Lost & Found

Members:

Conor Smyth - 12452382

Adam O'Flynn - 12378651

Supervisor: Brian Stone

Functional Specification for 3rd Year Project

Table of Contents

1. Introduction

1.1 Overview

The application is a geolocation aided Lost and Found iOS application. Users will be able to post on the application that they have either lost or found an item with some information regarding how the item can be collected or returned. They can also add a picture of the item to the listing. A special feature of the application will be the ability to drop a pin on a map of the location of where the item was lost or found with easy to access information about the item from that pin.

The application will act as a forum for people who wish to find an item they have lost or for people who wish to return an item that they have found. The listings will have a number of easy to understand features to help you get a good understanding of what the listing is about and how to contact the person who created the listing.

The application will have a number of different functions.

- The application will react to your location to see what listings are listed near you. There will be a function to disable this feature if you don't wish to see only the listings near you. If the user does not wish to reveal their location, they can not allow the app to access it.
- Users can create a listing which is a post on the application for a lost or found item.
- When creating a listing users must provide some information about the item, have the option of a picture and the option of dropping a pin on a map to display exactly where the item was found.
- There will be a categories page where listing will be split up into their respective categories allowing for users to easily search for specific listings.

- There will be an option for users to log in to see previously viewed listings and previously created listings.

1.2 Business Context

There is a potential for a couple of business contexts regarding this application.

1. **Advertising**: If we feel the application is doing well enough, we could potentially use adverts to generate some revenue to help with the hardware side of hosting the app. We would implement the advertising with a small banner ad at the bottom of the page. The user will be allowed close the advert for the current session if it bothers them.
2. **Paid Exposure**: If a user wanted to, they could bump their listing to the top of the list for a small fee. Again, if the user wants their listing to be seen by more people, we could implement a featured listing section for a small fee.
3. **Invest/Sell The Application**: If our application is successful, companies may want to invest in our application. Also, if successful, we may be approached by companies with an interest in purchasing our app.

1.3. Glossary

- **Objective C**: iOS programming language
- **HTTP**: Hypertext Transfer protocol. The language used for the world wide web to display web pages.
- **REST/RESTful programming**: REST is an acronym for Representational state transfer. The basic idea behind RESTful programming is to implement an API that allows the client to know nothing about how the structure of the API making use of how HTTP handles resources using keywords such as:
    - GET: request information.
    - PUT: update information.

    - POST: create new information.
    - DELETE: remove information
- **SQL:** Acronym for Structured Query Language. Used for storing and requesting information from a database.
- **iOS**: Operating system developed by Apple for use on apple software and hardware.
- **GUI**: Graphical User Interface. The interface that the users see in graphical form.

2. General Description

2.1 Product/System functions

- **Register Account**
  The user will have to register an account to place an listings or to email another user about a listing. The user will be able to look at listings without an account but will be unable to interact with them. Most functions will need an account. Account will be registered and verified with normal email.

- **Log into Application**
  After the user registers their account, they will need to log in. User enters their email and their chosen password.

- **Search for listing**
  User can search both lost or found categories or either one of them separately. User enters their desired search, application returns listings with match the search. User may also wish to search individual category types like electronics, tickets etc.

- **Place Listing**
  Once a user has registered an account and logged in, they can place a listing in the application. When the user goes into the listing page, they will need to enter required fields. These are; email address, location, last seen/found and category. They can add optional fields like a picture of the object, a reward (if applicable), and other means of contact such as phone number.

- **Edit a Listing**
  To edit a listing, the user needs to be signed in. Then, they need to go to their profile page to view their currently active listings. There, they will click into the listing and click an edit button at the top of the screen. They then save the changes and the ad updates.

- **Edit Profile**
  When logged in a user can edit their profile by clicking on the my profile tab and clicking edit at the top.

- **Check nearby listings on map**
  Since one of our main features of this app is to be able to check listings from a certain area, the user will be able to check listings from a map. The user will switch to the map tab and either enter a location or let us decide where they are through gps location. After this they can click into listings from markers on the map. The markers should show a listing preview with a picture if applicable.

- **Email owner of listing about listing**
  To enquire about a listing, the user either emails or calls the user with the information given in the listing. This information won't be available to the general public and you will need to be a member to be able to do this. When you click email the application will send you to your mail app on your phone so you can email the other person. No information will be exposed on the listing. It will be hidden behind buttons.

