Requirements Report

Table of Contents

Project Overview

What is the project?

Our project is a web app that uses Google maps data to organise meeting between friends. The app works be using scheduled events set by the user. Each event has a location associated with it. The location is chosen by the user through the google maps interface. Once the user has chosen a location they can set a time and description for the event. Once an event is created, all the user's friends can see the event, and sign up to attend.

Accounts can add each other as a 'friend'. Every account registered as a friend will be able to see the user's event activity. Users can interact with friends through friend requests, messages, blocking, and removal.

Each account has a unique ID associated with it. The account name is arbitrary and is only for display and recognition purposes. There will also be an ID associated with each account.

Who is the project for?

We are creating this project as part of a university group project module. Our 'client' in this case would be our supervisor Keith Dures. We will also present report to Frans Oliehoek, Michele Zito, Prudence Wong, and Sebastian Coope; so they could also be considered our 'clients'. By April the 28th we will need to submit a demonstration of our finished project, we will also need to submit a portfolio of our work for May the 12th.

What are our mission objectives?

One of the main aims of our web app is simplicity. The app will be designed to be light, easy to use, and user friendly. All the user's experience will come from the interface of the app, so we need to focus on making the interface as easy to use as possible. To contribute towards the simplicity of our web app we will use a minimalist design, the purpose of this is to avoid overloading the user with unnecessary features.

The app will start off with only the core features. By keeping the features few but essential we can cut down on complexity. If we want to add more features, we can do so through our iterative development process. This also allows us to adapt to changing requirement more easily.

Portability is also a major feature of our app. By making our app web based we can implement responsive design into our website. Through this we can support our app on different screen resolutions and devices.

Statement of Deliverables

Documentation

A user Documentation will be created to accompany the program, detailing all features that the user can use. As it is an application there should also be a simpler help guide created for new users.

Essential and Functional Requirements.

The program being created is an application designed around organizing social gatherings. The focus is on being able to use your location and automatically message members of a group within a certain radius of your current location what you are doing. An example would be that in a group of friends using this application you could say that you are having a coffee in location A and to notify all members of the group within 1 mile of your location and that you are having coffee.

A further feature would be the ability to plan social gatherings in the future. The location of the gathering can be placed on a map, and a time and date set. Other users can be invited and can accept. When it comes to the date of the party the location of users will be compared to the location of the party and it will alert them at what time they should leave. The application could also detect when a user is going to be late and alert the host of the event.

A final feature would be the ability for each user to create a calendar time-tabling their day. The program would then compare timetables to find when the most people were free. Users could also input their locations at certain times, so the program could take into account locations as well when planning. This could also be used to create a list of places in a certain category (e.g. restaurants) that would take all users the same amount of time to walk to.

These requirements fall into two categories: essential and desirable.

Essential Requirements:
-Create a group of users from which to organise gatherings and notify of spontaneous gatherings.
-Create spontaneous gatherings
-Notify all users in the group within the specified range of the spontaneous gathering. -
Create a gathering in the future that can be open for all users in a group to join.

Desirable Requirements:
-Inform the host when a user will be late to a social gathering.
-Inform the user when they should leave for the social gathering.
-Input timetables and find times when all users compared have free time.
-Formats pre-built into timetables to allow replication of blocks over consecutive weeks. -
Create a list of places that it would take all users a similar amount of time to walk to.

Methods of evaluation

The program is designed for university students and so can be tested by the team itself in part, but also by other students if necessary.

The requirements will be check to work within a reasonable time and to reasonable accuracy, with parameters to be defined later. Location services should also work to a reasonable degree of accuracy that is precise enough to be able to predict a user's location but has a boundary of error such that users within the area of the gathering will be said to have arrived.

