# CS22120 Project Plan

Group 6

|  |  |  |  |
|---|---|---|---|
| **Author** | sjn3 | **Address** | Department of Computer Science<br>Aberystwyth University<br>Ceredigion<br>SY23 3DB |
| **Config Ref** | SE_06_PLAN_01 |  |  |
| **Date** | 20/10/2014 |  |  |
| **Version** | 1.0 |  |  |
| **Status** | Release | **Copyright** | © Group 06, Aberystwyth University |

# Table of Contents

I. **Introduction** ........ 3

Purpose of this Document

Scope

Objective

II. **Overview** ........ 3

Android

Website

Server

III. **Use Case** ........ 5

Use Case Diagram

Use Case Description

IV. **UI Design** ........ 7

Android

Website

V. **Gantt Chart** ........ 11

VI. **Risk Analysis** ........ 12

VII. **References** ........ 13

VIII. **Document History** ........ 13

# Introduction

## Purpose of this Document

This document will lay out the plan for the Reserve Plant Species Recording (RPSR) system using diagrams and detailed overviews.

## Scope

This plan will describe what will be done in this project at a high level. It will also show when each part shall be done. It shall not cover exactly how the project will be done as this will be included in the design documents.

This document should be read by the client to ensure that the requirements of the system are met.

## Objective

The objectives of this document are to:

- Provide an overview of the entire system including the tools used
- Present the interactions of the system in the structure of a use-case diagram
- Give an initial idea of what the basic user interface will look like
- Show the time frame in which each part of the project will be completed during
- Draw attention to any possible risks and provide solutions that can be used if they occur

# Overview

## Android

### *Platform*

The specification given to us by the client has specified that the system is to feature a mobile Android application, so we will be developing the app in Android.

### *High-Level Architecture*

#### *Signing In*

When opening the app it will ask the user to sign in, register or use offline. If the android device is connected to the internet at the time, it will make contact with the database in order to confirm details of the user's name and password. The register button will send the user to a new screen asking for the user’s details, and using offline will restrict the user so they will be unable to upload any records to the database.

#### *Creating Records*

Creating records of plant species is the primary use of the Android app. When creating records, it will require fields that will later be uploaded and stored in the database. This will include species names, the location of where the species was located, and more.

#### *Menu*

After signing into the app, the user will be sent to a main menu where they will be able to navigate around the rest of the app. The menu buttons will be:

- New Record
- New Species
- View Records
- Go Online
- Exit

#### Syncing with the Database

In order to get data from recordings made by users on android devices, when a WiFi signal is found, the app will attempt to connect with the database in the background in order to upload records to the database. Until a WiFi signal is found, all recordings and data will be stored locally on the device.

### *Description of Target Users*

Users of the Android app will generally be botanists visiting sites and using company tablets to record species at said sites. They will be familiar with standard computer interfaces, so we will be keeping the interface simple and easy to use to ensure that the users are not worried about how to use the app.

## Website

### *Platform*

The website we will be developing in conjunction with the Android app will be compatible with pcs, macs, the most recent web browsers, tablets and other mobile devices.

### *High-Level Architecture*

#### Bootstrap

We will be using a bootstrap to help us develop a website that works well with mobile phone browsing as well as desktop, as it supplies and updates CSS and JS stylesheets. It has a good functionality with older browsers, and with the features it offers us, it seems like the only sensible way to develop the website we require.

#### PHP

The PHP that will be used in the website will be used to lookup details stored by the user. It is the language that will be used to communicate with the server when needed to gain access to the database, and for other tasks such as getting login details for the user and for database searches for data.

#### Location Production

We will be accessing a GIS lookup, one of the most widely used location data storage ideas, alongside the Google Maps API to produce a map for the website. It will use the database location lookup to emulate the location of records when looked up.

#### User Login

Using server based commands, user login data will be stored server side so that users can access their own uploads and can edit them at their discretion through their own account, as well as being able to look at other records in the database. Admins will also be able to access database records and edit them, and their credentials will stored server side along with the user details.