- **Remove a listing**
  When a listing is either returned or found, the user can delete the listing by editing it and placing a found or returned notice on their listing. If the user wants to take down a listing pre-emptively, they can do so the same way.

- **Report a Listing**
  If a user discovers one of their belongings is up on the application as a "found item" they can report the listing by clicking the report listing option found on the listing page. If we have enough evidence to back this up, the listing will be taken down and appropriate measures will be taken. Their account may be suspended or deleted if serious.

- **Log Out**
  If a user needs/wants to log out, they can logout by clicking on the my profile tab and clicking log out of account.

2.2 User characteristics

Our application will use the iOS platform. The user demographic we are targeting are people of all ages who have an iPhone or iPad with an Internet connection.

The expected expertise with software from our users is expected to be people with a basic understanding of how to use an iPhone app and also people of greater expertise.

Objectives and requirements expected from our users is an easy to use accessible application that behaves exactly as expected. Our plan is to create something that is very attractive to look at but also have the functionality and ease of use required to meet the expectations of our targeted demographic.

2.3 Operational Scenarios

Our application has three operational states; registered and not logged in, logged in and not registered. Therefore, there will be different scenarios depending on this.

- **Registered and not logged in:**
  When the user first registers their account, they will need to log into the application.
    - Log into Application:
      When a user is registered they can log into the application through their account. The user will enter their email address and their chosen password. If correct, they will move to the 'logged in' state and access all the features of the application.
- **Logged in:**
  When a user is registered and logged in, they will have full access to every action in the application. They aren't restricted from using any feature.
    - Search:
      When a user is logged in and wants to search for a desired item, they input what they would like to search in the search bar. The application returns listings matching the query. The user can refine the search with filters like lost, found, category or area.
    - Place Listing:
      A user that wants to place a listing will click on the place listing button on the main of the application or in the my profile page. Once they click this, they will be brought to a listing page with fields to fill in. Some are optional. When they are finished and happy with their listing they can confirm it by clicking "Post Listing".

- Edit Listing:
  Users may want to edit their listing information and update it. To do this, the user will need to go into the profile page and click my listings. There, they will be able to see and edit any of their currently live listings. Next, they choose the listing they want to edit and edit it accordingly. They are brought to a page, like the place listing page, that already has the information. The fields are editable again and the user can confirm changes by clicking "Update Listing".

- Check Nearby Listings on Map:
  This scenario is one of the main aspects of our application. The user may want to check their local area for an item lost or found item. Hopefully, the user will allow the application to find out where they are. Then the map will show their location with different listings around them/ They can interact with these listings and click into them. If the user doesn't reveal their location, they can search an area in the search box and view the listings in that area the same way.

- Email Owner of Listing:
  To email an owner of a listing, the user will need to go to the owner's listing and click the email button. They will be brought to the email app on their phone and from there, they can email the user. We may implement some messaging system inside the application in the future. We will show the user a warning that they are going to the mail app when they click email owner.

- Remove a Listing:
  When a listing has been found or returned, the user may want to remove the listing from the live search. They can do so by going to the my listing page and editing the listing to show that it is been found, returned or withdrawn. If they select withdrawn, they will

need to give a reason why. The listing isn't deleted and is kept on the users profile as a way to let other users see their history.

- Report a Listing:
  A user can report a listing if they have legitimate reasons to report it i.e. it's their stolen item. To report a listing, the user clicks the report listing button on the listing page and will be brought to a page to file the report. There, they will have to fill out the reason and their evidence. They may only report the same listing once. To combat spamming, users can only report a listing every few minutes and if they abuse this, their account may be investigated and suspended for a short while if found to be trolling.

- Log Out:
  Users can log out of the application when they are finished with using it or they want to protect their data. To do this, the user would select the profile tab and click the settings wheel at the top of the page. A drop down menu will appear with a logout option and various other options. They can click the logout button and will be brought back to the login screen.

- **<u>Not Registered and not logged in:</u>**
  To make our application more user friendly, you are not required to register an account or log in to see the basic functions of it. You can check listings and search the database of listings. They may view other user's profiles but may not contact them. The user will be able to use these functions when not logged in:

  - Search listings
  - Check nearby listings on map

  All other functions not mentioned above will be unavailable to unregistered users.

  One new scenario in this state is to register an account.

- Register Account:
  A user is able to browse and use the most basic functions of the app but will need to register an account to access the full application. When the user first downloads the app, they will land on the login page which will have a register account option. When they click this, they will be taken to a page where they input their name, email address, username and their chosen password. When they complete this form, they will receive a confirmation email to verify the account. After they verify the account, they will be allowed to login into their account and access the full application.

