HTML5 & JavaScript Programming Project

The Prawns Team

UWS, Paisley Campus

HTML5 & JavaScript Programming

Contents

[Team Members 2](#_Toc497342508)

[Member Roles 2](#_Toc497342509)

[UI Design Prototypes 2](#_Toc497342510)

[Photoshop UI Design 2](#_Toc497342511)

[HTML Prototype 3](#_Toc497342512)

[Design Document 5](#_Toc497342513)

[Project Summary 5](#_Toc497342514)

[Specification 5](#_Toc497342515)

[Use Case Diagrams 5](#_Toc497342516)

[Use Case Descriptions 5](#_Toc497342517)

[Flowcharts 7](#_Toc497342518)

[Object Types and Methods 7](#_Toc497342519)

[Contacts Type 7](#_Toc497342520)

[Login Type 8](#_Toc497342521)

[Application Components 8](#_Toc497342522)

[Object Collections 8](#_Toc497342523)

[Application-wide Data 8](#_Toc497342524)

[General Purpose Functions 8](#_Toc497342525)

[Event Handlers 8](#_Toc497342526)

[Testing 8](#_Toc497342527)

[Project Features 9](#_Toc497342528)

# Team Members

Banner ID: B00330023

Banner ID: B00330925

Banner ID: B00311466

Banner ID: B00330156

Banner ID: B00334101

## Member Roles

Programmers– B00330156, B00334101, B00330925, B00311466