### *Description of Target Users*

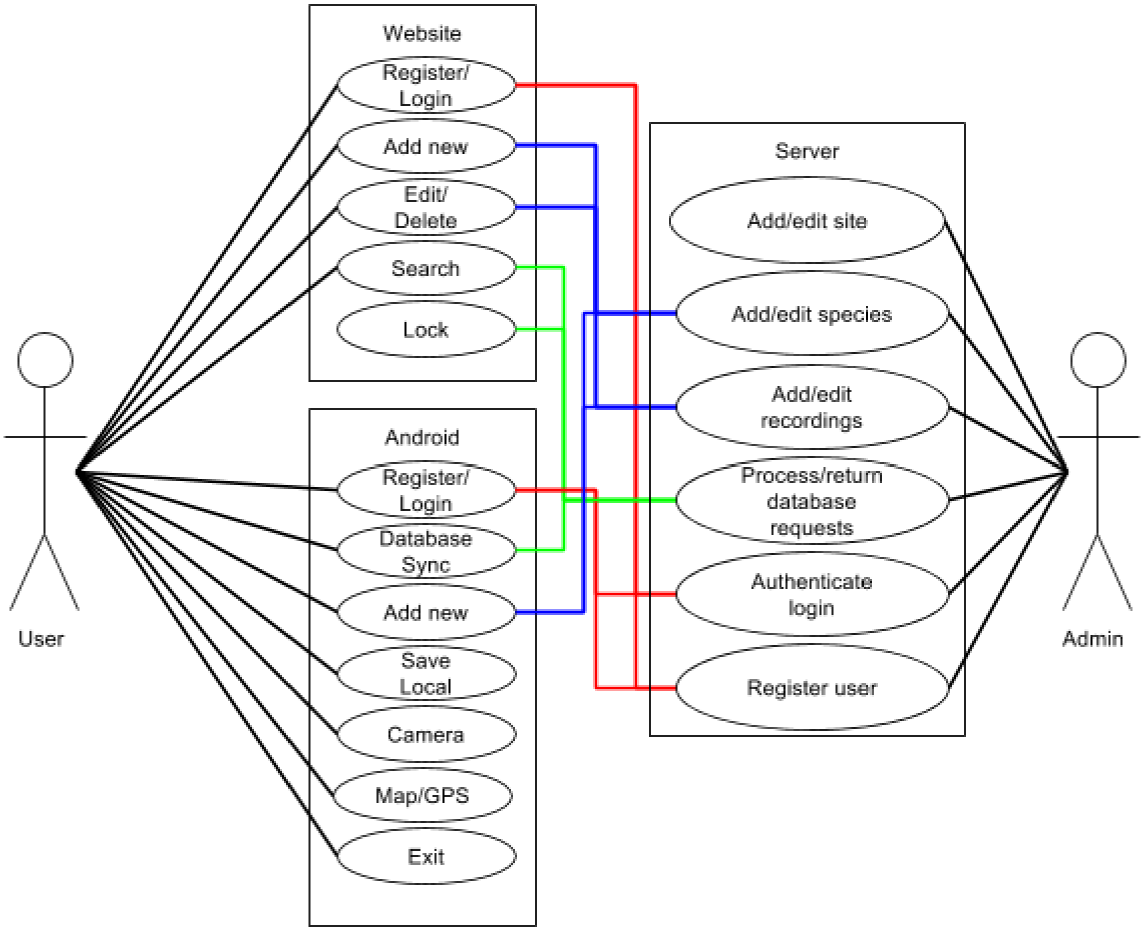
Both users of the android app and other naturalists working for RPSR will be using the website in order to access the database that will be filled with data from the app. The user interface will be similar to the app, to ensure the ease of use between the two systems.

## Server

To implement the server, the Java programming language will be used. We can take advantage of Java servlets which are used to extend the capabilities of a server. MySQL will function as our database which will store all our information. Java servlets will deal with communicating with the database by parsing HTTP requests.