Functional Requirements

1. Project must be a web app
2. App must function on traditional laptop and desktop devices
3. App must store any user data in a database
4. App must function in all major browsers (Chrome, Firefox, Edge, and Safari)
5. User must be able to create accounts
6. User will have personal ID
7. User will be able to add other accounts as friends
8. User will be able to remove friends
9. User will be able to create events
10. Events must be GPS coordinates
11. Events must have a time
12. Events must have a description
13. Events must be owned by one and only one account
14. Users will be able to join their friend's events
15. Users will be able to create groups
16. User will be able to join groups
17. Users must be able to select event coordinates using google maps
18. User must be able to login to the app
19. User must be able to logout of the app
20. Web app must maintain a browser session with the client while they are active

Non-functional Requirements

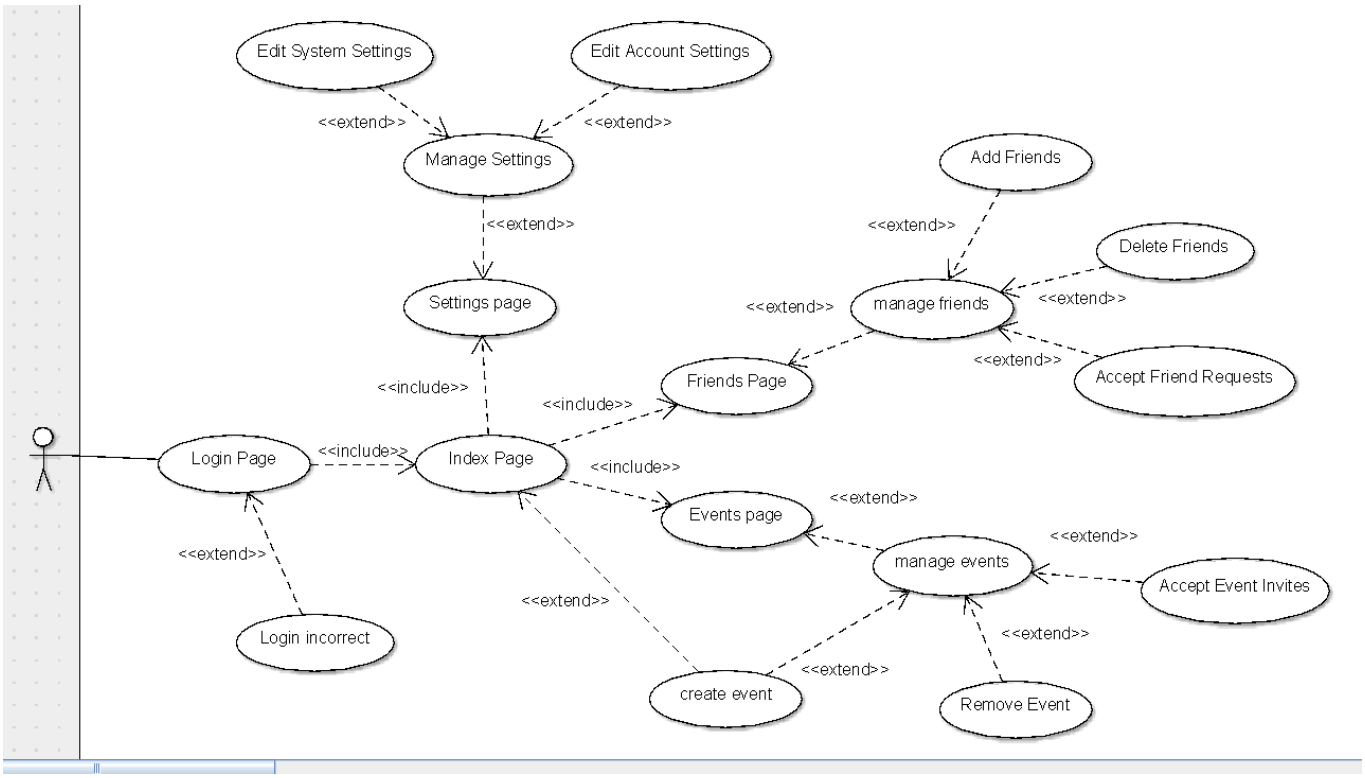
1. Web page must load in under 30 seconds for all major browsers
2. Messages must appear to be instantaneous
3. Events must appear to be instantaneous
4. Friends requests must appear to be instantaneous
5. Database transaction must be under 1 minutes
6. The website must be able to handle 10 concurrent users at a time
7. Website must be easy to use (test three users on the web app's major features, they must make fewer than three mistakes)
8. Logout time must take less than 10 seconds
10. Navigation between web pages must take less than 10 seconds
11. The web app must submit the user's details at least 10 seconds after they press the submit button

