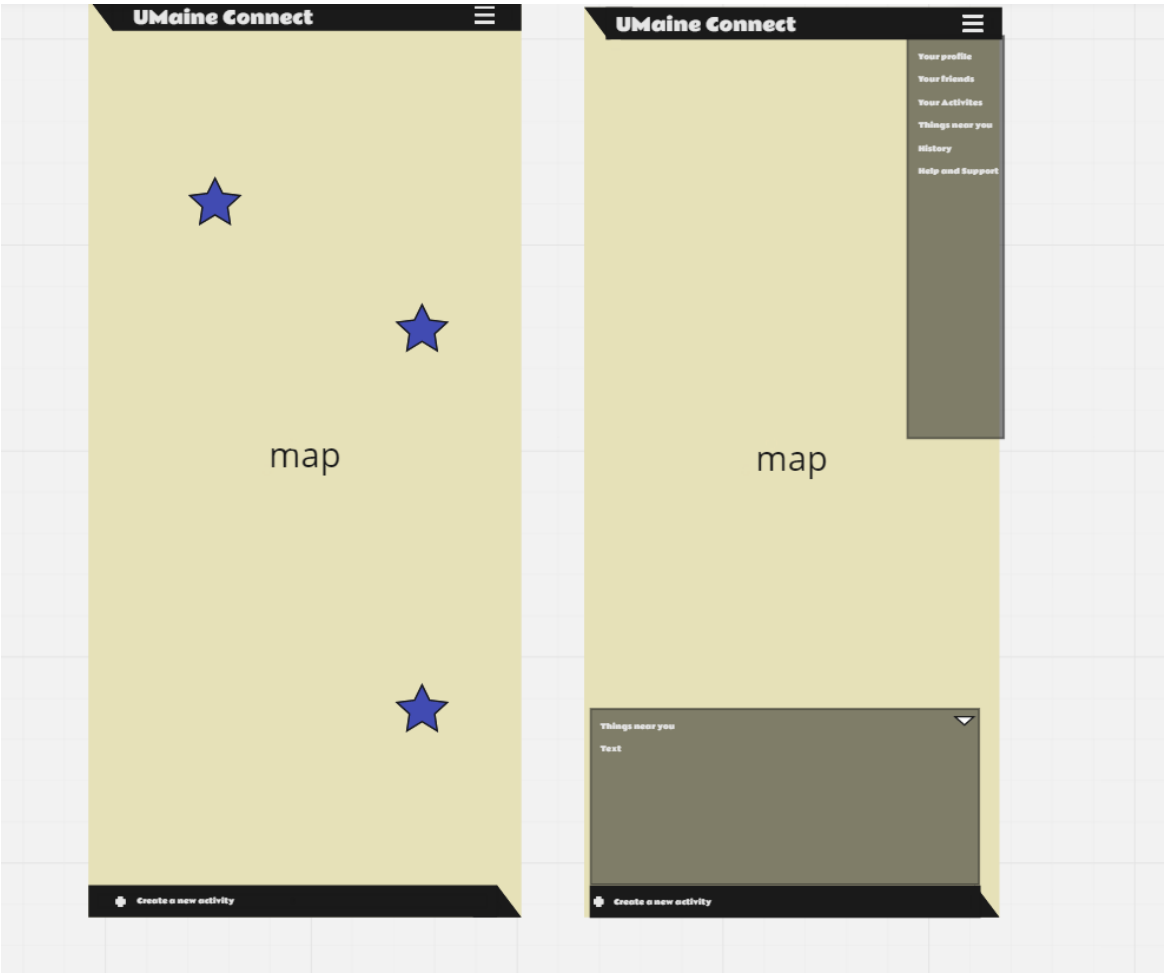
to the home page. The Google Maps API must be the largest graphic within the GUI. It shall be central and stand out among other design features.

Here are some wireframes of the webfront:

Desktop Screen:



Mobile Screen:



*Note, colors subject to change

3.2 Hardware Interfaces

3.2.1 UMaine Connect shall accept input from user keyboards and mice while on a desktop. If using UMaine Connect on mobile, it will accept touch. There will be no audio devices necessary to use this application.

3.3 Software Interfaces

3.3.1 UMaine Connect shall utilize Firebase, which is a NoSQL realtime database. Firebase will hold user and event information. UMaine Connect uses Google Maps API to provide a map interface that users will interact with to create and find events. A modern web browser that supports TLS 1.2 will be needed to load the UMaine Connect website.

3.4 Communications Interfaces

3.4.1 UMaine Connect will be accessible over HTTPS (port 443). Google Maps API and Firebase both also operate on port 443. Firebase which utilizes Google login authentication may notify users of new logins with their account on the service.

4. System Features

4.1 Seeing Events on the Map

4.1.1 Description and Priority

This feature will display a pin on the Google Maps interface where an event has been created. This data will be pulled from Firebase. (high priority)

4.1.2 Stimulus/Response Sequences

This feature requires no stimulus but should be available from the landing page.

The system will look at the events from the database and display them on the Google Maps UI. If an event is selected, a box will appear displaying the event description and attendees.

4.1.3 Functional Requirements

REQ-1: The integration of the Google Maps interface shall display a pin where events are.

REQ-2: The interface shall display the event description upon being clicked.

REQ-3: The interface shall display the number of attendees upon being clicked.

4.2 Adding an Event

4.2.1 This feature will allow the user to click a location on the map and add a pin on the Google Maps interface. They will then be able to set a description of the event in a text box. The user will also be able to generate a link to share their event.

