Week Four Milestone Report

Project Nest

Massey University Capstone 2016 - Group Two

Team Members:

Zoe Purdon-Udy (13047928)

Francis Greatorex (14136398)

Sam Hunt (14216618)

MJ Lee (13214654)

Contents

Documenting Requirements

Requirements analysis

Introduction

Purpose

The purpose of this project is to produce a mobile application to help Department of Conservation (DOC) volunteers complete the resetting of predator traplines. Our project will migrate their data collection from paper to the mobile app and will include services for locating the traps. The mobile application will be complemented by a web application which will host the admin services of the entire project and will include data aggregation and other useful information.

Scope

The app will be available for mobile users on Apple and Android mobile devices. Trapline coordinators will provide volunteers with login credentials to access trapline data.
The user group is likely to include people of a variety of age groups and levels of computer literacy, therefore the interface needs to be designed to be as simple and intuitive as possible.

Out of Scope

The project scope does not include support for tablets or other WiFi enabled handheld devices (eg. iPod Touch). It is beyond the project scope to support devices without the necessary sensors to determine location. The accuracy of our product is bounded by the accuracy of the GPS service in the phone. Our product will not support Windows phones.

Core Functional Requirements

- The mobile application will work on iOS and Android
- The mobile application should support operating systems from Android KitKat (version 4.4) and iOS 7 up to the most recently released versions
- The mobile application can be used to locate traps using acoustic and/or visual signals
- The master data for a particular trapline can be configured
- An instance of the mobile application configured for a particular user can be protected by password. A user can only access data for which they hold the relevant permissions
- The application can be used to capture data, this data can be synchronised with a central data store. The data storage can be migrated to another platform if necessary
- Captured data can be exported to a CSV file

Additional Functional Requirements

- The web application can be used to edit trapline or user data (by users with the applicable permissions)
- The web page will show aggregated data displayed in tables and charts
- The mobile application can be used as a ‘cacophonometer’ to send auditory data to the Cacophony Project
- Photos can be attached to captured data
- Damage to the traps can be recorded using the mobile application
- The app is available in both English and Te Reo Maori

Non-Functional Requirements

- Accessability
    - The application should be accessible to those with hearing loss who it is plausible may be part of the user group
- Documentation
    - Documentation will be included in the source code of the API and app (javadocs), to help future developers understand the code
    - Static Frequently Asked Questions (FAQ) page on the web application
    - User guide produced for the mobile application
    - User guide for admin users of the web portal
    - Installation guide
- Extensibility
    - The system can be extended, through the addition of new functionality or through modification of existing functionality, with little impact to the existing system functions
- Interoperability
    - The system should be able to work with other products or systems without any restrictions on functionality or access
- Maintainability
    - The product should be able to be maintained with ease i.e. faults are able to be quickly identified and resolved
- Open Source
    - The source code will be made available with a license that allows users to study, change, and distribute the software
- Platform Compatibility
    - The mobile application will be able to run on Android OS (all versions from Android KitKat 4.4 onwards) and iOS (all versions from iOS 5 onwards)

    - The web application can be viewed on all standard browsers (including, but not limited to, Microsoft Edge, Google Chrome, Safari, Firefox)
- Resource Constraints
    - The local storage required for the mobile application will be kept to a minimum because mobile devices often have limited storage space
- Responsiveness
    - The response time of both applications will be below a certain threshold (not including calls that rely on external resources)
- Scalability
    - The system should be able to handle a growing amount of work i.e. with the adoption of the service by more DOC regions adding users and data
- Security
    - The data stored in the system will be reasonably protected from unauthourised access
- Usability
    - Installation Usability
        - The mobile application can be easily installed
        - The web application is available on all common browsers (e.g. Microsoft Edge, Google Chrome)

Requirements with priority weighting

Expected Delivery means the date when the requirement is expected to be supported by the product. Priority Level describes the importance of this requirement in the eyes of the development team.