2.4 Constraints

- **Time**: This project is due Saturday 14th March. This leaves us little time to expand the application based on new user requirements suggested through user feedback.
- **Platform**: As we don't have a mac PC we do not have access to the required tools needed for iOS programming. We will use VMware and a Virtual machine to run OSX.
- **Moderation**: Moderation of the forums is very important to ensure all listings are legitimate and sufficient. We will have an option to report that a listing is insufficient or if the user has had any problems with the person who posted the listing.
- **Programming Language**: Having never programmed in Objective C it will pose some difficulties. We will learn Objective C through practise and online tutorials.

3. Functional Requirements

3.1 Sign Up/Sign In

- **Description**: The user will have the option to become a member or just browse application. The user will be prompted to sign in upon entering the application and they can choose to do so as they wish. To sign up the user will be supplied with a form to submit with a name and email address. The user must then confirm their email address through an automated validation email.
- **Criticality**: This requirement is essential for users who wish to make a listing or contact another user through a listing. Without signing up and using their account they will have little functionality other than just viewing listings.
- **Technical Issues**: For users to sign up we need to correctly store their information properly and securely in our SQL database. When users sign in we need to retrieve and validate the information entered correctly.
- **Dependencies with other requirements**: As users are required to have an account to make a listing or contact someone through a listing both of these requirements will depend on this requirement.

3.2 Place Listing

- **Description:** If a user wants to post about an item they must create a listing. The user must have an account to create a listing. To create a listing the user will be required to supply some information about the item that has been found and will have an options to post a picture and drop a pin on a google maps view for other users to see an exact location where the item was lost or found. The user will have to enter a category for the item allowing for more filtered searches.

- **Criticality**: This requirement is important for the application as the whole application is based around users creating, interacting and viewing listings.
- **Technical Issues**: To create a listing we need to carefully store the information supplied by the user. The information of the creating user must be stored correctly along with the listing information they have supplied for the listing in our SQL database.
- **Dependencies with other requirements**: This requirement depends on the user having an account therefore depends on the Sign Up/Sign In requirement.

3.3 Search for Listing

- **Description**: This requirement allows users to search for a listing. Users searching for a listing can search by supplying a search phrase which will return relevant results. Users can filter by category and also search using a google maps view to see pins put on the map for listings.
- **Criticality**: This requirement is important to allow users to search for an item they might have lost. The search will filter the results to allow the user to see only the listings relevant to what they are seeking.
- **Technical Issues**: Searching for an item will require very specific queries to our SQL database which may cause problems. We must also ensure that results from the search are relevant. We must ensure that results are not out of date or already resolved.
- **Dependencies with other requirements**: This requirement is dependent on user posting a listing. For a search to yield results there must be listings to search.

3.4 Edit Listing

- **Description**: This requirement allows users who have created a listing to edit the listing they have created. Users can go to their listing through searching or through their profile. Users will have an option to edit the

listing provided that they posted that listing. When a listing has expired or resolved users will not have the ability to edit the listing.

- **Criticality**: This requirement is not as important as others. It will be nice to facilitate users who may have made a mistake on their listing to edit and fix and mistakes they have made however this is not essential.
- **Technical Issues**: We need to be able to ensure that the only users that can edit a listing are the users who have created that listing. We also need to ensure edited information is updated in the SQL database correctly to allow for proper future retrieval. Users must be restricted after a certain time or if the listing has been resolved to edit a listing.
- **Dependencies with other requirements**: This requirement depends on a user having created a listing.

3.5 Edit Profile

- **Description**: This requirement allows users to edit the information on their profile. Users can access this function through the profile page on the application. They can edit their email address which requires validation.
- **Criticality**: This requirement is not essential however we feel it is important to allow users to mend their information if it has changed in any way.
- **Technical Issues**: Issues with this requirement involve updating the SQL database with the new information. The existing information will have to be changed.
- **Dependencies with other requirements**: This requirement depends on a user having a profile so it depends on users having Signed Up or Signed In.

3.6 Remove a Listing

- **Description**: This requirement allows a user to remove a listing that they have posted. Users can access their own listings through their profile or through a search and will have the option to remove the listing. The user

must mark that the item has either been found or returned. If the user wants to remove a listing preemptively they must supply a reason for removing the listing and must confirm but the listing is completely removed.

