Software Requirements Specification

For

Rescue Reminders

Prepared by:
Chloe Drazen: 604257550
Joey Gomez-Benito: 904140064
Jake Moghtader: 304140934
David Scheibe: 004124290
Christian Youngers: 304166778

CS130 UCLA

4/15/2016

Table of Contents

1. Introduction

1.1 Purpose

The product whose requirements are specified in this document, Rescue Reminders, is an application that enables the rescue of stranded marine animals. Rescue Reminders allows observers to easily record information about the animal and share it with the appropriate authorities. Witnesses must be able to photograph the animal, and send the photo along with their current location to the authorities. Due to the disheveled current state of rescue organizations, the reason for this application is to enable rescuers to find the specific location on the beach of a stranded animal. Rescue Reminders must consist of a website where the passerby can submit information about the animal, and an application that receives push notifications of these reports for the authorities to respond.

1.2 Intended Audience and Reading Suggestions

This document is intended for a variety of readers, including developers, project managers, users of the application (animal rescuers), as well as employees of the aquarium who are sponsoring Rescue Reminders. Developers must work on the implementation of the website and application; project managers must ensure that the application is being developed as agreed upon; and animal rescuers must confirm that the application contains all the information they must need to successfully rescue the animal. Rescue Reminders is broken up into a website intended to be used by everyone who finds a stranded sea animal and an application intended to be used by rescuers. It is suggested that if a reader of this document is focusing on one side of Rescue Reminders that they read the corresponding part.

1.3 Product Scope

There are multiple components to Rescue Reminders including:

- Website: The website must allow people who find stranded sea animals on the beach to contact the authorities and provide relevant information.

    - Benefits: The website must streamline the process of supplying information to rescuers by standardizing the format in which data is submitted and by providing precise locations for the rescuers to arrive.
- Application: The application must allow rescuers to look up past rescue missions as well as be alerted immediately when an animal is in need of rescue at a nearby location. Rescuers can then indicate that they are in the process of helping the animal.
    - Benefits: All nearby rescuers must have immediate access to all of the data provided by the website, and must be notified immediately when the animal is reported. The information reported is mobily accessible and can be used on cell phones to arrive at the precise location of the animal.
- Database: The database must store all past and current animal rescue reports.
    - Benefits: Previously, there was no one place to access all of the relevant data. This database provides centralized information about past and present animal rescues in a variety of locations that can be analyzed to recognize trends.

2. Overall Description

2.1 Product Perspective

Rescue Reminders is a new system that focuses on targeting the inability for current rescue teams to quickly and accurately respond to cases of stranded marine mammals on the beaches of California. The system must utilize both an iOS and web application in order to allow common beachgoers to upload case information for stranded marine mammals to the correct rescue teams, thus improving response times and ultimately saving the lives of many more marine mammals.

2.2 Product Functions

Web Functions

- A mobile optimized website to enable users to easily and quickly create and upload stranded marine mammal cases
- The website must contain a page to view currently active and recently closed cases via Google Maps
- The website must contain a simple form to upload pictures, location data, and marine mammal information to the database

- The website must submit case information to the appropriate rescue authority within a few minutes to ensure a speedy rescue

iOS Functions

- The iOS application must contain a main page viewable by rescuers with a list of currently active cases
- Each active case must have a separate page, reachable by clicking on a case from the main page, that contains all necessary information (location, marine mammal data, etc.) for rescue teams to decide and initiate response efforts
- Each case page must have Google Maps integration in order to quickly guide rescue teams to the location of stranded marine mammals
- Push notifications to rescue team user's mobile phones

Database Functions

- The database must contain location data, picture data, and marine mammal data that must be displayed to both the iOS and Web application

2.3 User Classes and Characteristics

Rescue Team Class

The rescue team class must be the system's iOS user class. This class has a log in to the Rescue Reminders iOS application and view pertinent information related to all marine mammal cases within their regional location. This user class must have defined locational parameters that limit its viewing capabilities to within a 30 mile radius along the California coast of the rescue team's current location. All marine mammal cases outside of this location radius must not be viewable by members of the rescue team class within this region.

Common Beachgoer Class

The common beachgoer class must be one of two classes that are integrated within the Web application. This user class focuses on the input of form data into the database via the website's main page form. This user class must have the ability to submit text and image data into the system database, as well as view the Google Maps page that shows current marine mammal cases throughout the California coast.