Requirement	Note	Priority Level	Expected Delivery
Mobile app. available on iOS & Android	This is a core requirement specified by the capstone project outline and necessary to achieve the non-functional requirements of usability and accessibility	1	Week Four Prototype (14/08/16)
Support for mobile OS beginning at Android V4.4 and iOS 7	Requirement specified by capstone project outline	2	Week Four Prototype (14/08/16) Will not be tested on multiple versions until later in the project

Mobile app. can be used to locate traps	This is the most essential functionality of our app.	1	Week Four Prototype (14/08/16) With features and fine tuning ongoing throughout the project
Master data for a trapline can be configures	Essential functionality for the app.	2	Week Eight Milestone
Security for the trapline data	Requirement specified by capstone project outline	2	Week Four Prototype (14/08/16)
App. can be used to capture data, data is held in a central store, data can be migrated to another storage service	This is essential to satisfy the purpose of the app. and ensure the extensibility of the app.	1	Week Four Prototype (14/08/16)
Data can be exported as a CSV		3	Week Eight Milestone
Web app. can be used to modify user and trapline information		2	Week Eight Milestone
Web page displays aggregated data		4	Week Eight Milestone
Mobile app. can be used to record audio and send to other open source projects	This feature is above and beyond the initial scope set out by the capstone project outline, however, the developers think it would add value to the mobile product presented to DOC	4	Week Eight Milestone
Photos can be attached to captured data	This feature could add significant value to the data recorded	2	Week Eight Milestone
The mobile app. can be used to record damage to traps and associated hazards	This feature could add significant value to the application	2	Week Four Prototype (14/08/16)

The mobile and web applications can be displayed in both English and Te Reo Maori		4	Week Eight Milestone

Essential Use Cases

<table>
<tr><th colspan="3">1. Create account/login (mobile application)</th></tr>
<tr><td colspan="3"><b>Preconditions:</b> User has not opened the app previously</td></tr>
<tr><td><b>User Intention</b></td><td colspan="2"><b>System Responsibility</b></td></tr>
<tr><td>Access the Nest NZ mobile application</td><td colspan="2">Display the login screen - prompt for email address and password</td></tr>
<tr><td>Enter email and password</td><td colspan="2">Check login information is valid</td></tr>
<tr><td></td><td>Display login confirmation/welcome message</td><td>Display error popup</td></tr>
<tr><td></td><td></td><td>Prompt the user to re enter their login information</td></tr>
<tr><td></td><td colspan="2">Display trapline selection screen (‘home’ page)</td></tr>
</table>

<table>
<tr><th colspan="2">2. Place trapline</th></tr>
<tr><td colspan="2"><b>Preconditions:</b>
Trapline has previously been created through the web application
User is logged in with trap modification permissions</td></tr>
<tr><td><b>User Intention</b></td><td><b>System Responsibility</b></td></tr>
<tr><td>To set up a new trapline</td><td>Display trapline selection screen</td></tr>
<tr><td>Select the trapline to modify</td><td>Show trapline as selected</td></tr>
<tr><td></td><td>Display trapline info page</td></tr>
<tr><td>Select menu</td><td>Display menu options</td></tr>
</table>

Select ‘Add Traps’	Show add trap screen
Add a trap at current location	Show trap as saved (via a confirmation screen)
	Cache trap information to the local storage
	Display the add trap screen
Repeat until all trap geolocations and data are recorded	
	Display trapline info screen

3. Perform reset of trapline

Preconditions: User has logged in

User Intention	**System Responsibility**	
Reset a trapline	Display trapline selection screen	
Select a trapline	Display trapline info screen	
‘Start’ the reset process	Show ‘Start’ as selected	
	If relevant: Display trapline direction screen	Display compass screen
Select a direction to reset the trapline	Display compass screen	
Record catch information	Display the catch screen	
Select a catch type	Display catch confirmation screen	
	Cache information to local storage	
	Display compass screen	

Project Infrastructure

Introduction

The process used to select the project management infrastructure was fairly informal. Most products were chosen for use based on team members knowledge and previous experiences.

Repository

Github was chosen as our repository due to several team members owning student accounts allowing for the creation of a private repository.

Communication

Team communication will be through the Slack application. Initially the team used Google Hangouts, this was chosen due to its availability through Gmail. Hangouts was not liked by the majority of the team, however, due to the difficulty of searching through messages and the lack of an easy ‘categorising’ feature for conversations. Slack has functionality which resolves the issues that were encountered with Hangouts. Facebook Messenger was also initially considered as a possible communication channel although some team members had objections to creating a Facebook account and so it was not chosen.

Issue Tracking System

Formal issue tracking will be completed through the built in Github issue tracking system. Trello will be used for informal issue tracking, progress tracking, and as a place to record ideas for future improvements.