- **Criticality**: This requirement is important to allow users to mark that a listing has been resolved as either found or returned. Also allows users to remove a listing that they may have submitted by accident.
- **Technical Issues**: The issue with removing a listing will be ensuring the listing is removed from the SQL database correctly so that the database stays consistent. We must also make sure that the user removing the listing is the owner of that listing.
- **Dependencies with other requirements**: This requirement depends on the Place a listing requirement as the user must a listing to remove before they can remove it.

3.7 Report a Listing

- **Description**: This requirement allows a user to report another listing if they feel that the listing is not legitimate. On each listing there will be a Report button to allow users to report the listing. The user must then supply a reason for reporting. Reported listings will be flagged and hidden so that it may be checked for it's validity. If a post is considered invalid the owner of the removed listing will be notified and warned. The same user may not report a particular listing more than once and can not report more than one listing every few minutes. If a user incorrectly reports valid listings too often they will be
- **Criticality**: This requirement is very important to the application as invalid listings will result in a very inconsistent forum. Users who feel that a listing is not valid may report the listing and this will help make allow for mostly valid listings.
- **Technical Issues**: Some users may be reporting listings for no reasons. This will be dealt with through removing their ability to report listings or in extreme cases remove their account and ban their email address. It

will also be difficult to validate listings and will require some moderation. It is also important to mark found or returned listings on our SQL database and remove them from searches.

- **Dependencies with other requirements**: This requirement depends on other users searching and viewing other listings.

3.8 Email owner of Listing

- **Description**: This requirement allows users to email another user about the listing that that user has submitted. There will be a button the listing that the user can press which will allow them to send the owner of the listing with any queries that they have about the listing.
- **Criticality**: This requirement is extremely important as it is the idea of the whole application. Users must be able to contact the user who has submitted a listing with any queries about the item whether the lost item is theirs or they have seen the item or found the item that the user is looking for.
- **Technical Issues**: A problem with this requirement will be correctly opening and an email interface supplying the correct email address to send to.
- **Dependencies with other requirements**: This requirement depends on the user having an account therefore depends on the sign up/sign in requirement.

4. System Architecture

4.1 System Architecture Diagram

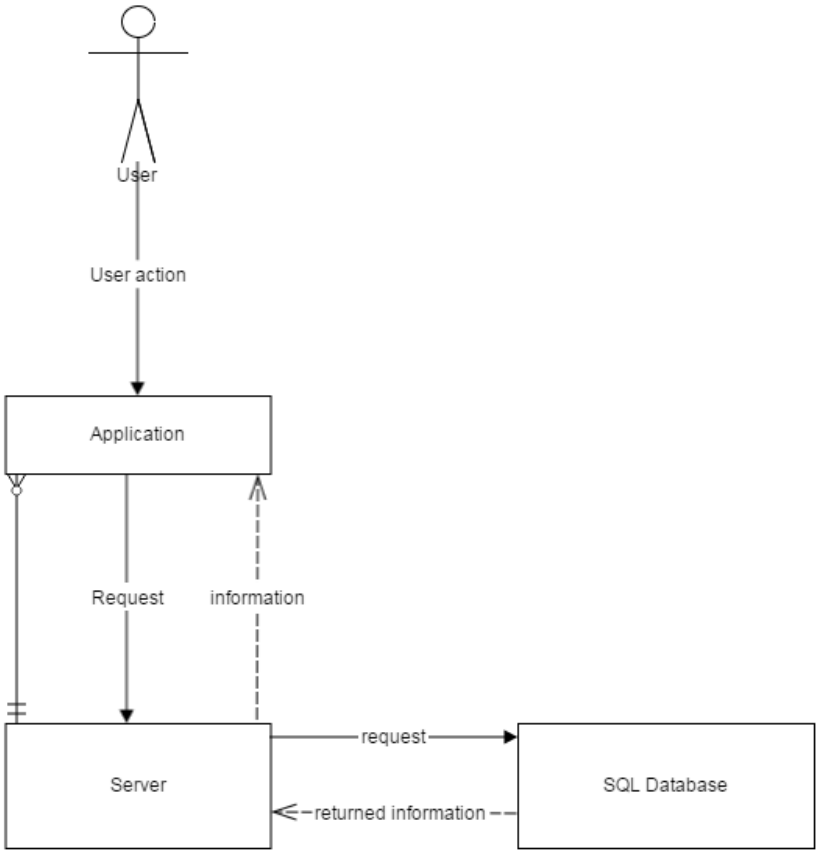


Fig 4.1

4.2 Application

The application will be the frontend of the system with a GUI and functions implement in Objective C. The GUI will respond to the users interactions with different functions that will send requests to the server to PUT, POST, GET or DELETE data.