Designer – B00334101, B00330023

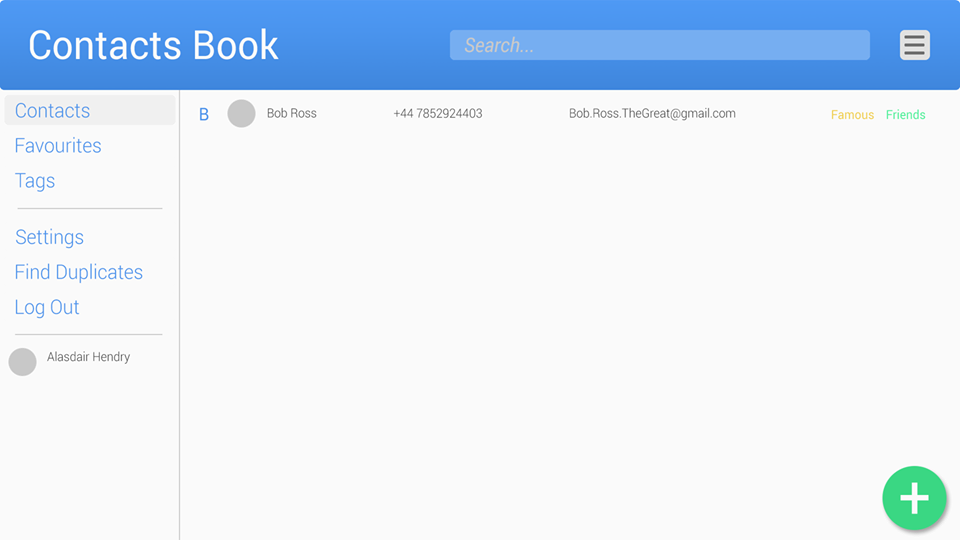
Documentation – B00330925, B00330156

Testing – B00334101, B00330023, B00330925, B00330156

Project Manager – B00330023

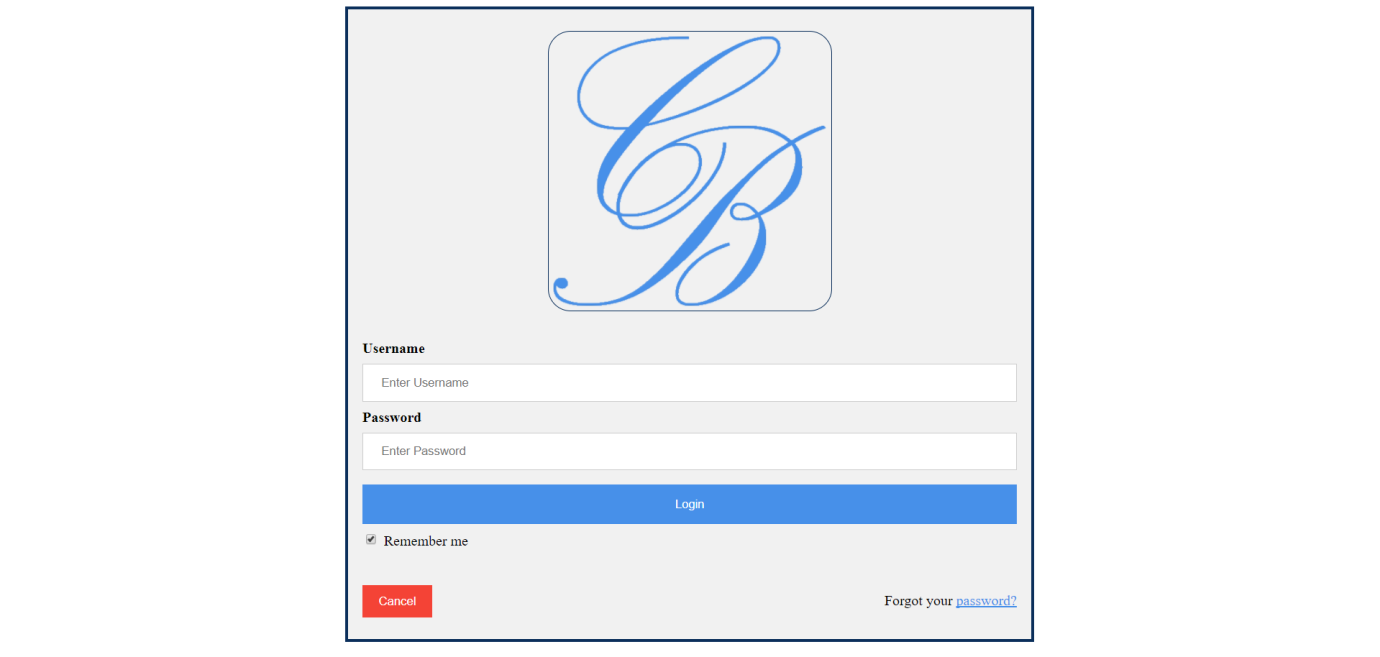
# UI Design Prototypes

## Photoshop UI Design

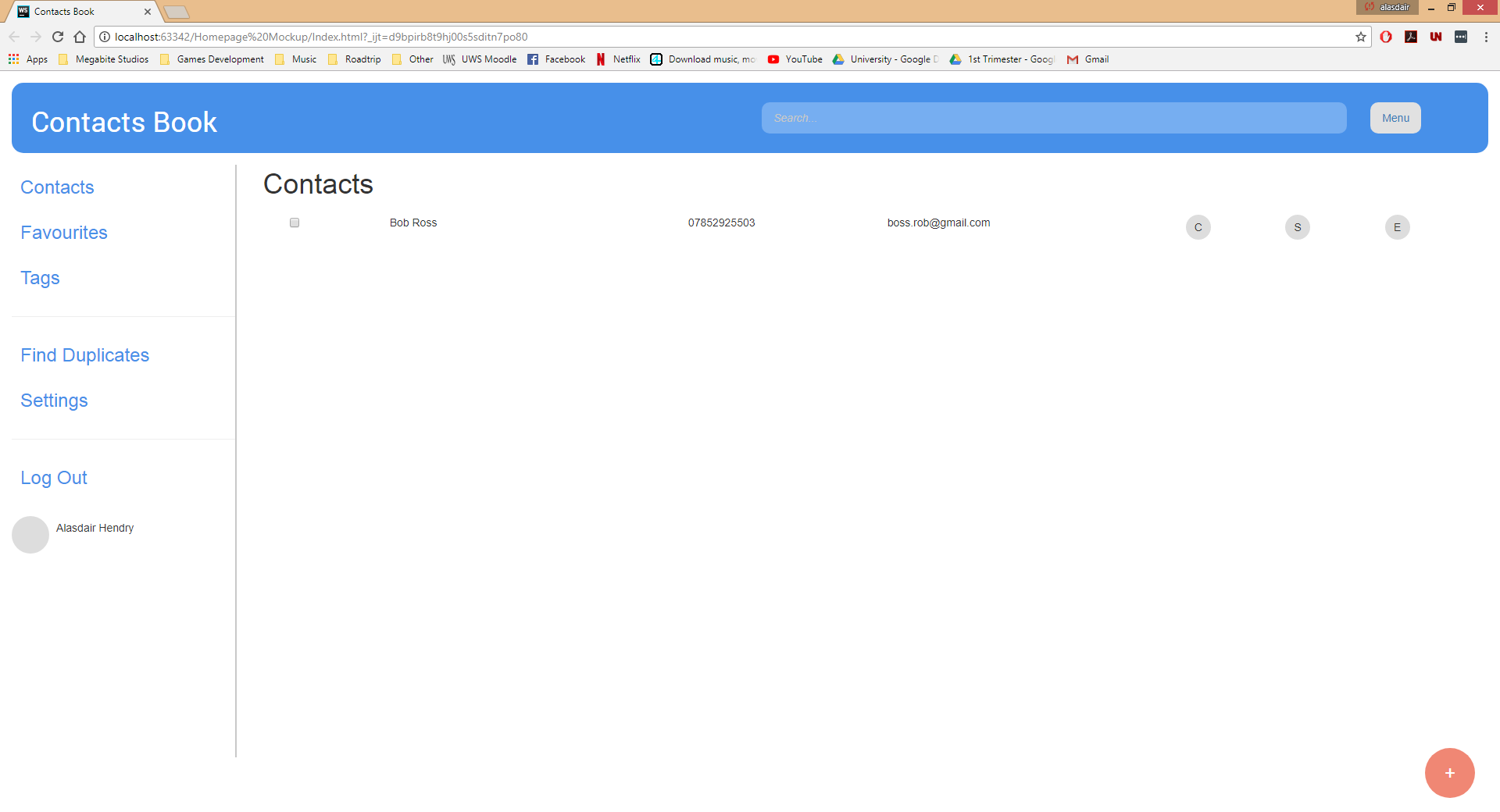


A mock-up of the UI for the Contacts Book was created in Photoshop in order to give a clearer idea of how the web-app would look and will be used as a template for creating the HTML page.

