LOCAL HYPE

LOCAL HYPE.

INTERACTIVE MULTIMEDIA DESIGN

YEAR : 2015
SAMANTHA NELSON : B00595677
MENTOR : DR GEORGE MOORE
PSG: 8

With special thanks to Dr George Moore, for
his guidance throughout the project.

***

To the Get Invited team for permitting use of
their API, and essential feedback provided.

Lastly to my family and friends for their
unwavering support, without I
would surely be lost.

CONTENTS

1 - INTRODUCTION

Planning to attend events is a process found to be somewhat confusing, with todays technological advances people can still remain unorganized with the process. There are many different mediums that can be utilized for communication between individuals. There is currently no platform that provides all the answers, leaving users to have to use multiple, most not even fit for purpose.

The first issue with the event planning process being that there is no one true source to retrieve events from. Many platforms promote events, from mobile applications to local tourism websites. Knowing that not one source will contain all the possibilities, leaves users with the only option of having to search multiple resources to find events. This adds many unnecessary hours to a task that should be simple. Secondary to this, these resources are often outdated with no real usable event listings.

The second issue presents itself as an organization issue. Once events have been gathered the process of inviting friends to these events begins. Without a common platform event invitations get sent via text, social media messengers even email. This becomes a task when not all invitees use the same platform, keeping track of details like this becomes a hassle.

Additionally, social interactivity today is mainly digital. With the average person spending four years of their life on their phone (Prince Ea). Social networks promote interactivity through them, rather than in person. When events get planned often any interactivity at the event is spent on a mobile device rather than with people. An application should promote connections between people, subtle facts that show user interactivity such as where they have been together without the user having to upload anything themselves.

1.1 SOLUTION

There is a real need to have one service that provides all elements needed to have an enjoyable experience. A reliable events source used to access local events dependent on location. A system layered on top of it to house users, links between these users and how they interact together. Inviting each other and organizing information surrounding the events they wish to attend.

Engaging users by making the experience rewarding such displaying how many friends they have connected with, and how many events they have attended so far. These interactions can be managed by the application, leaving the users to interact with each other. This will be more beneficial, rather than promoting a service where the focus is to add comments and pictures when attending the event.

1.2 CONTEXTUALIZING THE APPLICATION

There are similar applications, but none provide all the capabilities that are mentioned above. All contain useful features, but still cause the user to plan across platforms. For example Hype is a native application that was specifically developed for London that focuses on real time events. “Spontaneity is now. We feature what is happening right now or will start in no more than 3 hours” (Hype). The functionality and design of this application are to a high standard, yet it still detaches itself from the solution as once an event has been sourced it provides no way within its application to invite friends to the events. Arguably this is the most neglected factor within the market, many applications do provide event searching (Appendix A), yet due to the detachment of the planning aspect they become somewhat redundant. Thus the user relies upon other services like Facebook messenger to organize the sourced event.

1.3 AIM

Local hype is an events planning web application, connecting users to local events happening in their area and invite friends that they are connected with.

1.4 OBJECTIVES

- Gain approval through discussion, to use Get Invited API.
    - Ability to search upcoming events on Get Invited by location, looking to find events within a certain radius of the user's location.
- User experience development by evaluating similar applications.
- Define how users wish to use the application through surveys.
- Gain insight on tools and development practices needed e.g. API's through research.
- Give the application a strong brand image.
- Visually designed and user friendly.
- Proven feasibility of project through prototype integration with approved event API.
- Functional user account system.
- Ability for users to get directions to events, via walking driving or cycling.
- Implement additional features based on scope and time frames.
- Robust application, through development testing for errors.
- Fully usable application, through group testing.

1.5 OVERVIEW WORK UNDERTAKEN

With any project there will be several key steps taken to insure the project is a success. The feasibility of the project must be considered before commencing the planning stage. Once the application is proven to be achievable a plan for the project will be created. Requirements of the project will then be defined using user surveys. These requirements will then be carried out designing, implementing functionality and testing before finally concluding the project as a completed solution.