# Use Case

## Use Case Diagram



## Use Case Description

| **Android** | |
|---|---|
| Register/Login | Users will be able to register onto the system to start recording data, and the details |
| Database Sync | The device, when detecting a WiFi signal, will attempt to connect to the database in order to sync recordings that have been saved locally |
| Add new | Begins the process to create a new recording or species for the database, where more details about either are entered |
| Save Local | Saves any records created locally on the device so that it can be synced to the database at a later date. |
| Camera | Allows the user to take pictures to later use in recording data |
| Map/GPS | The device will use this automatically in order to pinpoint where abouts the recording is taken |
| Exit | Exits and closes the application, and closes any database connections there are |

| **Website** | |
|---|---|
| Register/Login | Users will be able to register or login with the same details as the Android app, as they will be using details from the same database |
| Add new | Users will be able to add new species or recording in a similar way to how the app works |
| Edit/Delete | Users will be able to edit or delete records they have created via the website. Admins will be able to edit or delete any record in the database |
| Search | Users will be able to search the database for records found within the database |
| Lock | Admins will be able to lock the database if maintenance is needed so that users do not enter new recordings while the database is updated |

| **Server** | |
|---|---|
| Add/edit site | Takes data about a site and either adds the site to the database if it doesn't already exist or updates the existing site's stored data. |
| Add/edit species | Takes data about a species and either add the species to the database if it doesn't already exist or updates the existing species' stored data. |
| Add/edit recordings | Takes data about a recording and stores it in the database. |
| Process/return database requests | Receives a request from the android app or website for items from the database and then sends that data back |
| Authenticate login | Receives login data from the android app or website and decides if that user is a valid. If so it authenticates the login and informs the app/website of this. |
| Register user | Receives user registration data and creates a new user in the database |

*Note: The server completes all use cases automatically, but the admin can also complete these actions through the server separately to influence the database and other sections of the system.*

# UI Design

## Android

### *User Navigation*

Figure 1 (below) is a flow chart which describes how the screens of the app are navigated. The arrows show how the user can navigate through the program

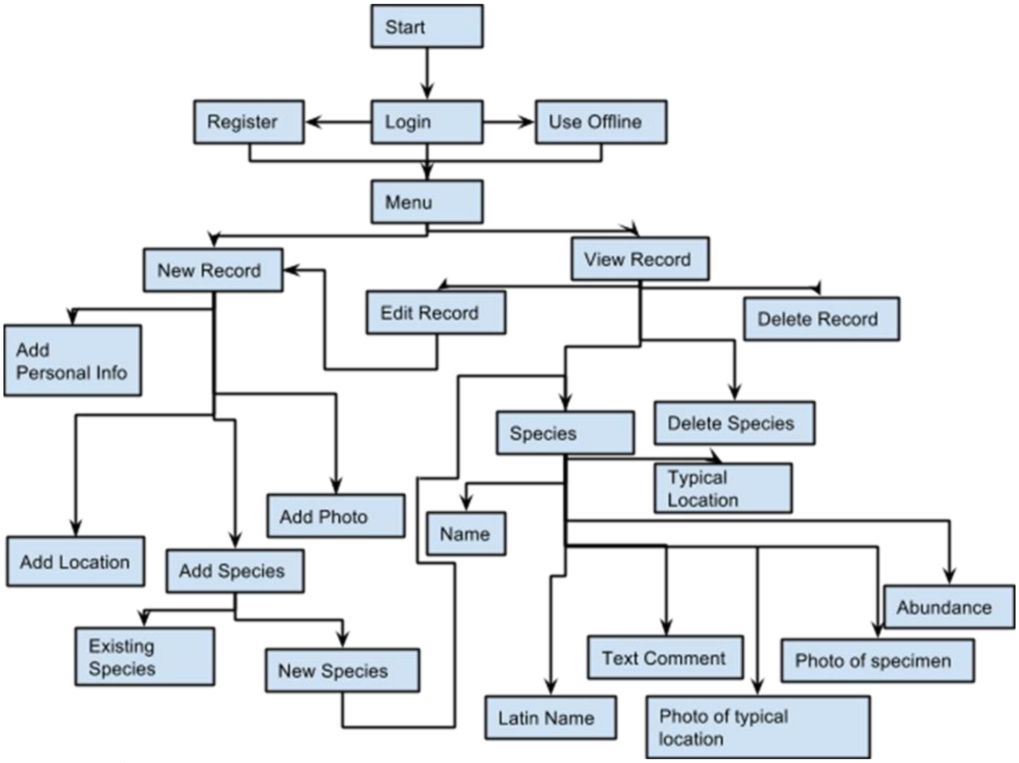


Figure 1: A flow chart of the app

### *GUI Design*

The diagrams contained in section 2 are the initial designs for the programs GUI. They are simple visual representations of the screens of the Android application. This is to give a rough idea of what the screens will look like but they are not a final design