4.3 Server

The server will receive and manage calls from the clients and respond where relevant with the information required. The REST api implementation will allow us to easily handle all requests made and how we shall deal with them.

4.4 SQL Database

This database will be holding all the information needed by the client to use the application properly. Requests coming from the client will be handled by the server and sent to the database to retrieve the information required. The information will then be sent back to the client through the server who will display the information in the correct format.

5. High Level Design

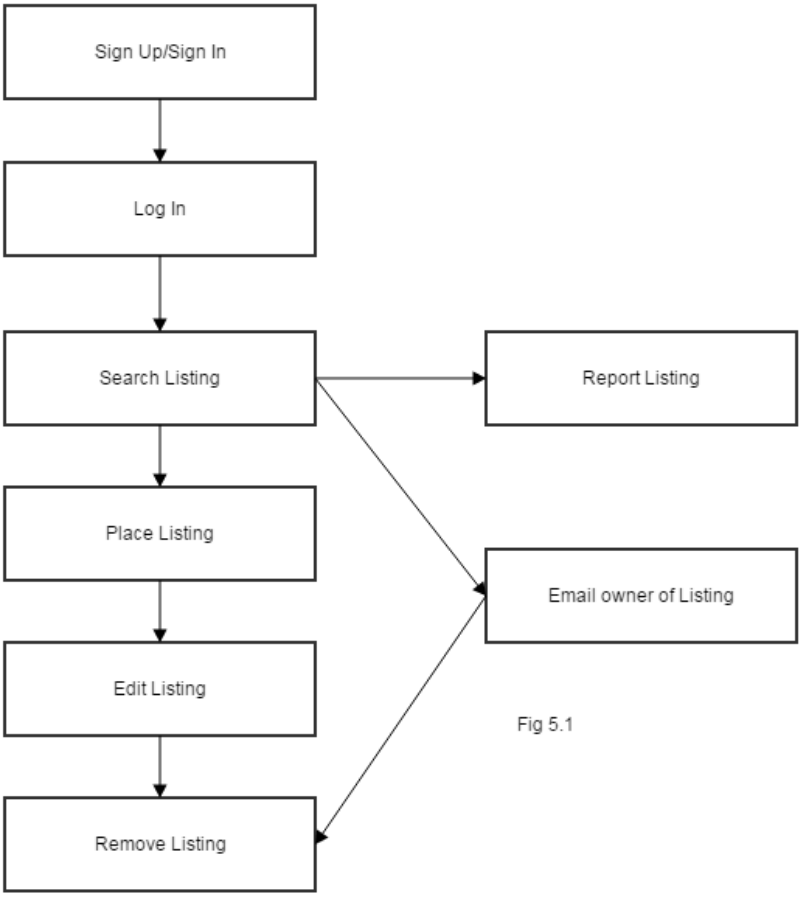


Fig 5.1

5.1 Figure One

- **5.1.1 Sign Up/Sign In:**
    - User signs up if not registered. If registered proceed to step two.
- **5.1.2 Log In:**
    - User logs into application

- **5.1.3 Search Listing**
    - User searches for a listing.
    - **5.1.3.1 Report Listing**
        - User reports a listing.
    - **5.1.3.2 Email Owner of Listing**
        - User contacts listing owner. Listing is removed.
- **5.1.4 Place Listing**
    - User places a listing giving required information.
- **5.1.5 Edit Listing**
    - User edits listing posted if required.
- **5.1.6 Remove Listing**
    - User removes listing marking that item was found or returned.