Desirable Requirements

1. Password encryption
2. App must function on mobile devices (phones and tablets)
3. Mirror server for the website
4. User will be able to block other accounts
5. At least one database backup
6. Password change option
7. Username change option
8. Website must be less than 2MB
9. Users will be able to send messages publicly
10. Users will be able to receive messages publicly
11. Report user option
12. Social media integration
13. Advertising
14. Use of cookies

Usecase Diagram

To help describe the requirements we have created a usecase diagram. This is the use case diagram for the user for our system, it consists of an entry point and a menu for the user to initially start using the program as all the options will be available for the user to use.



ID	UCU1
Name	Login Page
Description	The user enters the user interface and will be able to enter their username and password. And if the user has not got an account then they will create an account.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.User enters the menu screen. 2.User enters the username + password for their account. 3.IF User has not got an account the user will bring up a form and fill it in with the required info.
Extension Points	Extension from UCU2-Login Incorrect
Triggers	The user enters the app.

ID	UCU2
Name	Login incorrect
Description	The user has entered their username and/or password incorrectly, so to ensure security the user must enter these details again.
Pre-condition	- User has entered username and password incorrectly. - Device is Connected to the internet.
Event Flow	1.User re-enters Details 2.IF the user enters correctly then they will proceed as if they have the first time. 3.IF the user enters incorrectly again then they can repeat this step until they have got it correct.
Extension Points	Extends to UCU1-Login Page
Triggers	Incorrect username and/or password.

ID	UCU3
Name	Index Page
Description	The index is the main page where the map is for event creation.
Pre-condition	- User has entered username and password correctly. - Device is Connected to the internet.
Event Flow	1.User is on the main page 2.User can navigate and select menu Options.
Extension Points	Extension from UCU9-Create Event
Triggers	The User.
ID	UCU4
Name	Settings Page
Description	The settings page is where the user can change account and system settings for the webapp.
Pre-condition	- User has entered username and password correctly. - Device is Connected to the internet.
Event Flow	1.User enters settings page 2.User can select settings and modify them.

Extension Points	Extension from UCU5-Manage Settings
Triggers	The User.

ID	UCU5
Name	Manage Settings
Description	This is where the user can see the current settings, and the settings that are available to change.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.User selects a setting 2.User changes a setting within given parameters.
Extension Points	Extends to UCU4-Settings page Extension from UCU6-Edit Account Settings Extension from UCU7-Edit System Settings
Triggers	The User.
ID	UCU6
Name	Edit Account Settings
Description	This is where the user can change the settings of their account information, such as name, age, e-mail, etc.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.User selects and manipulates settings as wished.
Extension Points	Extends to UCU5-Manage Settings
Triggers	The User.

ID	UCU7
Name	Edit System Settings
Description	This is where the user can edit the websites settings to suit them, such as, colour-blind mode, and font size changing.

Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.User selects and manipulates settings as wished.
Extension Points	Extends to UCU5-Manage Settings
Triggers	The User.
ID	UCU8
Name	Event Page
Description	This page has all the events and event invites on it for the user to look through.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.The user can view events. 2.The user can create events. 3.The user can delete events.
Extension Points	Extension from UCU10-Manage Events
Triggers	The User.
ID	UCU9
Name	Create Event
Description	The user wishes to create an event at a location on the map, and invite their friends to this event.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.The user selects create event. 2.The user inputs a name for the event 3. the user selects a location and time for the event. 4. the user edits the event description. 5. the user sends out invitations to their friends for the event.
Extension Points	Extends to UCU3-Index Page Extends to UCU10-Manage Events
Triggers	The User.