Project Management Software

Google Sheets will be used for project timelines and task allocation (by the project coordinator) as it is easy to use, familiar to the project coordinator, and accessible by all team members. Trello will provide support for self management as each team member will record their own tasks and mark them as completed or in progress.

Project Diagrams

Microsoft Visio will be used to complete project diagrams including database designs and component architecture diagrams. Visio is free for the development team under the Microsoft student licensing agreement with Massey University and provides the ability to create professional diagrams using correct UML structures.

Process Automation

The build process for the mobile application code will be automated using Gradle. Gradle will be used as it is the default for Gluon. Gradle is known by the relevant team members and is simple for other team members to learn.

Ant is being used to automate the build process for the API/backend as it is the Netbeans default. Netbeans is being used as the IDE for this development because it has the Tomcat server library built in for testing on localhost.

Product Deployment

The mobile application for iOS will be made available on the App Store. The Android mobile application will be deployed on the Google Play Store.

The web server will be deployed via AWS.

Life Cycle Model and Project Plan

Introduction

The project has been broken down into four main components and each team member has been assigned the responsibility of coordinating the development in this area. This strategy of peer programming with the responsibility held by one team member (with oversight by the project coordinator/team leader) will help to ensure consistent progression throughout the development as each programmer is responsible not only for themselves but for an essential section of the final product. An iterative Agile life cycle model was chosen and major milestones were planned as a guideline for the delivery of certain features and functionality.

Life Cycle Model

An iterative model has been adopted by the development team. Our initial iterative process will focus on the delivery of a subset of the requirements. Each further iteration will enhance the software until the system is complete and ready for deployment. The team will start development with most of the requirements defined, however, the iterative model supports reviewing the requirements, and adding or delaying features, at the conclusion of each iteration.

The process used to select this model of development was fairly informal. The team of developers discussed common models (found using a Google search), a review of each of these, using online resources, showed an iterative Agile approach aligned well with the planned work process. In the table below (from tutorialspoint.com) the team found many of their desired workflow attributes were featured as ‘pros’ of the iterative model, e.g. a working product is developed early in the life cycle

Pros	**Cons**
• Some working functionality can be developed quickly and early in the life cycle. • Results are obtained early and periodically. • Parallel development can be planned. • Progress can be measured.	• More resources may be required. • Although cost of change is lesser but it is not very suitable for changing requirements. • More management attention is required. • System architecture or design issues may arise because not all

• Less costly to change the scope/requirements. • Testing and debugging during smaller iteration is easy. • Risks are identified and resolved during iteration; and each iteration is an easily managed milestone. • Easier to manage risk - High risk part is done first. • With every increment operational product is delivered. • Issues, challenges & risks identified from each increment can be utilized/applied to the next increment. • Risk analysis is better. • It supports changing requirements. • Initial Operating time is less. • Better suited for large and mission-critical projects. • During life cycle software is produced early which facilitates customer evaluation and feedback.	requirements are gathered in the beginning of the entire life cycle. • Defining increments may require definition of the complete system. • Not suitable for smaller projects. • Management complexity is more. • End of project may not be known which is a risk. • Highly skilled resources are required for risk analysis. • Project.s progress is highly dependent upon the risk analysis phase.

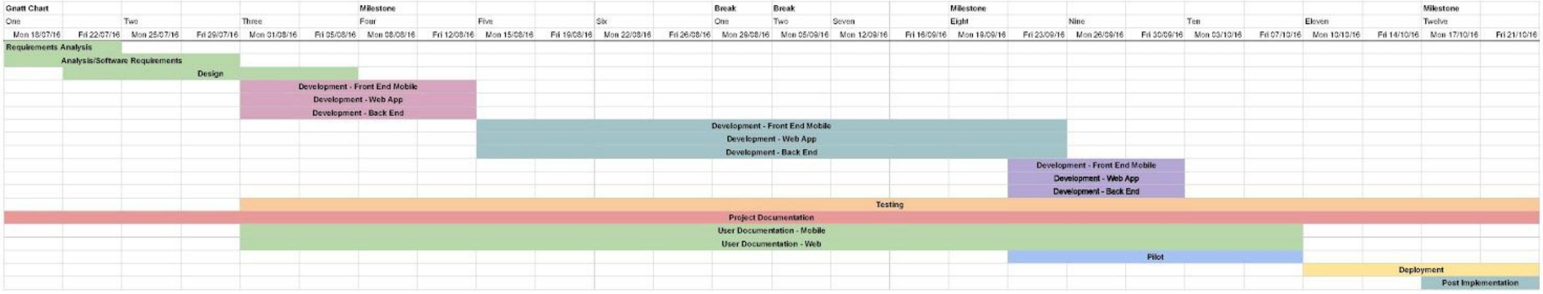
Table aquired from http://www.tutorialspoint.com/sdlc/sdlc_iterative_model.htm