#### Login Screen

Figure 2 (right) gives a rough idea of the initial tasks the user can do once they have started up the app. For the user to be able to log in, they must already have registered either on the app or on the website, and have an internet connection.

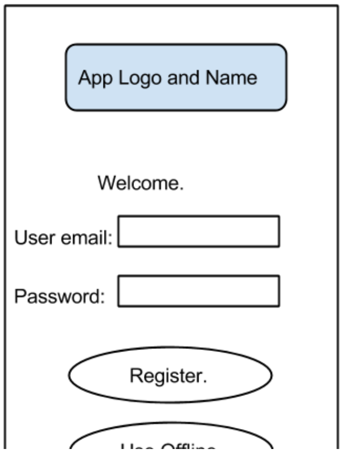


Figure 2: The login screen

### Register Screen

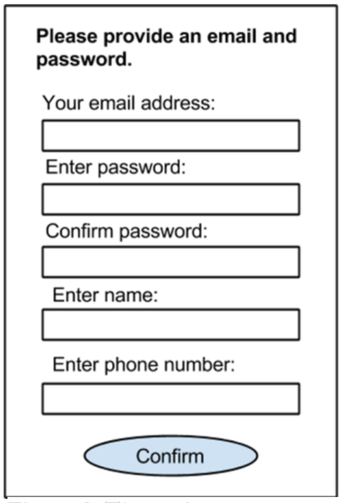

Please provide an email and password.

Figure 3: The register screen

The register screen requires the user to provide an email address and a password to be attached to the account. Additional information about the account holder can be provided on the website later.

### Menu Screen

The menu screen houses the functionality to add new records and species and also to edit pre-existing ones via a couple buttons. This page is also the main 'hub' as the menu button takes the user back to this page. There is also an option to login that becomes available if the user logged in offline.

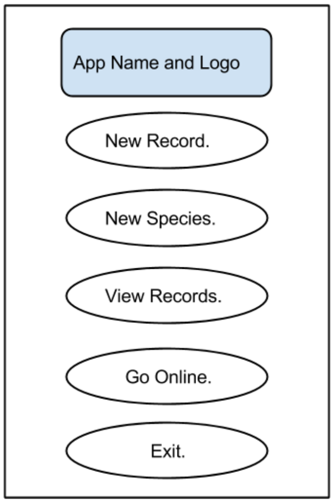


Figure 4: The menu screen

From the menu screen the user can also exit the program using the exit button at the bottom of the screen. This brings up a popup that asks the user if they are sure they want to quit.

As this is the main screen there is no menu bar at the bottom as all the elements in the menu bar are displayed on the screen to begin with. However on other screen the menu bar would be present.

### New/Edit Record Screen

This screen has a base form but depending upon if you are editing a pre-existing record or if you are adding a new record some of the fields may already be filled out with data. The user needs to input most of the data concerning the record, while the app will fill out the time and date for the user.

The user must input their email into the record so that the database knows who made the record when the app syncs with it.

