

Wits Software Development Group 2013-2014

WitsMapapp Work Documentation

Version 1.0

Presented by:

Jason Chalom

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| 1.0 | March 2014 | JMC | Added table of work, short descriptions of each component and table of contents. |

Contents

[Version 2](#_Toc383999887)

[Date 2](#_Toc383999888)

[Author 2](#_Toc383999889)

[Description 2](#_Toc383999890)

[Introduction 2](#_Toc383999891)

[Project Overview 2](#_Toc383999892)

[Project Objectives 2](#_Toc383999893)

[Success Criteria 2](#_Toc383999894)

[Project Team 2](#_Toc383999895)

[Note 3](#_Toc383999896)

[Table Of Work 3](#_Toc383999897)

[Project Flow Diagram 3](#_Toc383999898)

[Components 4](#_Toc383999899)

[Server Side Web UI 4](#_Toc383999900)

[Login Page 4](#_Toc383999901)

[First Time Login Page 4](#_Toc383999902)

[User Administration 5](#_Toc383999903)

[Campus Input Page 5](#_Toc383999904)

[Campus Administration 5](#_Toc383999905)

[Buildings Input Page Design 5](#_Toc383999906)

[Buildings Administration 5](#_Toc383999907)

[Database 5](#_Toc383999908)

[Users TBL 5](#_Toc383999909)

[Campus Tbl 5](#_Toc383999910)

[Buildings tbl 6](#_Toc383999911)

[API 6](#_Toc383999912)

[GET Campus 6](#_Toc383999913)

[GET Campus list 6](#_Toc383999914)

[Get Building 6](#_Toc383999915)

[GET Buildings List 6](#_Toc383999916)

[POST search criteria 7](#_Toc383999917)

[Error responses 7](#_Toc383999918)

[Mobile App 7](#_Toc383999919)

[Welcome Page 7](#_Toc383999920)

[Second Page 7](#_Toc383999921)

[Third Page 8](#_Toc383999922)

# Introduction

## Project Overview

This is the University of the Witwatersrand's Software Development Student group’s 2013 (and perhaps going into 2014) project to build a mobile application for Wits Marketing.

This app will allow the user to select on a map where they would like to go and from their current position to their destination it will calculate and display the shortest path using the Google API.

It will also display information about the campus they are on and the building they are going to. Both a description and an image will be displayed respectively. The ability to display more images may be added later.

## Project Objectives

* Create server to support the input of custom data such as campus locations, descriptions and images.
* Have a database to store information
* Have the input system be user password protected from unauthorized access
* Have a functional and well-built mobile app based on the Phonegap framework and JavaScript libraries

## Success Criteria

* A functional system which can tell where the user is from their mobile device
* Have a store of data about locations on campuses
* Ability to draw paths to locations, perhaps even have dynamically updateable paths.
* Have a server side administration web-app which is functional.
* All user interfaces have Wits colours and logo.

## Project Team

* Jason Chalom (@TRex22 on GitHub)
* Isaac Seshoka (@ick-seshoka on GitHub)

# Note

Go to the specification documentation for reference in terms of what is needed. This document is to compartmentalise the parts of the project and then divide the work equally.

# Table Of Work

|  |  |  |
| --- | --- | --- |
| **Components** | **Description** | **Time Estimate** |
| **Isaac Seshoka** | | **Total:** |
| Server Side Web UI | Work on the design of the web UI pages, the placement of required components and the PHP behind them |  |
| API | Work on the PHP to get the data from the database and the return it as a JSON object. |  |
| Mobile APP | Work on the design in terms of the specification. |  |
| **Jason Chalom** | | **Total:** |
| Database | Create a database which follows the specification | 1 hr |
| Google Maps API Components and other important components. | Create the components used in both the Server Side Web UI where the Google Map API is called to generate polygons and GPS coordinates and also in the Mobile APP for directions and maps. NB on getting zoom to work from old code or through Google. Work on getting the JavaScript which sends API calls, and receives JSON object data to inject into the server-side UI pages where needed. |  |
| Mobile App Tech | Work on getting the JavaScript which sends API calls, and receives JSON object data to inject into the mobile app. Similar JavaScript is used in the input page. |  |
| Tech Demo | Build a tech demo where parts can be used in the final project which shows a ‘functional’ mobile app which just shows the technology working without any real data or design in place so as to refine the technology. |  |
| Total Time: | | |

# Project Flow Diagram



# Components

## Server Side Web UI

Most the web pages will be included in the ‘input page project component’. (See the SpecificationDesigns.pdf document)

We already have part of the input page done. However that are parts that have to change to fit with the new specifications.