ID	UCU10
Name	Manage Events
Description	The user can manage their events on this page, by editing an event they have hosted/going to, or removing other events from their feed.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.The user can view and manipulate the events they are going to as they wish.
Extension Points	Extends to UCU8-Event Page Extension from UCU9-Create Event Extension from UCU11-Remove Event Extension from UCU12-Accept Event Invite
Triggers	The User.

ID	UCU11
Name	Remove Event
Description	The user wishes to remove the event from their timetable meaning that it will not be on their event feed.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.The user clicks the delete event button
Extension Points	Extends to UCU10-Manage Events
Triggers	The User.

ID	UCU12
Name	Accept Event Invites
Description	The user accepts an event invitation given to them by someone on their friends list.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.The user clicks either the accept or decline invitation button.

Extension Points	Extends to UCU10-Manage Events
Triggers	The User.

ID	UCU13
Name	Friends Page
Description	This page of the app is where the friends list is kept and where the user can see their friends.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.The user can view their friends list. 2.the user can accept or decline friend invites.
Extension Points	Extension from UCU14-Manage friends
Triggers	The user has selected this page from the index page.

ID	UCU14
Name	Manage Friends
Description	The user can browse and see all the friends they have on their account as well as see requests or manipulating their friends list by adding or removing users.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.User can look and manipulate the friends list how they wish.
Extension Points	Extends to UCU13-Friends Page Extension from UCU15-Add friends Extension from UCU16-Remove friends Extension from UCU17-Accept friend request
Triggers	N/A

ID	UCU15
Name	Add Friends

Description	The user has chosen to add someone to their friends list.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.User selects the add friends button. 2.User enters the details of the person they want to add. 3.User sends the request.
Extension Points	Extends to UCU14-Manage Friends
Triggers	The user clicked the add friends button option.

ID	UCU16
Name	Remove Friends
Description	The user has chosen to remove someone from their friends list.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.The user selects the remove friend button on the person they want to remove.
Extension Points	Extends to UCU14-Manage Friends
Triggers	The user clicked the remove friends button option.

ID	UCU17
Name	Accept Friend Request
Description	The user has received a friend request from another user of the app, the user can choose whether to accept or decline the request.
Pre-condition	- Device using the web-app is switched on. - Device using the web-app is connected to the internet.
Event Flow	1.The user clicks the accept or decline option on the request.

Extension Points	Extends to UCU14-Manage Friends
Triggers	The user has been sent a friend request from another user.

Preparation

What we need for legality purposes

- We need the **data protection act** and to make sure we are in the guidelines of protecting people's data. Since our app will be storing information like their name and age etc.

- **Terms of use for contributors** to make sure we can ban people if they are insulting or being hurtful towards other people.

- If we allow advertising (later on) we need to add **terms of use of advertisers**.

- We need **a privacy policy** if we take their email have passwords which we will later on in the development.

- We need a **website Disclaimer** to save ourselves if someone gets hurt when having a coffee or traveling to a location to meet up for an event etc.

What we need to help us with the functionality

We use adapt.js to do resolution specific behaviour, (script that does interface based events on how big the screen is), because we are using a webapp the screens in which people can view will all be different. So, we need to address this. We use 960 grid system splits screen up into 12 chunks, the blocks can change size and slip under each other, can split up the website and can give everyone a chunk of the website so it makes it easier to develop. This will help assign the work more evenly between us and allow for responsive design of our webapp.

What to research

- Research other social sites, look for similarities between them, simplicity tactics, take inspiration from how they have designed certain aspects.
- Research web apps to figure out what features they use, libraries, how they function.
- Research design trends to help our design stage.
- Research database design techniques.

Current Skills We Possess

We know basic CSS, html front end design of websites, we know client side scripting using jQuery and JavaScript libraries. We know basic SQL and PHP for the back end and database querying. Basic Photoshop for storyboards, we know how to use design

diagrams and functionality diagrams including state charts, flow charts etc. We can use our knowledge to get a good start on our development and then hone and develop our skills as we go through the project.