4.2.2 This feature will respond to the stimulus of a click or tap from the user. The system will respond to the tap by placing a marker on the desired location and open up a text box that will allow the user to input alphanumeric characters and icons. Upon the user pressing the “share event” button, a link will show up in a popup window.

4.2.3 REQ-1: The system shall allow the user to place a marker where they tap.

REQ-2: The system shall allow the user to write a description for an event in a text box after placing a marker.

REQ-3: The system shall store event data in the Firebase database.

REQ-4:The system shall allow the user to generate a link to their event.

4.3 Creating a Profile

4.3.1 This feature shall allow the user to create a profile where they can add an image and display name on their profile.
4.3.2 This feature shall accept the stimulus of a click or tap on the sign up tab. The response would then be to open up a pop up window where the user can input their email address, username, and password twice for verification. The system shall respond by closing the popup after displaying a statement indicating success.
4.3.3 REQ-1: UMaine Connect will allow the user to create a profile.
REQ-2: The web interface will allow the user to customize their profile with an image.
REQ-3: The web interface will have a popup for profile creation upon the user clicking the sign up button.

4.4 Friends List Management

4.4.1 This feature shall allow the user to add friends by username to their friends list. The user will be able to view their friends' profiles and remove their friends from the list as well.
4.4.2 This feature will accept stimulus from the user in the form of a tap or a click. This will then respond in the system opening up a pop up in which the user can see their friends or click another button to add a friend. Removal of a friend will occur when a user clicks the "remove friend" button on their profile. The system will respond by removing the friend from the user's friend list.
4.4.3 REQ-1: The system shall allow the user to send friend requests to other users.
REQ-2: The system shall allow the user to remove users added as friends.
REQ-3: The system shall allow the user to view a list of their currently added friends.
REQ-4: The system shall allow the user to view friends' events.

4.5 Viewing Event History

4.5.1 This feature shall allow the user to see a list of all of the events they have attended on UMaine Connect. The user will be able to see the location, event name, and event description in a log format designed by the development team.
4.5.2 This feature will accept the user stimulus in the form of a click or a tap on the "history" button. The system will then respond with a popup that shows a log of events they have previously attended.
4.5.3 REQ-1: The web interface shall allow the user to view a list of the events they have previously attended.
REQ-2: The web interface shall allow the user to view the details of each previously attended event.

4.6 Viewing Popular Events

4.6.1: This feature shall allow the user to sort events by the amount of committed users.
4.6.2: This feature will accept stimulus in the form of a click or tap on the “sort” button above the list of events.
4.6.3: This feature will allow the user to sort by the latest events, oldest events, and lastly most and least popular events.
4.6.4: REQ-1: The application shall allow the user to sort by newest events.
REQ-2: The application shall allow the user to sort by oldest events.
REQ-3: The application shall allow the user to sort by least popular events.
REQ-4: The application shall allow the user to sort by most popular events.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

5.1.1 The system shall be able to handle 100 login requests at a time.
5.1.2 The system shall be able to handle 50 events on the map at a time.
5.1.3 The application shall list all events on the map in under 15 seconds.
5.1.4 The application shall load user data within 12 seconds.
5.1.5 The application response time should be instant.
5.1.6 The system shall handle 500 active users at a time.
5.1.7 The system shall let the user log in within 10 seconds.
5.1.8 The application shall show friend requests within 5 seconds.
5.1.9 The system shall load the map within 5 seconds.
5.1.10 The system shall be able to handle 100 new daily users.

5.2 Safety Requirements

5.2.1 The application shall not list events that are in hazardous areas 90% of the time.
5.2.2 The application shall not allow blocked users to join the blockers event.
5.2.3 The application shall require users to be over 16 years of age.

5.3 Security Requirements

5.3.1 The system shall defend against 95% of security threats.
5.3.2 The system shall protect the contents of the website from malicious attacks.
5.3.3 The system shall follow the existing privacy regulations in the U.S
5.3.4 The system shall protect the private information of the user 100% of the time.
5.3.5 The system shall defend against cyber attacks 99% of the time.
5.3.6 The system must defend against cross site scripting attacks.
5.3.7 The system must be resistant to SQL injection attacks.

5.3.8 The system must defend against DDoS attacks 50% of the time.
5.3.9 The system shall require users to use passwords that follow the following policy: All passwords must be at least 8 characters and must contain 1 capital letter, 1 digit, and 1 special character.

5.4 Software Quality Attributes

5.4.1 The application must remain available 90% of the time.
5.4.2 The application must display correct output 99% of the time.

5.5 Business Rules

5.5.1 The application shall allow the developers to remove events that break the terms of service.
5.5.2 The application shall allow the creator of an event to remove their event from the map.

Appendix A: Glossary

API: An Application Programming Interface (API) is a connection between computers or between computer programs. It is a type of software interface, offering a service to other pieces of software.

GUI: A Graphical User Interface (GUI) is a form of user interface that allows users to interface with electronic devices through graphical icons.

HTTPS: Hypertext Transfer Protocol Secure (HTTPS) is an extension of Hypertext Transfer Protocol (HTTP), it is used for secure communication of a computer network.

SQL: Structured Query Language (SQL) is a programming language designed for managing data held in a relational database management system.

DDoS: A Distributed Denial of Service (DDoS) attack is a cyber attack that seeks to make a machine or network unavailable by temporarily or indefinitely disrupting services of a host connected to a network.

TLS 1.2: Transport Layer Security (TLS) 1.2 is the current industry standard for transferring data over a network securely.