The Wits colours and logo must be a part of the user interfaces. There should also be a way to navigate to most of the web interfaces.

### Login Page

The login page will be basic in design (See the SpecificationDesigns.pdf document). It will most likely come from what we already have. The Wits colours and logo must be a part of this page.

### First Time Login Page

There needs to be a page to initially create an administrator account (See the SpecificationDesigns.pdf document).

This page should only appear on a fresh database.

### User Administration

This page is used to display all users in the system, edit user information and also delete users.

### Campus Input Page

This page is where Campus information is inputted, it will use a Google Maps object to help locate a square which demarcates the campus selected. The user will ‘draw’ a rectangle around the area they wish to demarcate.

It will upload an image (Maybe multiple) of the campus and also a description of the campus.

### Campus Administration

This page will display a list of campuses in the system and will be able to delete a campus or edit its information.

### Buildings Input Page Design

This is the page buildings will be inputted onto. There will be a dropdown list of campuses currently in the system to select from. If there is no campus then the page should redirect to the Campus Input page. It will also use a Google Maps object to find the GPS coordinates of the building on the specific campus. By selecting a campus from the dropdown the map will be populated at that campus. The user will then create a point on top of a building to select it. The page will then upload the building information to the database.

### Buildings Administration

This page will display a list of buildings on a specific campus in the system and will be able to delete a campus or edit its information.

## Database

### Users TBL

* UserId (Primary Key) Autonumber
* Username (String)
* PasswordHash (String)
* Email (String)

### Campus Tbl

* CampusID (Primary Key) Autonumber
* GPSLat1 (String)
* GPSLong1 (String)
* GPSLat2 (String)
* GPSLong2 (String)
* GPSLat3 (String)
* GPSLong3 (String)
* GPSLat4 (String)
* GPSLong4 (String)
* Description (String)
* ImageLocation (String)

### Buildings tbl

* BuildingID (String)
* CampusID (Foreign Key from Campus table)
* GPSLat (String)
* GPSLong (String)
* Description (String)
* ImageLocation (String)
* BuildingAbbreviations (String) There can be a few of them which will then be delimited.

## API

The API is the main part of the system as it connects the server to the mobile app. All it will return are JSON objects with HTTP headers (Auto-handled) and the data. {v1} refers to the version as in keeping with RESTful API design. It will probably just be v1 as no other version is planned for now. The {} after the /object/ are the variables which are sent to the API to return the data required. The Google API will be connected directly from the server side or mobile-app side as it will be faster. Our API is just to get the internal data from the database.

### GET Campus

~/api/{v1}/campuses/{CampusID}

This will return a JSON object of the specific campus its information and the http header.

### GET Campus list

~/api/{v1}/campuses/

This will return a JSON list (IEnumerable object) of campuses in the system including the http response header.

### Get Building

~/api/{v1}/buildings/{BuildingID}

This will return a JSON object of the specific building its information and the http header.

### GET Buildings List

~/api/{v1}/buildings/

This will return a JSON list (IEnumerable object) of buildings in the system including the http response header.

### POST search criteria

~/api/{v1}/search/{query}

This will return a JSON object of the campus and building or an error not found (Error 404 Bad Request in the http response header)

I’m not sure if this is needed.

### Error responses

Only a few specific errors are required:

HTTP 404 Not Found

HTTP 400 Bad Request

HTTP 500 Internal Server Error

HTTP 500 Internal Server Error (With custom message), Google API malfunction

HTTP 200 OK

There should be custom internal error messages for the user and developer depending on the specific problem. (Unless it’s a 200, or a standard 404/400)

## Mobile App

The mobile app will have three swipe-able pages.

I have an idea with the technology which creates the three swipe-able pages as a dynamically loaded while being quite open to design. However the design must not be too memory heavy or cluttered as to make the program either hard to use or slow to use.

### Welcome Page

This page will have a campus dropdown menu which when selected will show a building dropdown menu so that the user can select a building which they wish to go to.

There will also be a search bar so that the user can search using a room number. The API will handle calculating the building from its abbreviation and the floor from the first number. (This will have to be checked in the database design and also the building input page.)

By clicking a next arrow, swiping left or clicking a button the program will bring up the second page.

### Second Page

The second page will bring up a map, with the current location of the user and their destination drawn for them. It will also plot a course for them to take to get there. (There may need to be a recalculate button to recalculate the path or to have an auto-refresh ability). The Google Directions API could be used. <https://developers.google.com/maps/documentation/directions/>

### Third Page

This page will display the information on the current campus and also the selected building. It will have images of the two and the abbreviations of the building as well. (Some information may already exist in the Google Map).