Major Milestones

There are three major milestones for this project:

1. **Week Four (14/08/16)**
   The demonstration of a working prototype of the mobile application with a subset of the required functionality implemented
   Requirement analysis and design specifications completed
2. **Week Eight (25/09/16)**
   A prototype with all desired features implemented
3. **Project completion date (23/10/16)**
   All functionality thoroughly tested across the required platforms.
   The mobile application ready for deployment through the Apple App Store and the Google Play Store.

The Gantt chart below shows the planned stages of development, and when other major components of the project will be completed.



This chart can also be viewed at https://docs.google.com/spreadsheets/d/1kVLXAU8-WfXK5kPNbqRe_8Wi0OQh0K5fQwwEWPsVW-0/edit?usp=sharing

Coordinating Roles Specification

Zoe Purdon-Udy

Team leader/Project coordinator

Testing coordinator

Write weekly team leader reports
Write/compile milestone reports
Facilitate communication between team members, project supervisor, Henning Koehler, and the 'user base' (primarily Stephen Marsland)

Ensure each team member is completing the necessary testing for their code
Complete integration tests, perform acceptance testing, and test for the non-functional project requirements

Francis Greatorex

Gluon lead & mobile coordinator

Create the mobile applications (both Android and iOS) using Gluon

MJ Lee

Web coordinator

Research a cloud provider
Create the web application

Sam Hunt

Backend coordinator

Define the RESTful API
Manage server side code

Technology Selection

Introduction

Technologies to be used in the development of this project were chosen based on the following criteria:
Ease of gaining licensing to use the technology
The current skills of each team member (and how quickly team members unfamiliar with the technology could learn the basics)
Support available for the technology
Industry opinion of the technology

Each technology was decided upon during initial group meetings. The only category formally researched was the web services provider. This research is linked in a below section.

Technologies selected

Gluon

A Java-based cross platform mobile development kit

Gluon was an easy decision for the team as all members are confident with Java development. Francis has also completed some work for the Gluon team so has a good understanding of the product. Gaining a student license for both the mobile and cloud Gluon products was very simple.

Amazon Web Services (AWS)

Cloud services

AWS will be used to provide cloud storage (S3 bucket) and to host our server (EC2) and postgres database. MJ completed research on our options for cloud services which can be viewed at https://docs.google.com/document/d/1biuBPQ6sabQDfYhBYSTe8mhz_VuFwJ06PrK2UjY0a7o/edit?usp=sharing. The decision to use AWS was based on this information; primarily the cost, and MJ's experience working with the service in a previous course.

Github

Online project hosting using Git

Francis and Sam both hold student accounts with Github allowing the team to create a free private repository to host the project code. We selected Github over the alternative options due to experience working with Github by team members and a preference for the Github web interface.

Tomcat

Web server

Tomcat is the standard for working with Java Server Pages (JSPs) and Java Servlets, and has been used by team members previously.

Gradle

Open source build tool for continuous delivery

Gradle is a simple and easy to use build tool, which is widely used throughout the software development industry. Gradle is the main build tool supported by Gluon, therefore it will be used to write build scripts for the mobile application.

Ant

Pure Java build tool

Ant will also be used as a build tool on this project. It will be used by the backend coordinator because it is the default for the Netbeans IDE.

PostgresSQL

Object-relational database management system supporting SQL

PostgresSQL was primarily chosen because the backend coordinator has experience working with it and no other team member had strong preferences for another database.