## HTML Prototype

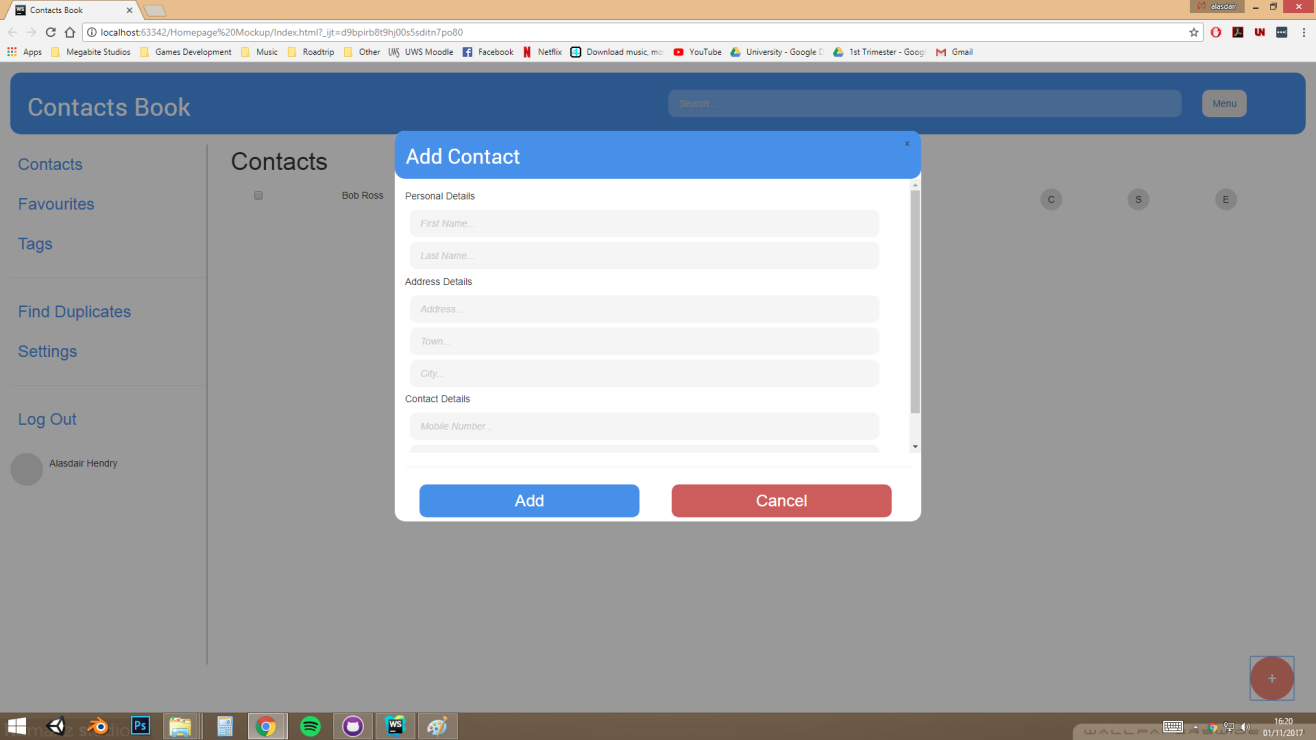


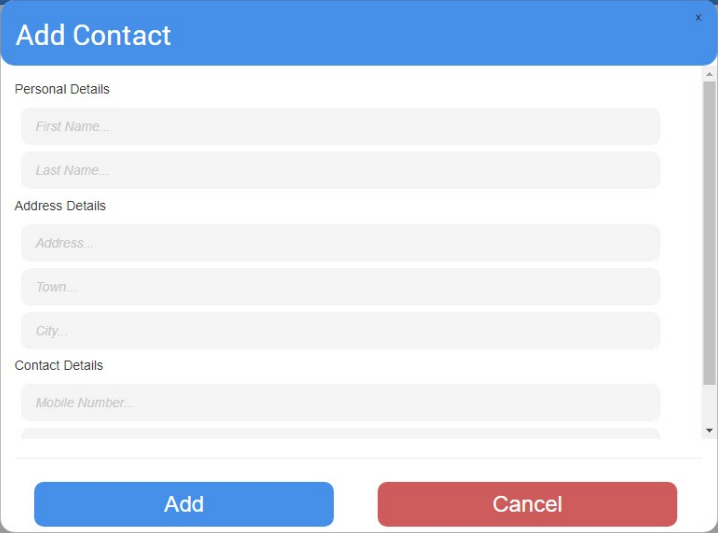
The team has created a Login page that will be displayed to the user before they are able to view their Contacts Book. The user will be required to create an account before they can begin using the Contacts Book and will have options such as ‘remember me’ so that they will not have to log-in every time they use the app, as well as being able to reset their password.



We tried to stick as close as we could to the mock-up design as possible in order to avoid making any errors while developing the web page. This will allow us to make sure the project goes to plan and avoid making any mistakes due to confusion.

Once the user has logged in to the app there will be several features they can use. On the left hand side the menu will be displayed, where they can: search contacts, create new contacts, add contacts to favourites or groups, check for duplicates, change settings, and log out of the app.