5.2 Figure Two

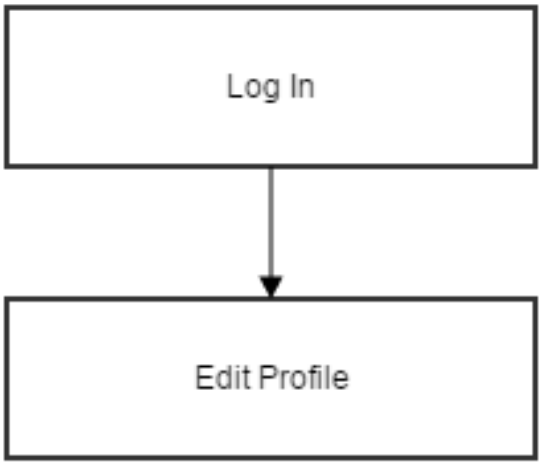


Fig 5.2

- 5.2.1 Log In
    - User logs in with credentials
- 5.2.2 Edit Profile
    - User edits profile with new information

6. Preliminary Schedule

6.1 Summary

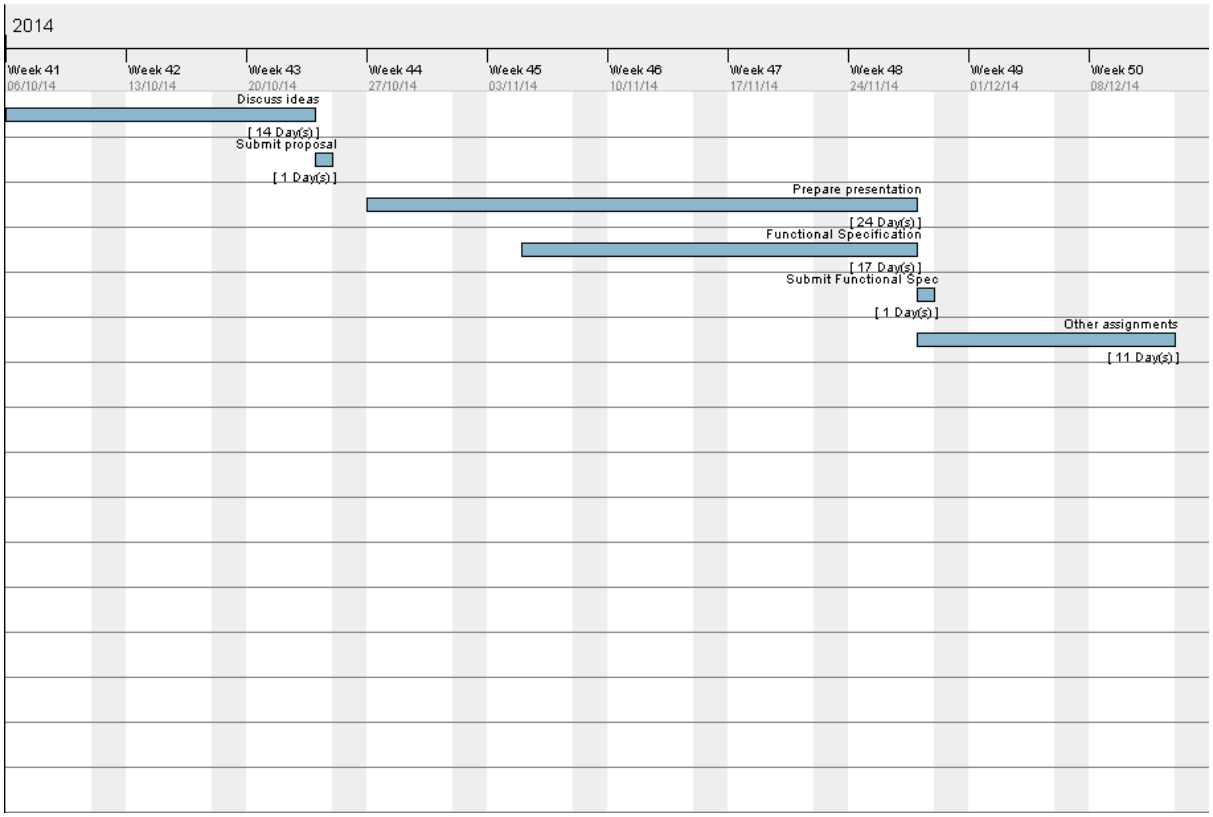
Here we have three GANTT charts showing the preliminary timeline. The tool used for creating this was GanttProject, a free project management tool.

Part One shows the time up to the end of semester showing due dates for the first submissions of the project.