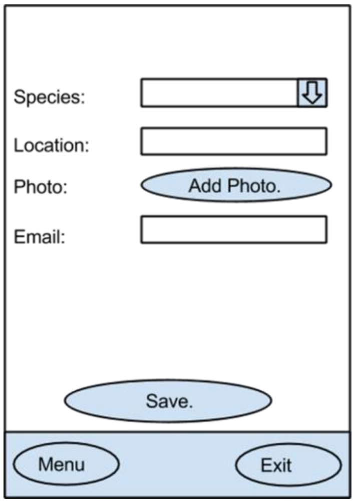


Figure 5: The new/edit record screen

### View Record Screen

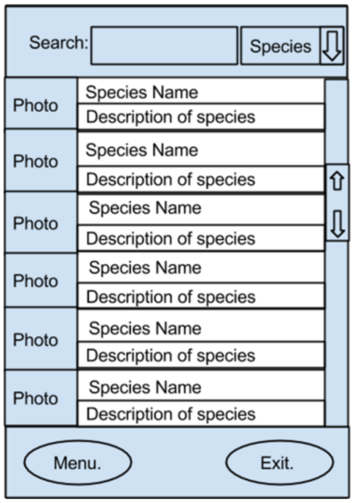


Figure 6: Species search screen

From the view record screen the user is able to scroll through existing records of species and view the individual records for each of those species to see where is has been recorded before. The user is also able to remove records either whole recording sessions or individual records for one species that they have made. Also able for viewing are other peoples records.

From here the user can do multiple things such as look through their recordings and records that have been saved and also look at the records of species that are on the database (Figure 6) and see the general information about specific species (Figure 7).

Clicking on one of the species from the list will take the user to the page about that species. On Figure 7, if the back button is pressed on the user is taken back to the view records screen.

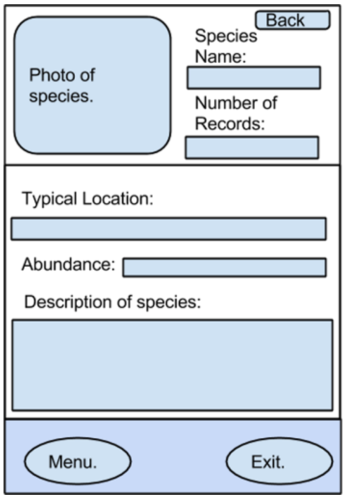


Figure 7: Individual species screen

### New Species Screen

Figure 8 provides an idea of what adding a new species via the android app would look like. The user would be required to add in required fields but would also be able to add in some of the more unknown data such as the Latin name of the plant later on, either through the website or via editing the species record on the app. The required fields would be the common name, a photo and the typical location of the plant. Other data like the common name and the abundance level could be added later on if the user did not know them.

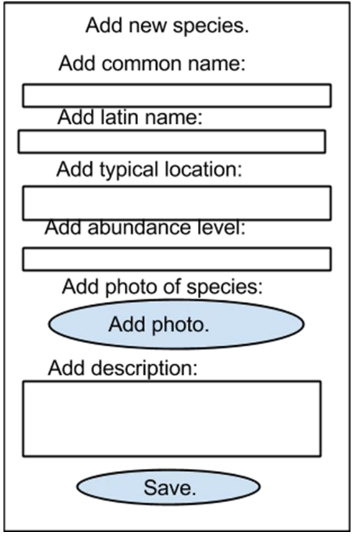
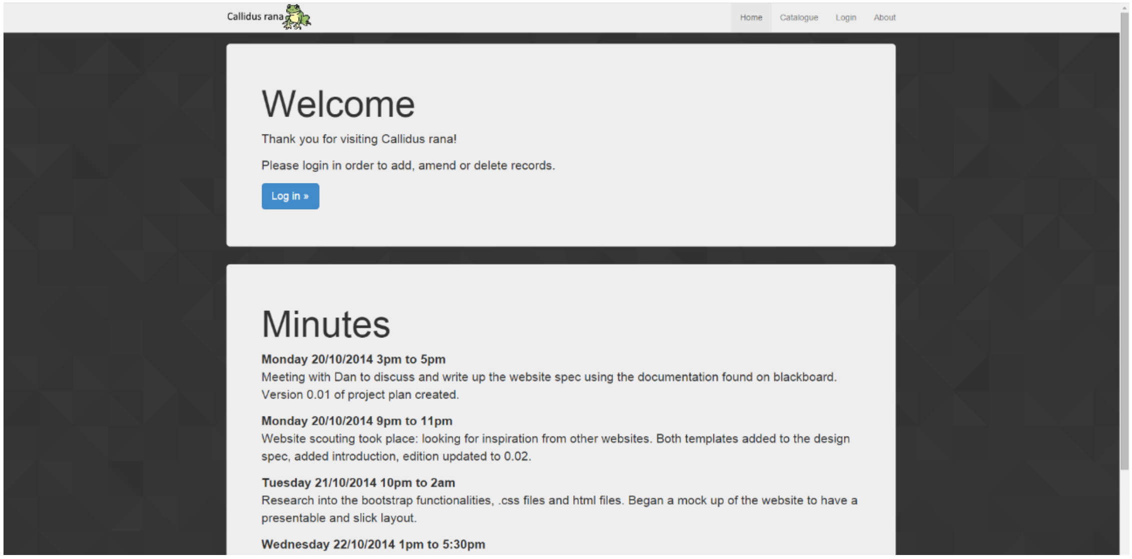