1.6 REPORT OVERVIEW

This report will provide project management documentation for the application. Providing an overview of how each of the above stages are conducted and determining the success of each. Detailing how each of these stages are managed with clear reasoning as to why the project was conducted in a particular way, using particular methods. It provides clear justification as to how the project is carried out.

2 - CONCEPT DEFINITION AND TESTING

2.1 IDEA GENERATION

The project commenced with idea generation for the application. Three initial ideas were considered, all of which had benefits and drawbacks. The final conclusion as to which would be taken forward as the selected application would depend on the most feasible option.

- Local hype – Events based application
  Responsive web application, that uses the Get Invited API to source local events. User profiles that allow friends to send and receive invitations, based on these local events.

- Split – An online money organiser
  Account based responsive web application for managing money on a weekly/monthly/yearly basis. The ability to view graphical data of the information entered throughout these periods on a dashboard as well as remaining monthly/weekly expenditure available.

- Strike - An online ping-pong games management system
  Responsive interactive website. Front end users have the ability to register for the Ping-Pong event and book a booth service (booth users can order drinks from that booth). Staff can log into a back end where they can control the Ping-Pong as it happens as well as track drink orders coming from the booths.

2.1.1 Selected application

While all of the applications had potential there was a clear division for marketability. Strike online game management proposes an application best built for a service that already exists. While Split the online money organiser would be a custom application there is many already like it in the market all preforming the same user functions. Local hype is a concept that is relatively unique and while extensive competitor research (Appendix A), is undertaken it is clear from a starting point it does not have as much competition. Local hype was the projects application of choice.

2.2 REQUIREMENTS ANALYSIS AND SPECIFICATION

2.2.1 Background of the Project Effort

Creation of an events application, that will allow users to send and receive invitations from connected friends to local events based on the their current location.

There is a real need to have one service that provides all elements needed to have an enjoyable user experience while searching for events. A reliable events source used to access local events dependent on location. Without an application of this spec users will continue to use social media platforms not fit for purpose.

Due to no similar applications being available the opportunity for this application is high. Not only does it present itself as a useful application but also solving a problem that users currently have. Removing the events process from social media allows users to preform this function more effectively.

2.2.2 Goals of the Project

Goal: Application works in all major browsers

Purpose: Working across browsers allows for more users to access the system.

Advantage: More users that have access will increase the popularity of the application as well as the reputation as users expect web apps to work on all major browsers.

Measurement: Testing the application across all browsers before release.

Goal: Users will find the application usable.

Purpose: Simplistic user interface design insures that users understand how they are meant to interact with the application.

Advantage: With a simplistic design you increase the user base so all technical levels find the application easy to operate.

Measurement: Amount of sign ups to the application.

Goal: The application will return event results immediately.

Purpose: Returning event listings immediately, insures the user will continue to use the application.

Advantage: With an immediate listing of events users will continue to use the application and not become frustrated with long loading time frames. This can also help the popularity of the application over competitors.

Measurement: Users profile growth over set time frame to identify application usage.

2.2.3 The Client

The client of this application will be the creator of the application. The application will be conducted as a project for the creator. Possibly reaching commercial ability post completion, but for in-house consumption firstly. The reasoning behind the products credibility is driven from extensive commercial research.

2.2.4 The Customer

As the project is initially for in-house consumption the customer will be also be the client. Responsible for the credibility of the application as well as recommendations towards possible changes will be project mentors. They will help influence the creator of the application when making decisions. Great insight for post project consumer needs will be conducted below through user research. Users of the application will provide a greater insight into project needs instead of relying solely on the application creator.

2.2.5 User Groups

The application will be used by a verity of people; there are no clear groups known that won't use the application but rather identifiable categories of user that will be expected to use the application more frequently.

User category: Students/ Young professionals.
Reason for usage: Locating events to attend with friends.
Technical experience: High.
Issues: Lack of engagement can result in the loss of users.
Solution: Keeping track of events and engagements to keep interest.
Priority: Key users.