To create a new contact the user clicks on the red plus icon in the bottom right of the contacts page, this will then bring up a pop up window. In this window the user can fill in the information needed for any new contacts. The team designed the window to be easily understandable even to those who are not accustomed to working with web pages. The ability to have the contacts add window on the same page as the contacts book page allows users to navigate the page clearly and if they accidentally click on the icon they can just cancel out and not have to wait on a separate page loading.

# Design Document

## Project Summary

The team aims to build a web page that will be used as a contacts book, it shall be similar to those found in applications such as Apple Contacts & Gmail Contacts. The features should include basic details (name, home/work contact numbers, work/home emails, address, birthday marital status, birthday and any other notes). The web page should be able to create a text version of this information that would be able to be copy & pasted into an email. It should also be possible to send emails to multiple contacts at once. A reminder should appear to the user the day before a contact’s birthday. Local storage shall be used to store the data. We also aim to implement Geo-Location in order to display where the contacts work/live based on postcodes and addresses and display on a map.

## Specification

* Take in contact details
  + Take in contact name
  + Take in contact mobile and home phone number
  + Take in contact home address Lines & post code
  + Take in contact work address Lines & post code
  + Take in work/home e-mail addresses
  + Take in contact birthday
  + Take in contact marital status
  + Take in other notes
* Make use of local storage to save contact details
* Create a text version of the contact’s email address to use to send an email
* Be able to send emails to multiple contacts
* Use addresses to mark contact work/homes with geo-location
* Display contact work/homes on a map

## Use Case Diagrams

Use Visio to create Use Case Diagrams

## Use Case Descriptions

|  |  |
| --- | --- |
| **Name** | Create Contact |
| **Description** | User will want to create and save contacts and details to the contacts book so they won’t have to type in their email every time they want to get in touch with them, with this they can simply select the contact they wish to email/phone and get their details. |
| **Triggers** | The user clicks the create new contact button. |
| **Actors** | The user of the app. |
| **Preconditions** | A contact does not exist in the Contacts Book. |
| **Postconditions** | A new contact and their details will be saved to the Contacts Book. |
| **Main Course** | 1. User will be prompted to add in contact’s forename & surname(s). 2. User saves these details and is then prompted to add contact’s address and post code. 3. User is prompted to fill in the contact’s work/home e-mail addresses. 4. User is prompted to add contact’s work/home/mobile phone numbers. 5. User will be prompted to add the contact’s marital status. 6. User will be prompted to fill in additional notes 7. Contact will be saved when user clicks save button. |
| **Alternate Course** |  |
| **Exceptions** |  |

|  |  |
| --- | --- |
| **Name** | Delete Contact |
| **Description** | The user may have no need for a contact’s details anymore and will be able to delete this contact from the app. |
| **Triggers** | Use Case will be triggered when the user clicks on the “Delete Contact” button. |
| **Actors** | The user of the wep-app. |
| **Preconditions** | The contact is saved in the user’s Contacts Book. |
| **Postconditions** | The contact and their details no longer exist in the Contacts Book. |
| **Main Course** | 1. User is prompted to confirm whether or not they wish to delete the contact. (See AC1) 2. Contact is deleted. |
| **Alternate Course** | AC1 determines what will happen when user does not want to delete a contact.   1. User does not wish to delete contact and clicks “Cancel” button. User is returned to previous page. |
| **Exceptions** |  |

|  |  |
| --- | --- |
| **Name** | Create Account |
| **Description** | User will need an account to access the Contacts Book. |
| **Triggers** | User clicks “Create Account” button on Login screen. |
| **Actors** | User |
| **Preconditions** | User will not have an account and will not be able to access the Contacts Book app. |
| **Postconditions** | User will have an account and will be able to |
| **Main Course** | 1. User will be prompted to enter their e-mail address. 2. User will be prompted to create a username. 3. User will be prompted to create a password. 4. User will be prompted to re-enter password in order to confirm the password is correct. (See EX1) 5. Account will be created when user clicks continue button. |
| **Alternate Course** |  |
| **Exceptions** | EX1 determines what happens when the passwords don’t match.   1. User will be notified that the passwords don’t match and will be asked to try entering them again. |

## Flowcharts

Use Visio to create Flowcharts.