Figure 8: New species screen

## Website



Welcome

Thank you for visiting Callidus rana!

Please login in order to add, amend or delete records.

Minutes

**Monday 20/10/2014 3pm to 5pm**
Meeting with Dan to discuss and write up the website spec using the documentation found on blackboard. Version 0.01 of project plan created.

**Monday 20/10/2014 9pm to 11pm**
Website scouting took place: looking for inspiration from other websites. Both templates added to the design spec, added introduction, edition updated to 0.02.

**Tuesday 21/10/2014 10pm to 2am**
Research into the bootstrap functionalities, .css files and html files. Began a mock up of the website to have a presentable and slick layout.

**Wednesday 22/10/2014 1pm to 5:30pm**

Figure 9: Website layout

In the nav bar, we will have a logo in the top left of the screen, and in the right hand side of the nav bar, there are buttons for Home, Catalogue, Login and About. The home button takes you back to the main page of the website with the welcome screen. The Catalogue button will take the user to the page where they will be able to search, add, delete and edit data from the database. The Login button will let the user log in to their personal account. The About button takes the user to the page that contains disclaimers and information about the site.

The segregated sections are to help portray each section of the website, so the user can easily tell when there is new information on the website. The background is displayed in the way shown in Figure 9 to make the website easy to read

# Gantt Chart

# Risk Analysis

| Event | Likelihood | Severity | Calculated Risk | Mitigation |
|---|---|---|---|---|
| **Falling behind schedule** | **0.5** | **0.8** | **0.4** | Good communication of status between team members. Have a thorough and complete plan and reschedule if falling behind. |
| **Programmers too ill to work** | 0.7 | 0.4 | 0.28 | Ill members are to let the other group members know as soon as illness becomes apparent so that their responsibilities can be shared out among remaining programmers |
| **Laptop/PC malfunctions** | 0.3 | 0.6 | 0.18 | Work is to be backup to regularly to Google Docs and GitHub, and work can be continued in computer rooms on campus until laptop/pc is fixed |
| **Persistent bugs** | 0.7 | 0.7 | 0.49 | Team members are to ask the group for advice on the bug in order to prevent it becoming persistent. If the bug cannot be solved in a reasonable time frame, a workaround must be found |
| **Implementation difficulties** | 0.4 | 0.9 | 0.36 | A thorough plan should prevent most implementation difficulties. Explorative programming before timetabled coding is needed for difficult sections of code. If difficulties occur despite planning, sidelining or dropping extra features may be necessary |
| **Client changing specification during development** | 0.4 | 0.8 | 0.32 | A group meeting will be arranged as soon as possible to bring everyone up to speed with the new requirements for the system. |
| **Connection issues with GitHub** | 0.1 | 1.0 | 0.10 | Calculate whether the problem is to do with GitHub or with our own computers. In both cases, inform the rest of the group with the issue and communicate effectively what things everyone is working on. |

# References

N/A

# Document History

| **Version** | **CCF No.** | **Date** | **Changes made to document** | **Changed by** |
|---|---|---|---|---|
| 0.1 | N/A | 2014/10/20 | Initial design of document | sjn3 |
| 1.0 | N/A | 2014/10/28 | Final document | sjn3 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |