Software Requirement Specification Document: Date planner Application

By: Dominic Crofoot, Travis Early, Jordan Fok, Sean Hill, & Nicholas Ho

October 26, 2018

Revisions:

- **26Oct2018: First draft (by: Dominic, Jordan, Nicholas, Sean, and Travis)**
    - First draft of SRS document created.
- **8Nov2018: Revisions from customer adapted (by: Jordan Fok)**
    - Tables and figures are now labeled (table 1, table 2).
    - Multiple grammatical errors corrected. Mainly replacing personal pronouns.
    - Fixed formatting in section 4. No longer including EIR3, and deleted a row from table.
    - Completed a sentence section 1.3.
    - Reworded sections 1.3, 3 and its sub sections, and section 6 to make ideas clearer.
- **19Nov2018: Revisions from Professor Haque adapted (by: Jordan Fok)**
    - Changed title of 3.4 from “Attributes” to “Internal Interface Requirements
    - Made FR1 more specific in description.
    - Addressed sections 3.6 and 3.8 (maintainability and Security/privacy respectively)
    - Gave the document page numbers in the table of contents and throughout the document.

Table of Contents

1. Scope

1.1 Purpose -

This document supplies the software requirements for an unnamed date planning website intended for use by anyone who wants to go and do an activity. The date planner application will take in user preferences, treat them as filters, and return to the user various suggestions for dates based on the user's preferences.

1.2 Scope -

Our yet unnamed web app serves as an all-in-one location for people to find a location or event to spend time based on a variety of user supplied conditions.

1.3 Overview -

Section 1 provides the system overview, which includes details about the web app. It also includes the document overview, detailing what each section contains. Section 2 contains any documents that were referenced when making this document. Section 3 contains a stat machine that displays how users can use the app, and requirements for both functional and external interface. Section 4 explains the method in which requirements will be qualified. Section 5 explains how the requirements will be referenced throughout the other documents (SSDD, STD, etc.) Lastly, Section 6 are various notes that explain acronyms, reach goals, and general comments from the developers.

2. Referenced Documents

2.1. Government -

2.1.1. MIL-STD 498, 5 December 1994

2.1.1.1. Template for entire SRS document

2.1.1.2. Link: MIL-STD 498

2.2. Non-Government -

2.2.1. Loopback

2.2.1.1. A node.js framework used for back-end services. Loopback will be used by the application for user authentication and retrieving of the users filter history.

2.2.1.2. Link: https://loopback.io/

2.2.2. Google Maps API

2.2.2.1. Data supplied by Google Maps APi that will be used for filter, results tab, and map.

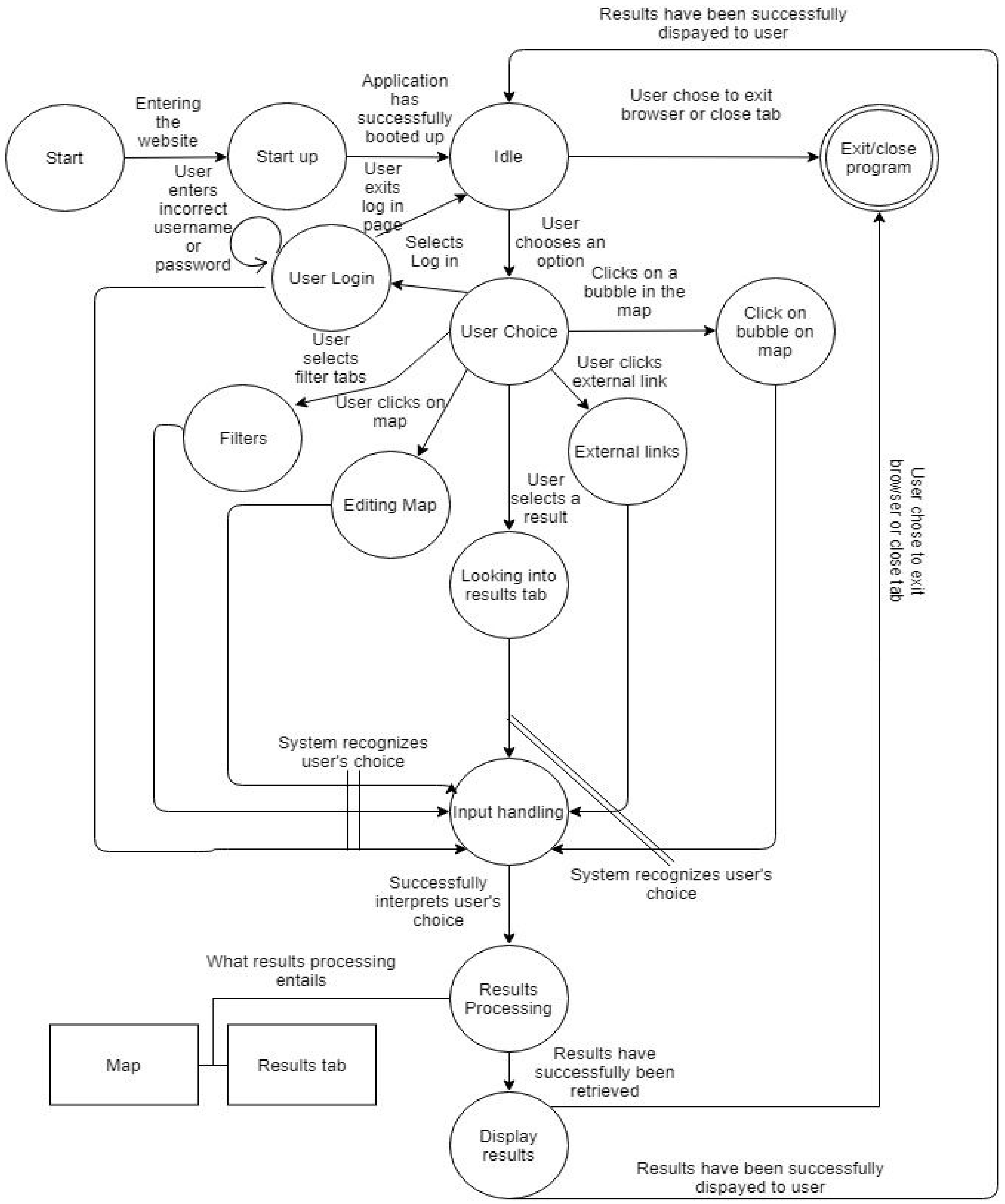
2.2.2.2. Link: https://cloud.google.com/maps-platform/