What we need to learn

We need to learn how to set up a database, set up PHP server efficiently, we need to find out how to program encryption into out app. More advanced Photoshop, learn better design techniques. We will research these and develop these as we progress through the project.

Data required

We need geolocation (giving GPS co-ordinates), we need people's names and email (later on) to log in. Google maps data (coffee location etc.). We expect the user will input most of this information.

Design Methods

For the interface: Using CSS, HTML. We want to use the 3 click system (no information is more than 3 clicks away), we can make storyboards to plan what the interface can look like using Photoshop or drawing. We decided we should create wizard of Oz type prototyping to check whether certain interfaces can work with our webapp. We will be using MVC modal view controller to make it more efficient. Have a minimalist's design with our web app as we don't want to overload the user with information. We will take a Rapid prototype development approach with our project. (iterative development).

Design Documentation

Will consist of storyboards, ER diagrams for the database, state charts, use case diagrams, user input, questionnaires, interviews to find out what the user wants. low and high fidelity prototyping, pictures of prototypes, flow charts. We will use these design techniques to crate our interface and style of our web app and hopefully end up with a simple but interesting design.

Required Hardware

- **Server space**

Since our project is going to be a web app we need a server to base our website on. The server space provided by The University of Liverpool is free and should be enough to launch our web app, assuming we don't have much initial traffic. If we attract outside traffic (outside meaning outside the university), then we may need to move to a more capable private server.

- **Live prototyping / demonstrations**

For live demonstrations, we may also need to use mobile devices, laptops, and desktops; all of which is readily available, either owned personally or by the university.

- **Work stations**

Every member of the team has a personal laptop or desktop computer. If any of these machines break, or if any of the machine powerful enough we can use the computers in the computer science labs

Required Software

- **Web server solution stack (XAMPP)**

To develop our webapp we will need a local web server, this can be provided by the free open source web server solution stack XAMPP. By using XAMPP we can develop both the client side and server side ends of our web app without the need setup on the server. The reason we chose XAMPP in particular, is that most of our team members have basic experience with XAMPP, PHP, and SQL; reducing the cost in time for learning new tools.

- **IDE and text editors (Anything)**

All of our team members are free to use which ever IDE or text editor they want. It would cost too much time in training to force everyone to use the same IDE or text editor. Allowing everyone to use the tool they are most familiar with will also make them a lot more productive.

- **Source code management (Dropbox)**

To minimise training time, we will use a tool everyone on the team is familiar with. Dropbox will be used to store and keep our source code and documentation. This does mean we will miss out on the features like version control and access control provided by websites like GitHub or Google's cloud source repositories. This means we will need to carry tasks like version control and bug tracking manually, however, it will save us time in training.

- **Graphic design (Photoshop)**

In the likely event that we need to create custom images for our website we will need to use a graphic design application. Out of the possible options GIMP and Photoshop seem most appealing. GIMP is free and open source meaning all our team have access to it, but most of our team have experience using Photoshop. On top of this Photoshop provides a 30-day free trial. Since graphic design will be a more specialised task in this project we won't need *every* team member to have access to the software; because of this we will use Photoshop (through Photoshop CC – free trial).

- **Debugging (Anything)**

There is no need for every member of the team to use the same debugging software. There is no financial cost for most debuggers, there are no special features in any debugger that would benefit the whole project if every team member used it, and everyone has their personal preferences. For the same reasons, we haven't chosen a particular IDE, and we will not choose a particular debugging tool.