AngularJS

Enhanced HTML for web apps

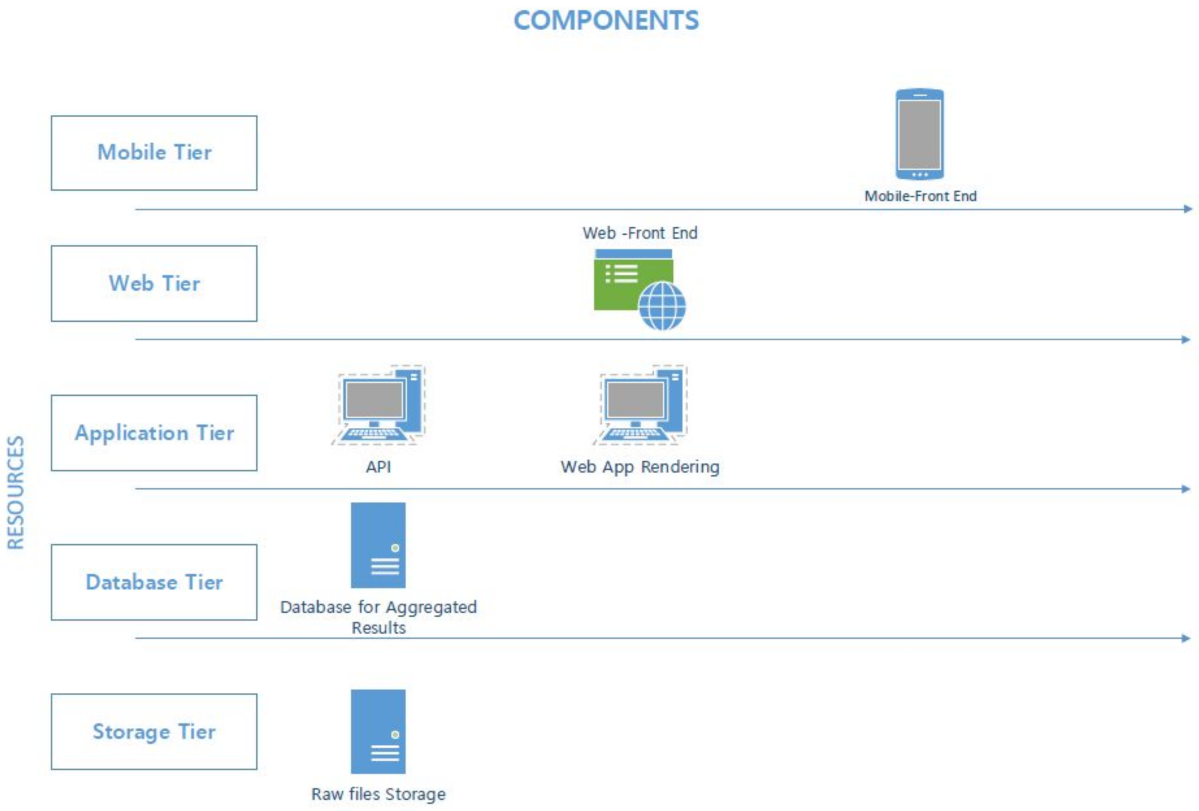
AngularJS was chosen because it is commonly used in industry and the team members involved in the web app development wanted to learn it. It adds to JavaScript to allow for quick development of web front ends and gives the ability to modularise HTML and JavaScript providing good separation of concerns.