# Object Types and Methods

## Contacts Type

The contacts instance will take all need information from a user that is needed to enter a new contact, the following are its properties:

* fullName: A singular line string field consisting of the contact’s name.
* mobileNumber: A number field containing the contacts mobile phone number.
* homeNumber: A number field containing the contacts home phone number.
* homeAddress: Multi-line string field that takes the contact’s home address.
* homePostcode: String field containing the home postcode of the contact.
* workAddress: Similar to homeAddress but contains the work address of the contact.
* workPostcode: Similar to homePostcode but contains the postcode of the work place for the contact.
* email: A multi-line string field containing both or either of the contact’s personal or work email.
* dateOfBirth: A date field encompassing the contact’s date of birth, so users can be reminded a day before a birthday occurs.
* martialStatus: String field containing the martial status of the contact.
* supplementaryInfo: String field containing any additional information the user deems fit to store about the contact. For example, “Jim from work” to discern between two contacts quickly without having to know second names etc.

The following are the methods which are accessible to the contacts instance:

* saveData(): Saves the contacts details to local storage.
* toString(): Converts the contact to string so the user can copy text and use the information elsewhere.
* getEmails(): Finds other emails stored on the contacts book if the user wishes to send an email to multiple people.
* getLocation(): Marks the contacts home or work place on a map through geo-location using the contacts postcode and address for either or home and work.
* isBirthday(): Checks the contacts dateOfBirth next to the current date and if one day or less till the dateOfBirth then it will alert the user.

## Login Type

The login instance will be used for the user logging into their own account to access their contacts book. Its Properties are:

* username: Single line String field containing the user’s own created username consisting of letters and numbers.
* password: Single line String field containing a lengthy mix of characters without the allowance for spaces and special characters #%-\_\*.
* email: The email address associated with the users account.

The login instance has the following methods available:

* checkUsername(): Checks the username entered is correct with the database.
* checkPassword(): Checks the password entered is correct and is the password associated with the username entered.
* errorLogin(): Returns an error if username and/or password are wrong.

## Application Components

### Object Collections

* User Account
* User’s Contacts

### Application-wide Data

* User’s account username

### General Purpose Functions

* Converting time to easier to understand formats for the user

### Event Handlers

* On button clicks

# Testing

When testing our project, we intend to use a mixture of black box and white box testing. Through testing the project ourselves we will be using white box testing, testing in which the tester knows the internal composition of the structure and design of the software. By using white box testing we will be able to thoroughly test the website using our knowledge of the code we created when developing the website and to attempt to create and reproduce errors where we expect them to occur.

Black box testing would be done using people other than ourselves who are unfamiliar with the project as a whole. This type of testing will be useful in addition to the white box testing as people out with the development team have a chance to test out the project as a whole which can lead to people spotting mistakes the development team may have become blind to since they are working on the project so often. Black box testing can also be done with people of multiple different levels of technical literacy, another programmer could test out the program and think of issues logically however a less technically savvy person may stumble across issues that others may not have seen. Using less experienced people is also beneficial in gaining feedback on the design and their experience with the site, this will be useful criticism and feedback especially for the ease of use the site has for all people and not just people who have substantial knowledge of how websites do and should work.

# Project Features

The above features are things we fully intend to implement into our final project. The following features are articles in which we will attempt to implement into our final project as extra features that go above and beyond the project specifications.

The first of these suggested features is the ability for the application to detect which device you are accessing it from, i.e. desktop or mobile. If you are accessing it from a mobile, then beside the emailing section there will be a button/icon for SMS that when clicked would take you from the project site to the devices built in texting application (or the user’s chosen default texting application) with that contacts number entered the field ready to send the message you wish to.

A further feature that could be implemented would be the ability to sign-in to the website using an already existing google account. This would allow users to access the website easily without having to register and remember separate account details just for the Contacts book site.