3. Requirements

3.1. Required States and Modes -

3.1.1. Table 1 (below) displays how the flow of the program from startup, to moving about the program, and exiting the program.

Table1


Results have been successfully dispayed to user
Start
Entering the website
Start up
Application has successfully booted up
Idle
User chose to exit browser or close tab
Exit/close program
User enters incorrect username or password
User exits log in page
User Login
Selects Log in
User chooses an option
Clicks on a bubble in the map
User Choice
Click on bubble on map
User selects filter tabs
User clicks external link
Filters
User clicks on map
External links
Editing Map
User selects a result
Looking into results tab
User chose to exit browser or close tab
System recognizes user's choice
Input handling
System recognizes user's choice
Successfully interprets user's choice
What results processing entails
Results Processing
Map
Results tab
Results have successfully been retrieved
Display results
Results have been successfully dispayed to user

3.2. Capability Requirements -

Capability requirements will be expressed in terms of their description, their required input to the software, the processing upon that input, and the output the software will produce as a result of the processing.

Functional Requirement 1 (FR1):

The application shall provide users with a limited number of results from Google Maps API that fit the filters given by the user

Input: A combination of predetermined filters selected by the user.
Processing: Uses the input to create a search query within the Google Maps API to develop a table of results that align with the given input.
Output: A list of activities that satisfy the input filters.

Functional Requirement 2 (FR2):

The application shall display and populate a map with pins denoting the location of activities returned by the software.

Input: A combination of predetermined filters selected by the user.
Processing: Uses the input to create a search query within the Google API to develop a map with pins of the results that align with the given input.
Output: A Google Maps representation of the results of processing the user input filters through the Google Maps API.

Functional Requirement 3 (FR3):

The application shall store the search history of users who are logged into an account. If a user wants a previous result or set of results, this feature allows them to do so

Input: A combination of predetermined filters selected by the user.
Processing: Uses SQL procedures to store the values given by the user into an SQL Database located on the UMBC servers.
Output: N/A

3.3. External Interface Requirements -

External Interface Requirement 1 (EIR1):

The application shall query the Google Maps API upon a set of given filters to develop results that align with the filters.

Input: A combination of predetermined filters selected by the user.
Processing: Uses the input to create a search query within the Google Maps API to develop a table of results that align with the given input.
Output: A list of activities that satisfy the input filters.

External Interface Requirement 2 (EIR2):

The application shall utilize the IBM Loopback framework to develop user accounts and store data using MySQL.

Input: A username and password, or a token.
Processing: This will be sent to our REST services that interfaces with our MySQL database and query the user's information with the provided input.
Output: profile information and a list of the user's filter history.

3.4. Internal Interface Requirements-

Availability

The application will be available through the public internet at all times.

Security

The provided security will be a token based system, which will identify a user and retrieve data based off the token.