Architecture and High Level Design

High Level Design

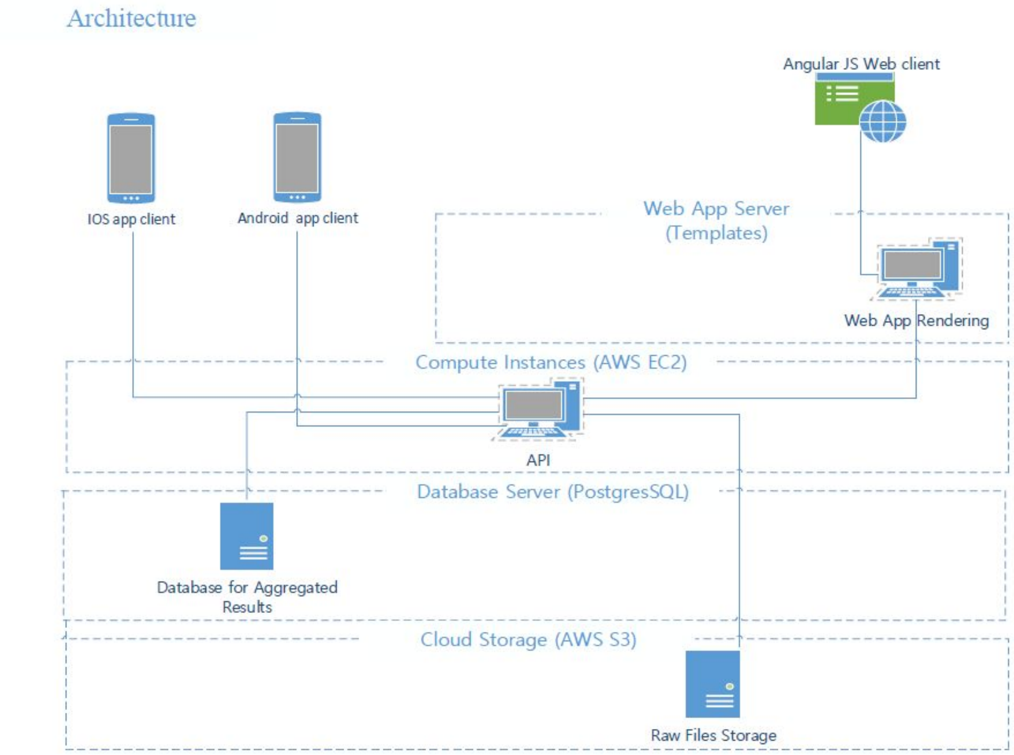
The main components of the software are isolated from one another through the use of JSON and HTTP. HTTP and JSON are cross technology and therefore all languages have the tools to use them. In this way the decisions about web technologies and the language for mobile development could be made independently.

Component Design

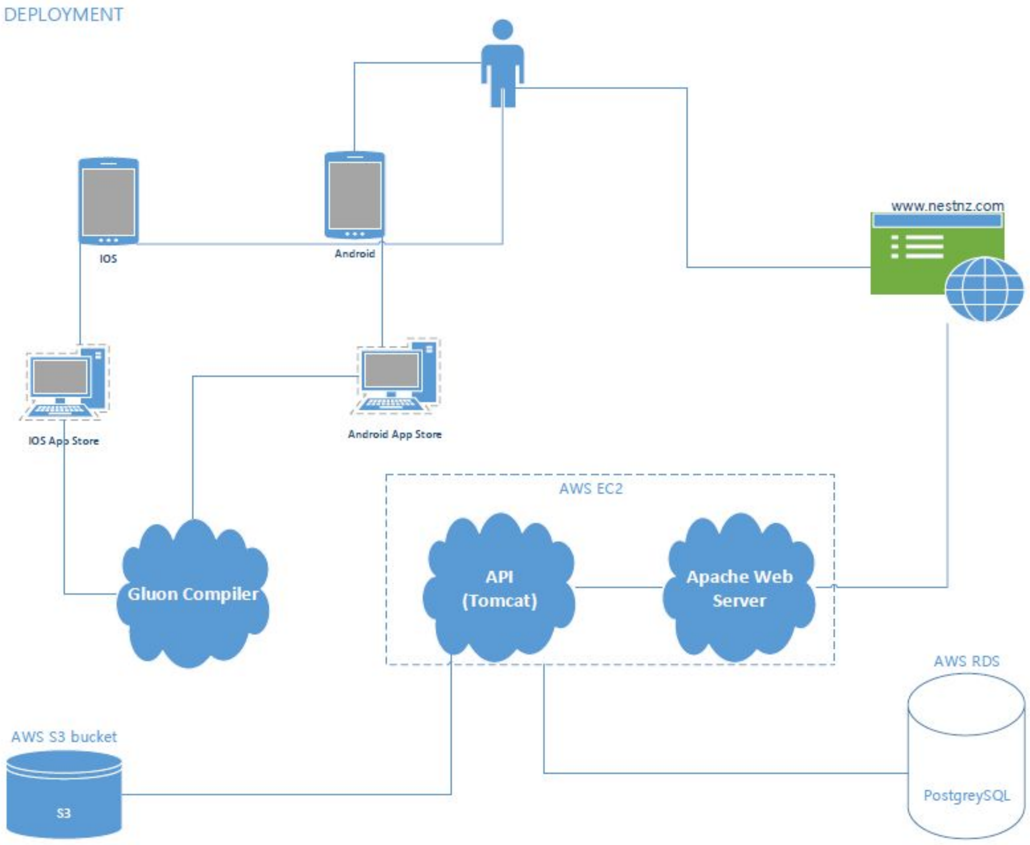


The Component Design and the following two design diagrams can be viewed at https://docs.google.com/document/d/1puCiuNL5_WXb8E9qtStG2g345FVGSBpstEfxYQb3Keg/edit?usp=sharing