Administrator Class

The administrator class is the second class that is integrated within the Web application. This class's main focus is the updating of system information pertaining to rescue teams across the California coast. Users of this class are members of rescue teams across California and must have access to update information pertaining to their rescue team, such as phone, address, and location data as well as update personnel information for each rescue team.

2.4 Operating Environment

Rescue Reminders is a complex system with iOS, Web, and Database integration. For the iOS application, we must utilize Apple's current iOS 9 which enables all users (rescue team members) with an Apple iPhone updated to Apple's current operating to access the Rescue Reminders application. The web application must be running various environments and software programs, detailed further in Section 3.3, including a REST API which allows for communication between the database and web server. The iOS application must also be integrated with the same REST API in order accomplish the task of connecting the web server to the iOS application. In the background, there must be an AWS host server which hosts MySQL and run Node.js in order to correctly serve files and data to and from the system components.

2.5 Design and Implementation Constraints

- Design constraints include the integration of a single MySQL database with both a web and iOS application. This limits our design to a singular solution that can handle both of these types of requests and correctly serve data to/from the system users.
- The system must be a real-time system constantly serving information to the rescue teams looking for stranded marine mammal cases.
- Utilizing iOS design features and mobile internet data, the rescue team users must be able to quickly check for updates to the system and receive push notifications to their devices.
- Administrators must continually update team member information and maintain organizational information with regards to their own rescue team
- The database must be constrained to an Amazon Web Services server in order to host our MySQL database
- The web server must be able to communicate to rescue organizations via multiple methods, including email, text, and possibly phone

2.6 Assumptions and Dependencies

- Rescue Reminders relies on the fact that multiple marine mammal rescue organizations must provide us with the correct contact information
- This system also depends on the ability of rescue organizations to have a means of receiving case information within their own system, or the rescue organizations must only be able to view information through our iOS and Web application rather than through their own systems.

3. External Interface Requirements

3.1 User Interfaces

Website:

The web portion of the application must consist of a main page with a form for the user to fill out and another page with a map that must show active cases near the user. The main form must be a simple mobile optimized page with a description of the purpose of the form, checkable buttons, a button to upload an image, and a button to send the data to the database. The top of the page must be a header bar with the name of the application and a logo.

iOS App:

The iOS app must feature two main screens, a list of open cases that must show a picture, the status of the animal, and how long the case has been active for. Clicking on an open case must take the app to the next screen, which must show a map with the location of the animal, a button to change the response status, a button to get directions to the animal, and other information such as a description of the situation and a picture of the animal.

3.2 Hardware Interfaces

Website:

The only hardware interfaces needed to interact with the web application are a keyboard and mouse.

iOS App:

The user must use the touch sensors on an iPhone to interact with the iOS application.

3.3 Software Interfaces

Website:

The web application must be hosted on an AWS server with a MySQL database installed and Node.js running to serve files and run a REST API. The frontend must be

built using HTML, CSS, Javascript, jQuery, Bootstrap to help with mobile responsiveness, and Angular.js to follow an MVC format.

iOS App:
The iPhone app must be a native iOS application which interfaces with the back end through the same REST API.

3.4 Communications Interfaces

Both the website and the iOS app must use HTTP to communicate with the backend.

4. System Features

4.1 Rescue Team Use Case

4.1.1 Description and Priority

A rescuer needs to be able to view reports in the iOS app in order to take action. In addition, they must be able to update the report with statuses such as heading out, rescued, etc.

4.1.2 Stimulus/Response Sequences

Stimulus: Rescuer opens the app and logs in
Response: List or map view of reports in their area of deployment

Stimulus: User clicks on specific report
Response: Detailed information about report pops up

Stimulus: User clicks button to add update and enters update status
Response: Server updates that rescue file with the updated status

Stimulus: Server receives a report in the rescuer region
Response: Mobile push notification

4.1.3 Functional Requirements

REQ1-1 Mobile application requests rescue data from the server
REQ1-2 Mobile application renders the rescue data in a table with the ability to refresh
REQ1-3 Mobile application renders a specific rescue page with details

REQ1-4 Mobile application has the ability to send status updates/photos to the server for a specific rescue
REQ1-5 Immediate and automated push notifications sent to the rescuer if they are in the region

4.2 Common Beachgoer Use Case

4.2.1 Description and Priority

Common beachgoers need a quick way to report stranded animals. The website must give these users the ability to report animals with the ability to mark the location, photos, and other details that will help with the rescue.