Maintainability

The services are scoped to a limited amount. By limiting the services, adjustments to the system and adding new features will be easier.

3.5. Internal Data Requirements -

The application must maintain the data related to the user. Since the application interfaces with the Google Maps API, it will not be maintaining the search result data. Instead the design revolves around reducing the load of data stored. The application will be maintaining the users profile information, and a filter history of the users searches. The system revolves around the user retrieving their own information. So a transaction should follow all the properties of a database transaction (Atomicity, Consistency, Isolation, Durability).

3.6. Adaptation Requirements

3.6.1. The system will be compatible with the latest version of Google Chrome and Mozilla FireFox. The date planner will be updated as newer iterations of the browsers are released.

3.7. N/A

3.8. Privacy and Security

3.8.1. The following information will be saved by the app: username, password, and filters from a time of use. The application will not require emails and it will not save choice of results. This does not pose as a serious threat to attackers because the information they could get their hands on are just as accessible through various social medias. Even though the user's data will be properly encrypted, the data that is at risk has no use to anyone.

3.9. CSCI Environment Requirements

The CSCI must run on the two most used currently supported web browsers, Google Chrome version 70.0.3538.110 / November 19, 2018, and Mozilla FireFox 63.0.3 / November 15, 2018.

3.10. Computer Resource Requirements

3.10.1. Computer Hardware Requirements

The CSCI can be run on any computer capable of running the web browsers detailed in section 10.9 as well an internet connection.

3.10.2. Computer Hardware Resource Utilization Requirements

The CSCI is bound by the hardware resource utilization requirements of the web browsers detailed in 3.9.

3.10.3. Computer Software Requirements

The CSCI requires being connected to a MYSQL database server (i.e the browser).

3.10.4. Computer Communications Requirements

All computer communications will be handled through the web browsers detailed in section 3.9.

3.11. Software Quality Factors

3.11.1. All generated results lists shall be created with the most up-to-date information from Google Maps API. As Google Maps API's info updates, the date planner will show the most up to date information/results.

3.11.2. The CSCI will be available for use at any time.

3.12. Design and Implementation Constraints

The CSCI is limited by the Google Maps API. It will not be able to manually add search results that are not already available through Google. Conversely, as the Google Maps API improves, the search functionality will provide better results. The development is limited by the number of languages supported by Google Maps API.

4. Qualification Provisions - Table 2 displays what technique for qualification each requirement in section 3 will undergo.

Table 2:	**Demonstration**	**Test**	**Analysis**	**Inspection**	**Special Qualification methods**
FR1	✔				
FR2	✔				
FR3					
EIR1					
EIR2					✔

5. Requirements Traceability

5.1. This section addresses how various requirements will be tracked throughout the rest of the other documents (Development, testing, etc.) and a simplified explanation of the functional requirements.

FR1 (provide variety of dates given user input): The application shall display the same results in the list as are displayed on the map. Every result shall have a map pin associated with it, and vice versa.

FR2 (populate a map with pins and locations): Same as FR1.

FR3 (search history for logged in users): Logged-in users shall be able to see the same filters they have used previously and shall be able to generate results lists from their previous searches.

6. Notes

6.1. Acronyms:

6.1.1.	CSCI	Computer Software Configuration Item
6.1.2.	API	Application Program Interface
6.1.3.	MIL-STD	Military Standard
6.1.4.	N/A	Not Applicable
6.1.5.	REST	Representational State Transfer
6.1.6.	TBD	To Be Determined

6.2. Reach Goals:

6.2.1. At the moment, this application is not meant for financial gain, but it has been considered for the future in the form of commercials endorsement or sponsorships.

6.2.2. A feature not currently implemented is a timetable/list a users can use to create an itinerary for an entire day. The idea is the user could insert a search result into a list sectioned out into 24 hour blocks. The list could then be printed, shared, or even be saved into the user profile. Currently for the scope of the project, this is not currently feasible.

6.3. Reasons for Sections marked N/A:

6.3.1. 3.6 - No extra data will be provided by the system,

6.3.2. 3.7 - No realworld damage will be caused by the application,

6.3.3. 3.8 - Although the app has user profile capabilities, until the use of sponsorships is implemented, there is no reason to have safe guards. The login will only be a username and password with no other personal information attached.

6.4. Although a date planner has already been designed, the team's research found "Ways to ask to go on a date", "How to ask someone on a date", and a date planner, but it required personal information in the form of an email to be sent to the company. After some time, the customer would receive back results based on the personal questionnaire. The team's date planner acts as a faster, more convenient way of doing this.

6.5. User profiles is only meant to save searches at the moment. No other personal information will be required by the user besides a username and a password.