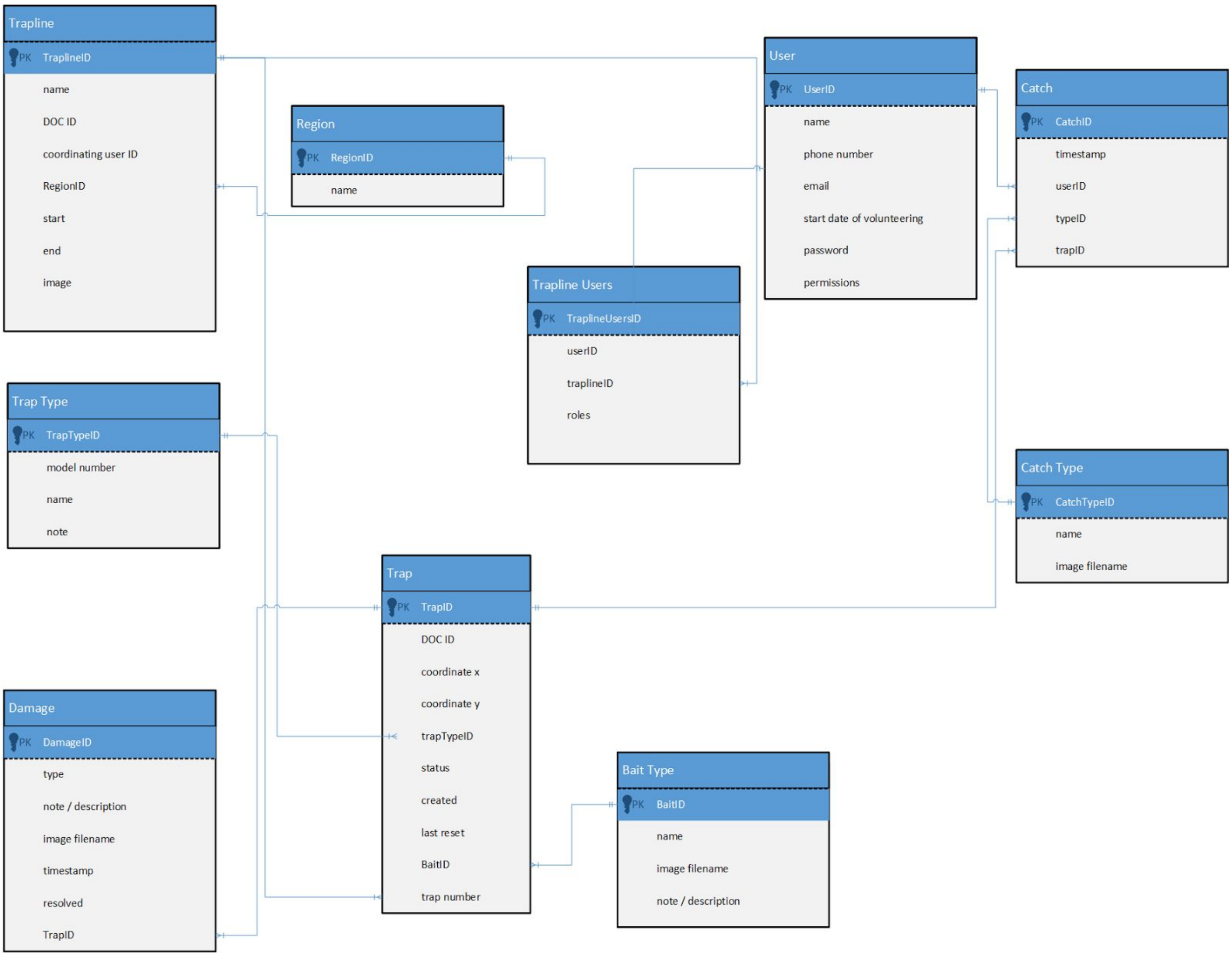
Architecture Design



Deployment Diagram



Database Design



This image can also be viewed at https://drive.google.com/file/d/0ByC-t7ntakfXZmdBWksxUEVNWEU/view?usp=sharing

RESTful API Specification

The RESTful API Spec can be viewed at https://docs.google.com/document/d/1zPCohwShxziF4YAV0E3PPZWvtZiAU8KaAOtFYIzzbc4/edit?usp=sharing

Risk Management

Introduction

This document will outline the possible risks to the successful completion of the project and will describe how the project team plans to mitigate those risks.