4.2.2 Stimulus/Response Sequences

Stimulus: User loads the page and fills out the form
Response: Server takes the rescue data and deploys it to the correct marine rescue crew. The user will be notified on the response with the marine rescue crew, as well as a way to donate to help the animal they rescued.

4.2.3 Functional Requirements

REQ2-1 Website form has to collect location data
REQ2-2 It has to be able to submit photos from a mobile device
REQ2-3 It has to give users the ability to record other notes
REQ2-4 All inputs have to be able with touch only REQ2-5 Website form has to have a CAPTCHA form to prevent fake reports
REQ2-6 Form has to submit to the server, and the server has to deploy the correct rescue team
REQ2-7 The form response should show the rescue agency as well as have an option to donate to help with the rescue

4.3 Administrator Use Case

4.3.1 Description and Priority

Each rescue organization will have an administrator who adds/removes/updates rescuers. They are responsible for creating users with the correct contact information.

4.3.2 Stimulus/Response Sequences

Stimulus: User loads the administration page on the website
Response: The user, if an administrator, is presented with a list of current users they can click on to edit as well as add/delete functionality

Stimulus: Admin clicks on user in list
Response: Page opens with the ability to edit information and assign a temporary password. User has to click save to keep the changes made on the page

Stimulus: Admin clicks on add user
Response: Page opens with the add user form. Admin assigns temporary password that has to be changed upon the first time the new user logs in

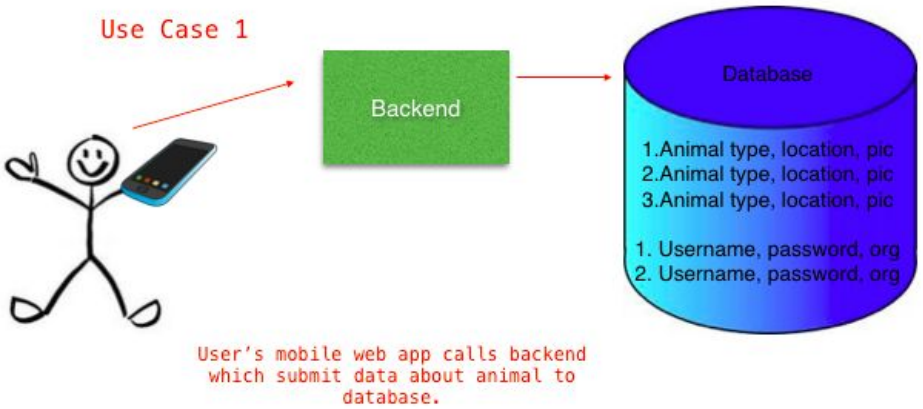
Stimulus: Admin clicks on the delete button next to the user
Response: Page asks for confirmation, and if confirmed, tells the server to delete the user

4.3.3 Functional Requirements

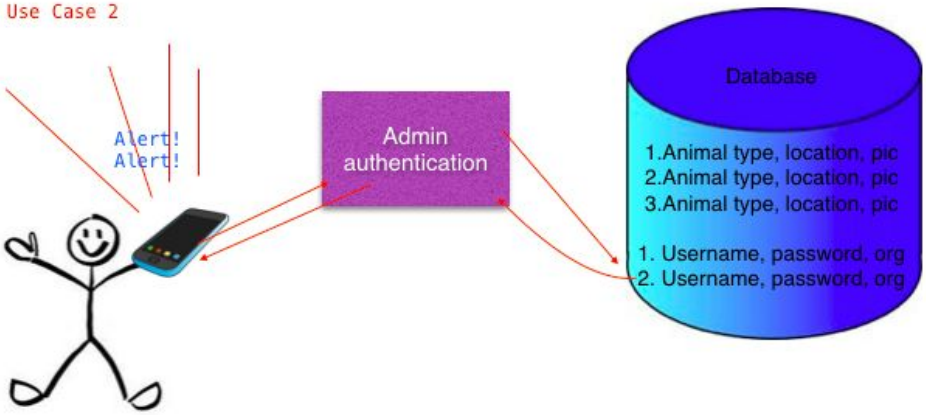
REQ3-1 Loading the admin page shows a list of users and actions the administrator can perform.
REQ3-2 An admin can click on a user to edit information or set a temporary password for the account.
REQ3-3 An admin can click the add user button to fill out the new user information and set a temporary password for the new user.
REQ3-4 An admin can, with confirmation, click a delete button next to a user to delete them.