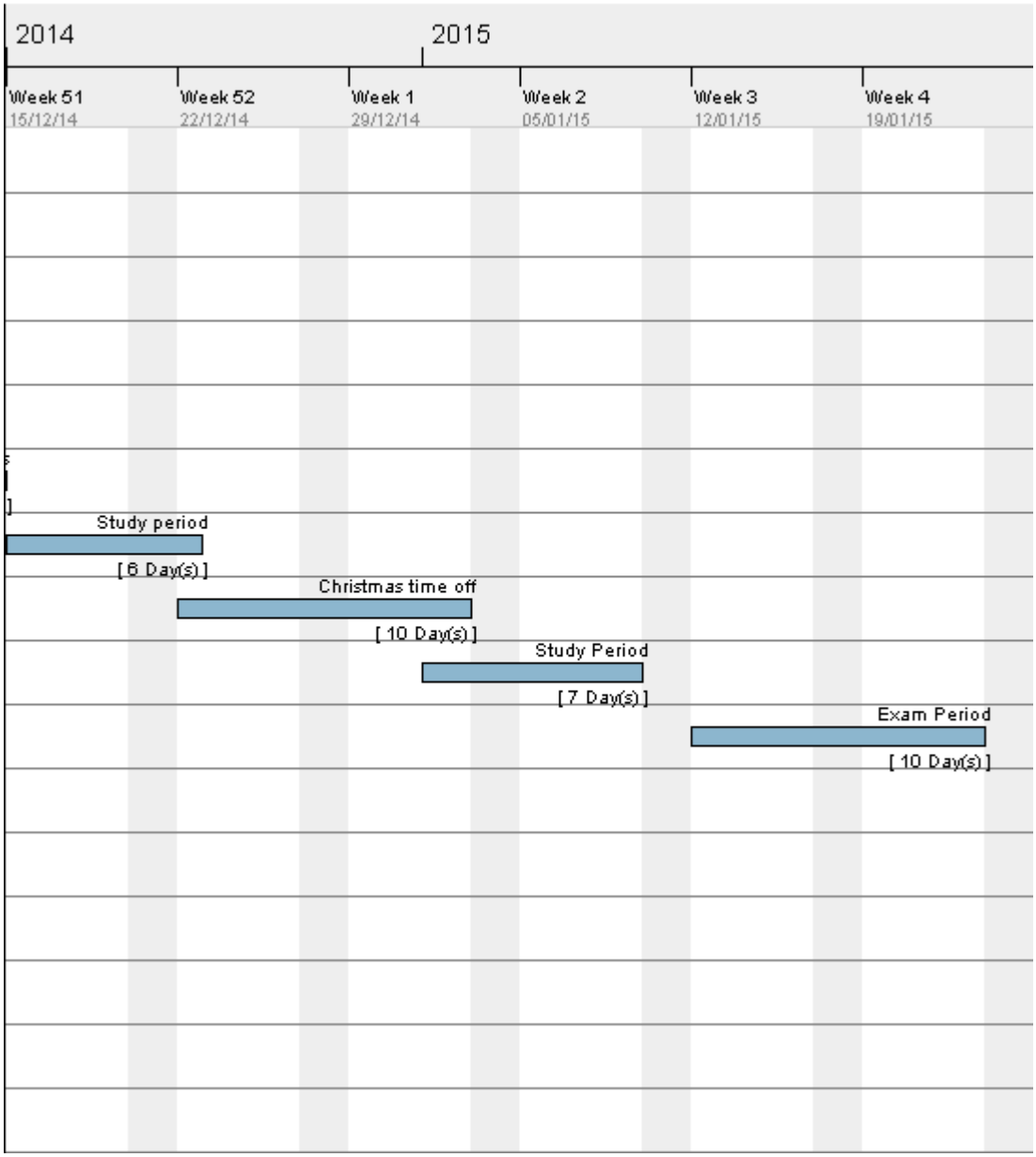
Part Two shows the time around christmas where we have study periods, holidays the exams. This time will be spent preparing for exams and discussing the project.

Part Three shows the project design and implementation timeline. This runs from after the exams until the due date. This time will be spent designing implementing testing and finalising our application.