Risks

A member of the team is unable to continue their participation in the project (High risk & severe impact)

A team member may need to leave the project unexpectedly for either an extended period of time or indefinitely

The possibility of this risk influenced the choices made by the development team on the technologies to use. Technologies that were known to members of the team were chosen and each member was encouraged to familiarise themselves with the technologies used by others in the team. The team's communication strategies also aim to lessen the impact should this situation eventuate; there is a record of what each team member has achieved and their goals for the near future on Trello.
If a team member should leave the project, each remaining member of the team will take on a portion of that member's responsibilities. The increased workload will be distributed according to the overlap between the leaving members skillset and those remaining, and the current workload of the team.

Loss of code (Medium risk & severe impact)

Code for the project may be lost

To prevent this occurring the team will make use of a central code repository (on Github) and individual members of the team will be encouraged to regularly backup their work.
If code is lost it will be the responsibility of the coordinator whose section the lost code was relevant to, to ensure the adequate re-completion of the work.

Feature creep (Medium risk & low impact)

The features/deliverables of the product may be continually added to causing over-complication of the software

An occurrence of feature creep is a risk to the project due to the implementation of additional features holding up the development of core functionality. The prevention of feature creep will result from the initial requirement analysis documentation outlining the desired functionality and additional features, with their priority weighting clearly defined. The development of additional features will ordinarily only begin when a previous feature has been fully incorporated into the product.

GPS in the Manawatu Gorge (Medium risk & severe impact)

GPS may not be accurate in the areas that the mobile application is required to be usable

This risk will be addressed as quickly as possible (on the 14/08/16 field trip to the Manawatu Gorge where team members will assess the accuracy of the GPS location data given to them by existing applications available on the Android app store). If it is the case that GPS is not available, the project requirements will need to be reevaluated and the outcomes adjusted as to what is reasonably achievable with the technology available.

Server Availability (Low risk & medium impact)

The server may go down causing our applications to have no live backend

Our choice of web services provider has lessened the probability of this risk so that it is extremely minimal and can be ignored with confidence. Amazon Web Services (AWS) guarantees 99.5% uptime in their Service Legal Agreement. The design of the mobile application will effectively eliminate any issues with server unavailability, data recorded via the mobile app will only be uploaded to the server when it is available.

Quality Assurance

Testing Strategy

Developers will be in charge of writing unit tests for their own code. The person who develops each feature will have a greater understanding of what needs to be tested than the testing coordinator, though the testing coordinator will be in regular contact with each developer to ensure they're including the necessary tests. The test coordinator will write integration tests to find errors/bugs in the interaction between software components and issues with interfacing with external resources. Functional tests will be completed to verify the create, read, update, and delete (CRUD) methods for the API, and to verify all the methods defined in the RESTful API Specification. SeleniumHQ will be used for browser automation testing for the web application. Acceptance testing will be completed in two stages:

1. Alpha Testing : The four members of the development team will test the application, in various realistic situations (including setting up traplines and locating specific points in settings where the app would be realistically used)
2. Beta Testing : The app will be given to DOC volunteers from the local region to trial under 'real world' conditions. The feedback from this phase of testing will determine any feature changes to be completed before the product release.

Issue Tracking Policy

Github provides built in issue tracking including the ability to add new issues, mark current issues as complete, and assign issues to specific repository contributors.

The policy for dealing with new issues that have arisen since the last team meeting (and which have not been assigned to someone on creation) will be to discuss the issue as a team and assign the resolution of this issue to the team member with the most relevant skill set, taking into consideration each team member's current workload. Issues will be prioritised, as will feature changes, according to their importance for the progression of the project.

References

https://aws.amazon.com/ec2/sla/
http://www.tutorialspoint.com/sdlc/sdlc_iterative_model.htm
https://en.wikipedia.org/wiki/Interoperability
https://en.wikipedia.org/wiki/Extensibility