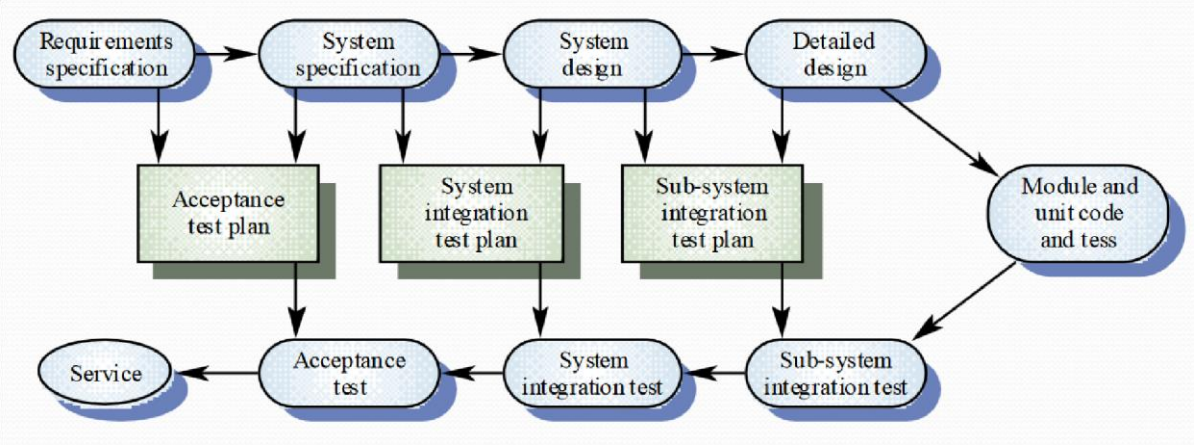
- **Communication (Discord)**

To keep everyone on the team updated on the project's progress we need an online communication platform. There are plenty of options to choose from, and many useful features we could take advantage of. Since every team member is familiar with the software Discord, we can create a new server dedicated to our project.

- **O/S (Anything)**

Since our web app will be platform dependent there isn't any requirement to develop on a particular operating system. Most of our team members will be working on Windows, but Ubuntu and Mac OS will also be used.

Required Testing



Our testing will be loosely based on the V-model of development provide in comp201, lecture 22, slide 22. Using this model we will have four levels of testing:

1. **Unit testing**: testing individual methods against a range of possible inputs
2. **Sub-system integration tests**: testing all the sub-systems
3. **System integration tests**: testing the system as a whole
4. **Acceptance test**: testing the system complies to the given requirements and specification 5.

For **unit testing** we will need to develop a range of test cases to make sure every possible input for each method is handled correctly. Implementing automated unit tests at an early stage will allow us to spot bugs that are introduced into the system as they are introduced.

Sub-system integration testing will allow us to test the individual sub-systems in our web app. Since there are many distinct areas of our app (e.g. the scheduling system, map system, web interface, back-end, database, etc.) we need to make sure that each area works as expected before we can integrate them into a larger system.

System integration testing will be the stage in where we combine all the sub-systems into one large system. After all of these sub-systems have been combined we need to make sure that the resulting system works correctly.

Acceptance testing will be carried out once all preceding tests have been passed. We can check our system against the requirements and specification to make sure everything is covered; once every requirement has been satisfied we can deliver the final product.

Since the development process is going to be iterative we may cover each of these stages multiple times as features are introduced.

Project Schedule

Requirements Stage

30/1	Team Preparation

…		
8/2	Team Preparation	
9/2	Group Meeting: Initial Ideas	
10/2	Requirements Doc Work	Research
…		
16/2	Requirements Doc Work	Research
17/2	Requirements Submission	
18/2	Research	
…		
20/2	Research	
21/2	Requirements Review	

Design Stage

22/2	High level system design	Research	
23/2	High level system design	Research	
24/2	Group Meeting: High Level Design		
25/2	Interface Design	Documentation	Research
...			
1/3	Interface Design	Documentation	Research
2/3	Group Meeting: Interface Design		
3/3	Back-end Design	Documentation	Research
...			
8/3	Back-end Design	Documentation	Research
9/3	Group Meeting: Back-end Design		
10/3	Database Design	Documentation	Research
…			
14/3	Database Design	Documentation	Research
15/3	Design Document Compilation		
16/3	Group Meeting: Database Design and Design Doc Review		
17/3	Design Submission		

Development Stage

18/3	Skeleton Prototype Dev.	Testing	Documentation
…			
23/3	Skeleton Prototype Dev.	Testing	Documentation
24/3	Group Meeting: Prototype Iteration		
25/3	Interface Prototype Dev.	Testing	Documentation
...			

