5.0 Requirements Specification

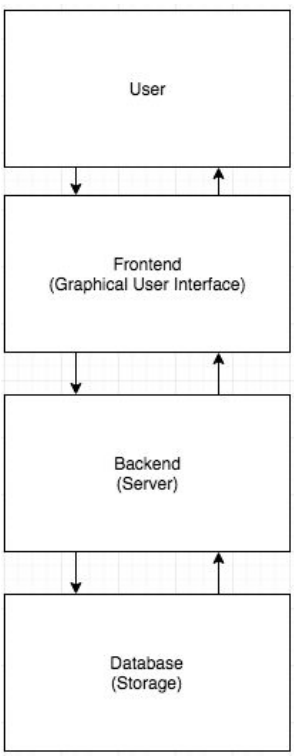
Outline of Requirements Specification

5.1 Introduction
5.2 Functional Requirements
5.3 Performance Requirements
5.4 Environment Requirements
5.4.1 Development Environment Requirements
5.4.2 Execution Environment Requirements

5.1 Introduction

This Software Requirements Specification (SRS) documents the requirements for the Trailze mobile phone application. Trailze is an open source iOS application that displays all of the trails in California. Trailze has two main functions: live trail navigation and social network integration.

The Trailze system architecture is comprised of a smartphone-based user interface (frontend), a server (backend), and a database. The frontend consists of maps, buttons, icons, etc. which allows users to accomplish the two main functions of Trailze. The server backend and database will be designed to store detailed trail information as well as trails users have already been to or plan to go to. Below is a high-level UML diagram of the system components of the application.



5.2 Functional Requirements

5.2.1 The Graphical User Interface (GUI) shall have a login page.
5.2.1.1 The login page shall provide spaces for users to input their username and password.
5.2.1.2 The login page shall have a button for new users to create an account.
5.2.1.3 The login page shall have a button for users who forgot their password.
5.2.1.4 The login page shall have a button for users who forgot their username.
5.2.1.5 The users shall be redirected to the main map page once credentials are verified.
5.2.2 The GUI shall have a map page that is based on the user's current location.
5.2.2.1 The trails shall be searchable.
5.2.2.2 The map shall update to show trail directions once one has been chosen.
5.2.2.3 The map shall provide information on the distance of the trail.
5.2.2.4 The map shall be downloadable, to help the user navigate the trails when network connectivity is lost.
5.2.2.5 The map shall continue providing live navigation once network connection has been restored.
5.2.3 The GUI shall include a list of trails the user has already been to.
5.2.3.1 This list of trails shall be searchable.
5.2.3.2 This list of trails shall update once another trail has been accomplished.
5.2.3.3 This list of trails shall provide information on the total distance and elevation gains of the trails.
5.2.3.4 This list of trails shall provide information on the average difficulty level of all the trails.
5.2.4 The GUI shall include a list of trails the user plans to go to (future trails).
5.2.4.1 This list of trails shall be searchable.
5.2.4.2 This list of trails shall update once a trail has been accomplished.
5.2.4.3 This list of trails shall update once a new trail has been added.
5.2.4.4 This list of trails shall provide information on the total distance and elevation gains of the trails.
5.2.4.5 This list of trails shall provide information on the average difficulty level of all the trails.
5.2.5 The GUI shall include a view for a specific trail.
5.2.5.1 The view shall provide the distance of the trail.
5.2.5.2 The view shall provide the elevation gain of the trail.
5.2.5.3 The view shall provide the difficulty level of the trail.
5.2.5.4 The view shall have a button to add the trail to either the future trails or the accomplished trails.
5.2.5.5 The view shall provide the average time to accomplish the trail.
5.2.6 The GUI shall provide a menu button.
5.2.6.1 The menu button shall provide a pop up of all the available button functions, which are categorized with specific icons.
5.2.6.2 The pop up shall have a button to report if a trail is closed.

5.2.6.3 The pop up shall have a button to report wildlife.
5.2.6.4 The pop up shall have a button to report weather conditions.
5.2.6.5 The pop up shall have a button to report trail conditions.
5.2.6.6 The pop up shall have a button to report turns that are confusing and/or difficult to find.
5.2.6.7 The map shall update to display these icons once these reports have been disclosed.
5.2.7 The server shall populate the database with trails the user has accomplished.
5.2.8 The server shall populate the database with trails the user plans to go to.
5.2.9 The server shall populate the database with user information.
5.2.10 The server shall perform regular analysis of reports to update the map accordingly.

5.3 Performance Requirements

5.3.1 User Login should take no more than 5 seconds.
After inputting login credentials, and successfully passing verification, a user should not wait more than 5 seconds before they are redirected to the main map page.

5.3.2 Fast Navigation
Users should not experience delays while navigating between maps and buttons.

5.3.3 Quick Search Query Results
Users should see results of a search within seconds of initiating the search.

5.3.4 Accessibility
The system will meet the minimum requirements for software accessibility.

5.3.5 No Reception
Users should be able to download the map of the trails to use incase they lose service and are no longer able to use live navigation.

5.3.6 Fast Reporting
Users should be able to quickly report problems with and species found on the trails.

5.4 Environment Requirements

5.4.1 Development Environment Requirements

Category	Requirement
Frontend	Swift
Server	Node.js, npm, Mongoose
Database	MongoDB

5.4.2 Execution Environment Requirements

Category	Requirement
Frontend	iOS Device
Server	Cloud-hosted server
Database	Cloud-hosted database