**Software Requirements Specification**

**for**

**TrailBlazer**

**Version 0**

**Prepared by Blake Hunt**

**04.20.25**

**Table of Contents**

Table of Contents

Revision History

1. Introduction
   1. Purpose
   2. Document Conventions
   3. Intended Audience and Reading Suggestions
   4. Project Scope
   5. References
2. Overall Description

2.1 Product Perspective

2.2 Product Features

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

1. System Features

3.1 System Feature 1

3.2 System Feature 2

1. External Interface Requirements

4.1 User Interfaces

4.2 Hardware Interfaces

4.3 Software Interfaces

4.4 Communications Interfaces

1. Other Nonfunctional Requirements

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

1. Other Requirements

Appendix A: Glossary

Appendix B: Analysis Models

Appendix C: Issues List

**Revision History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | | Date | Reason For Changes | Version |
| Swapped Language | 04.17.25 | | Development issues with the team behind Tk, community patch doesn’t work. | n/a |
| Switched to Winforms | 04.24.25 | | Running out of time to figure out other languages | 0.1 |

**1. Introduction**

* 1. Purpose

TrailBlazer is designed to allow people to plan out hiking trips and other outdoor activities. This is meant to replace maps, and GPS with one program.

* 1. Intended Audience and Reading Suggestions

This program is intended to be used by the general public for various purposes.

* 1. Project Scope

Light hikers could use it to plan short hikes. All types of commuters could use it as an offline replacement to GPS. As well as these more typical uses, it could be used in disaster situations to evacuate given they pre-plan the evacuation route.

**2. Overall Description**

2.1 Production Perspective

TrailBlazer is the first version of this program.

2.2 Product Features

Users will be able to plan A to B to C style trips offline. Multiple trips can be saved via a calendar. There will be a place to save details relating to individual trips.

2.3 Operating Environment

TrailBlazer can work in both Windows and some Linux OS’s. It should work on standard desktop and laptop computers, as well as various micro computers.

2.4 Design and Implementation Constraints

The program must work on windows systems, as well as Debian, and Ubuntu based Linux OS’s. The program should be able to run on the lighter hardware of a microcomputer (dual core CPU, 1GB of RAM). There should be some security features to protect people from being followed using data from the app.

2.5 User Documentation

TrailBlazer is meant to be used on whatever computer that the user downloads it to, some online documentation based to downloading and installing the program on various Operating Systems would be helpful. At some point a user manual will have to be created. An optional into to the app could be added.

2.6 Assumptions and Dependencies

The program will have to be tested on a microcomputer which will have to be self sourced.

**3. System Feature 1 (template)**

3.1 System Features

3.1.1 Description and Priority

3.1.2 Stimulus / Response Sequences

3.1.3 Functional Requirements

3.2 System Feature 2

**4. External Interface Requirements**

4.1 User Interfaces

The program will open up with a password screen, once this is logged into users will be shown the calendar screen, showing all saved planned trips. There will be a button to add new trips leading to a trip creation screen. Once this is finished you will be taken back to the calendar screen with a new trip planned. Clicking on a trip will show the user the route of the trip, along with a list of readable directions. Users should have full CRUD control of their trips.

4.2 Hardware Interfaces

I need to look into this more.

4.3 Software Interfaces

To be determined.

4.4 Communication Interfaces

TrailBlazer is meant to be usable entirely offline, however using available data to route paths, and get changes in infrastructure will be needed to create a product that people will want to use over other options. This might become a problem when working out security.

**5. Other Nonfunctional Requirements**

5.1 Performance requirements

The program should work on computers with a low grade dual core CPU, and 1GB of RAM.

5.2 Safety Requirements

Ignoring security features could leave the door open for people to break into the apps data and find out the planned location of users. Therefore we will have to create some sort of encryption behind the password screen.

5.3 Security requirements

Need to look into encryption before I can understand the necessary requirements.

5.4 Software Quality Attributes

**6. Other Requirements**

Appendix A: Glossary

Appendix B: Analysis Models

Appendix C: Issues List

Template made by someone else

Names Mentioned:

Norm Kerth

Karl Wiegers