30/3	Interface Prototype Dev.		Testing	Documentation
31/3	Group Meeting: Prototype Iteration			
1/4	Back-end Prototype Dev.		Testing	Documentation
…				
6/4	Back-end Prototype Dev.		Testing	Documentation
7/4	Group Meeting: Prototype Iteration			
8/4	Database Dev.		Testing	Documentation
...				
13/4	Database Dev.		Testing	Documentation
14/4	Group Meeting: Prototype Iteration			
15/4	Front/back end integration	Back end db integration	Testing	Documentation
...				
20/4	Front/back end integration	Back end db integration	Testing	Documentation
21/4	Group Meeting: Prototype Iteration			
22/4	System Integration		Testing	Documentation
…				
25/4	System Integration		Testing	Documentation
26/4	Group Meeting: Final Iteration and System Review			
27/4	Demo Preparation			
28/4	Demo Submission			

29/4	Portfolio Dev.
...	

10/5	Portfolio Dev.
11/5	Group Meeting: Portfolio Review
12/5	Portfolio Submission

Risk Assessment

- **Absent team members**

If any team members for whatever reason can't work on the project, then their work will need to be distributed across the team. The schedule is structured so that there is always one main task with more than one team member allocated. If any team members are missing, then they will be replaced by re-allocating members from this task.

- **Falling behind schedule**

Over the next few months it is likely that at some point our project will fall behind schedule. In that case, we will need to re-allocate team members away from tasks such as documentation and testing. This may compromise the quality of the project, but it should help us meet our deadlines and milestones on time. In the case that we fall too far behind schedule we may have to skip certain activities, or reduce the size of each design/development cycle.

- **Delays**

It is likely that delays will occur during the projects lifespan. For example, the server needs to be made available for live testing, our tester needs code from the developers to test, prototype reviews rely on everyone finishing development, documentation, and testing on time. To minimise delays we will need to communicate regularly. We will do this by keeping track of the project's progress on our communication platform Discord and through Vital.

- **Broken Hardware**

Since there is little hardware involved in the project the risk of hardware braking is low. Since the servers are maintained by the university, and our workstations can be replaced by university computers, the only time hardware failure would be an issue is if the university servers were not available during our project demo. In this case, we would need to find a private server to work from.

Required Skills

- **Front-end web development**

Since the interface of our project is going to be entirely web based we will need team members with front-end development skills. We will need to cover HTML and CSS for the presentation of our web app, and JavaScript with JQuery to add interactivity. Most of our team has some experience with front end web design so training won't be as much of an issue.

There are plenty of resources online for front-end web development. W3Schools contains tutorials on all the languages and tools we will need for the front-end of our project.

- **Back-end web development**

Our web app will need to have a back-end to provide dynamic content, and to handle interactions between the user and the database. We will use PHP as our server side language, and MariaDB as our database query language. Since our team only has limited

experience with back-end development we will need to spend time training our team. We will use XAMPP as a training platform.

As with front-end development, there are plenty of resources online. W3Schools contains tutorials on PHP and SQL.

- **Database design / development**

Our web app will require a database. Most of the time spent on the database will go into design, and development will be closely tied with back-end development. Since most of our team have covered databases extensively we won't need to spend too much time on training. We do need to research how we will integrate our database with our back-end.

All the group has covered two modules on databases, one of them covering design and development. Any material we need will be available in the module slides and text books.

- **Graphic design**

The interface of our project will likely custom images. Since this is quite a specialised task we don't need to train the entire team. We expect graphic design to be a very small part of our project so we will likely spend as little time training for graphic design as possible.

The designs for our web app will be very specific, so we need to find web tutorials that cover the type of design we want to create.

- **Google Maps JS API**

The main feature in our project relies on the Goggle Map JavaScript API. No one in our team has experience with this API so we will need to make sure we read through and fully understand all the documentation before we start the project.

The Google Maps JavaScript API website contains all the documentation we need.