4.4 Use Case Diagrams

The following diagrams outline the use cases discussed in the above section:

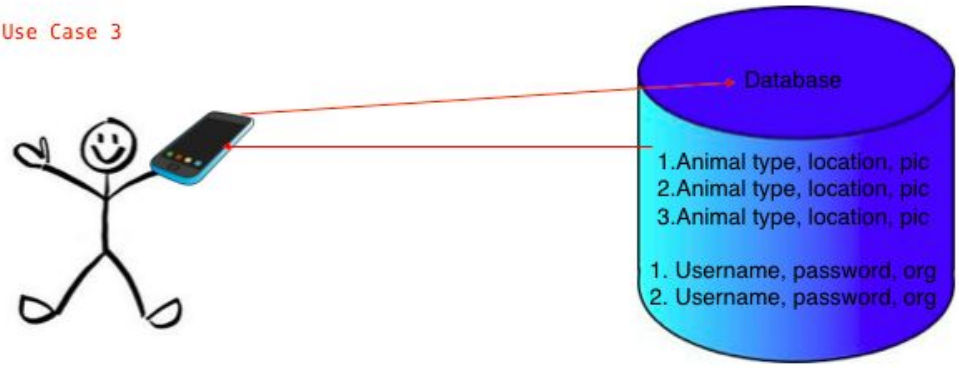


Use Case 2



Rescuer receives alert of stranded animal and logs into rescue iOS app.

Use Case 3



Rescuer's app queries database for information about nearby stranded animals.
Database returns information on relevant animals.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Performance across the system must be a very reasonable amount of time. A person who finds a marine mammal must be able to have their report of the incident reach the appropriate agency within minutes at the most in order to ensure that the mammal has a chance to be saved.

5.2 Security Requirements

Three parts of the service require authentication. Firstly, public posting through the website requires a feature to prevent massive amounts of fake submissions. Secondly, Agency rescuers require login and password for their authentication in order to receive the appropriate notifications from reports and sightings. Thirdly, the application must be an administrative portal from the website that requires a login and password.

5.3 Software Quality Attributes

Users of Rescue Reminders must have an iOS-compatible phone. The website must work on most major internet browsers such as Chrome, Firefox, Safari, and Internet Explorer. Rescue Reminders must be hosted on an AWS server, allowing for the expansion through cloud computing system services. Software must be simple to use where most people can submit a basic form on the website in a matter of seconds and the application must have a simple notification to understand.

6. Timeline

	Website	Server	Mobile App
Week 4	● Ability to submit text rescue report to server	● Form REST API for reports ● Basic login system	● Basic login page ● TableView with report information from server
Week 5	● Location	● Photo	● Map view of

	should be submitted with form ● Photo uploads in form ● Admin panel draft (TBD)	upload/display support ● Admin REST API (TBD)	reports ● Photo renders
Week 6	● Admin panel ● Donations ● Map view of rescues for public	● Push notifications ● Automatic rescue deployment by jurisdiction ● Admin REST API ● Status updates	● Push notifications ● Token-based authentication ● Status updates in reports (text/photos)
Week 7	● **Deployment** ● **MVP testing with client**	● **Deployment** ● **MVP testing with client**	● **Deployment** ● **MVP testing with client**
Week 8	● Successful rescues page with photos	● Directory API ● Successful rescues API	● Rescue directory (TBD) ● Offline functionality for remote areas
Week 9	● Search engine optimization for website ● Instructions for beach	● Data backup ● Reaching out to clients to startup their admin portals	● Submitting to Apple App store ● Making sure it fits Apple's app guidelines

	goer		
Week 10	● Refinement ● Backlog	● Refinement ● Backlog	● Refinement ● Backlog