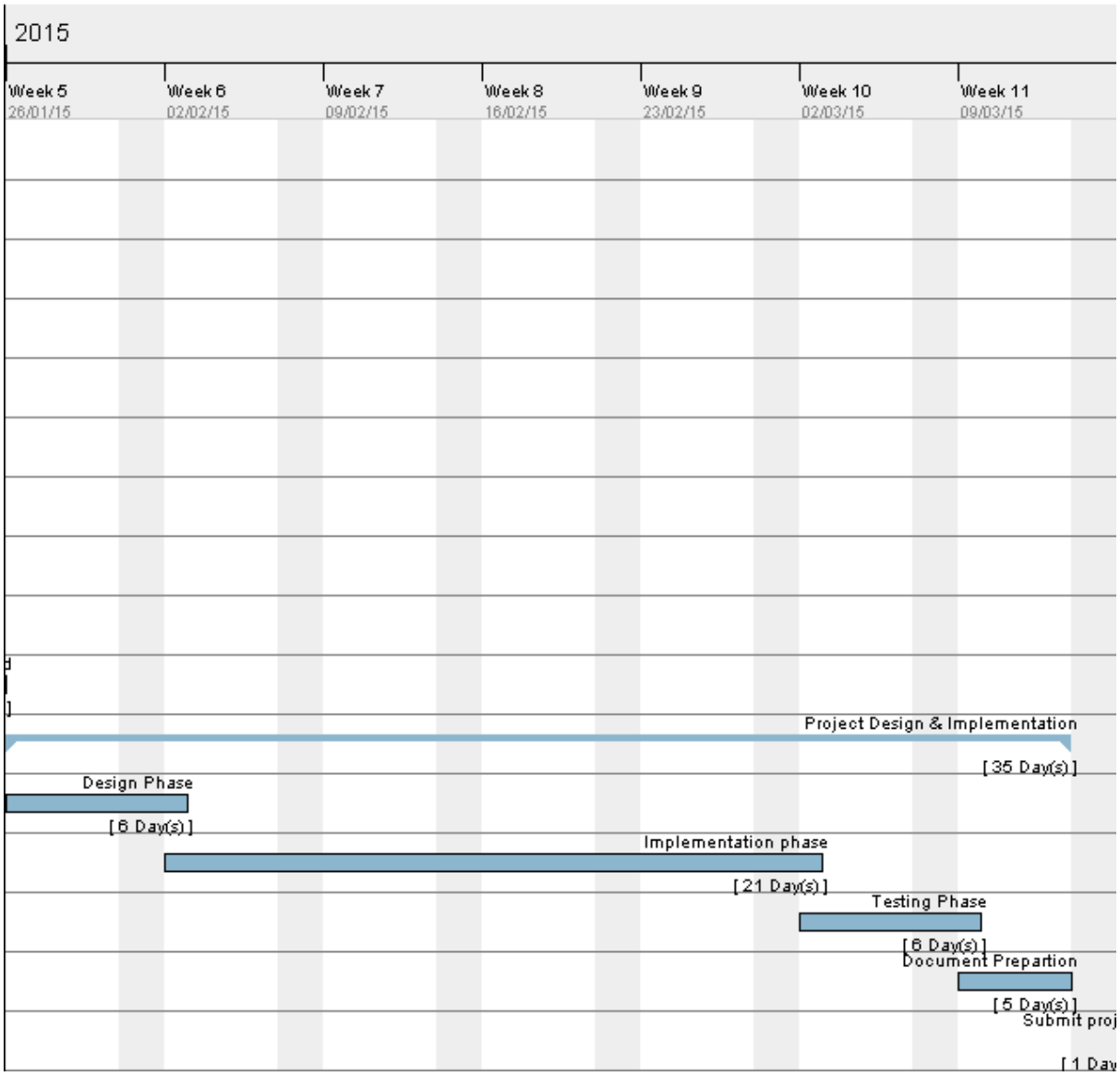
6.2 Part One


2014
Week 41
06/10/14
Week 42
13/10/14
Week 43
20/10/14
Week 44
27/10/14
Week 45
03/11/14
Week 46
10/11/14
Week 47
17/11/14
Week 48
24/11/14
Week 49
01/12/14
Week 50
08/12/14
Discuss ideas
[14 Day(s)]
Submit proposal
[1 Day(s)]
Prepare presentation
[24 Day(s)]
Functional Specification
[17 Day(s)]
Submit Functional Spec
[1 Day(s)]
Other assignments
[11 Day(s)]

6.3 Part Two


2014
2015
Week 51
15/12/14
Week 52
22/12/14
Week 1
29/12/14
Week 2
05/01/15
Week 3
12/01/15
Week 4
19/01/15
Study period
[6 Day(s)]
Christmas time off
[10 Day(s)]
Study Period
[7 Day(s)]
Exam Period
[10 Day(s)]

6.4 Part Three



7. Appendix

- REST - http://en.wikipedia.org/wiki/Representational_state_transfer
- Google Maps API - https://developers.google.com/maps/documentation/ios/
- Xcode - https://developer.apple.com/xcode/
- Objective C - http://en.wikipedia.org/wiki/Objective-C