User category: Parents.
Reason for usage: Organising event information for family.
Technical experience: Low – High.
Issues: If planning from home the events will only show within a radius.
Solution: Allowing filtering of miles to events so can be planned from further.
Priority: Secondary users.

User category: Travellers/Tourists
Reason for usage: Finding events locally within the area they are visiting.
Technical experience: Low – High.
Issues: An overcomplicated design will result in it not being usable.
Solution: Minimal design will allow anyone to follow the information design effectively.
Priority: Secondary users.

2.2.6 Personas

STUDENTS

32%

QUINN

AGE: 19
INCOME: £6000 pa
HOUSEHOLD: Lives Alone

"IT NEEDS TO WORK FAST, INSTANT RESULTS"

Nineteen year old Quinn is a female student. She is always on her smart phone trying to keep connected to her friends throughout a busy day. Likes to use applications but gets put off when they are too complex to use, often any chance to use an application is only for a few minutes at a time. Socialising is a scheduled activity for Quinn; around working she only has a few dedicated hours to spare meaning she wants a structured way to plan her week. Living in a city can be tempting for a student operating on a budget so she must always be conscious as to how much she is spending.

YOUNG PROFESSIONAL

27%

MATT

AGE: 23
INCOME: £19000 pa
HOUSEHOLD: Lives With Partner

"ALWAYS ON THE MOVE, IT NEEDS TO BE ACCURATE"

Matt is a recent graduate and now young professional, he has his nights to himself and now find himself able to plan more activities due to his increased income. He likes to organise nights out in advance so it is important to him to view events ahead of time. Matt lives with his partner who also works a busy schedule so it is important that event invitations are easily managed so they can organise to attend events toegther.

PARENT

18%

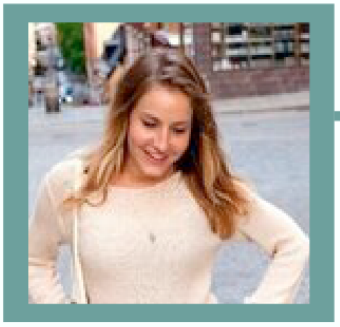
JOHN

AGE: 32
INCOME: £27000 pa
HOUSEHOLD: Wife & 2 Kids

"I'M SPONTANEOUS, SO IT NEEDS TO BE UP TO DATE"

John is a thirty two year old father. When it comes to the weekend he likes to be spontaneous and do the first fun activity that comes to mind with his family. He hates searching for activities on the internet as he feels it wastes to much time. His family home is outside the city yet his partner works in the city centre, planning activities needs to be clear and concise so it doesn't become laborious to organise.

TRAVELLER



23%

JAMIE

AGE: 26
INCOME: £17000 pa
HOUSEHOLD: Lives Alone

"MINIMAL, AND EASY TO UNDERSTAND"

Jamie is a tourist traveling from France, he has a good understanding of basic English but is not when reading large paragraphs of content. He enjoys music and food. While moving from city to city he often needs to plan last minute the activities he wishes to do. While traveling he can run into friends he has made on his travels, some may inform him in advance they will be going to the same countries, though often it is last minute and a happy surprise.

2.2.7 Constraints

Constraints are a part of any project; working within these can help increase the success rate of a project. The developer along with user input can generate a set of constraints; as long as these get monitored appropriately they can be used to control the scope and success of the project.

Description: Application will not work in older browsers.

Rationale: The creator will not have the time frame to insure that the application will work in more dated browsers.

Fit criterion: The application will not be tested for browsers under IE 10.

Description: The application will be tested on all modern browsers.

Rationale: The web app will work on all modern browsers as the creator can insure the scope of the project allows for this. Secondary to this if time scale allows the application will then commence testing on IOS devices.

Fit criterion: Testing the application works in modern browsers (Chrome, Firefox & Safari). Once completed testing will commence on selected IOS devices.

Description: Integration with new Get Invited API.

Rationale: Get Invited API is a local resource, opposed to its competitor that is based in America.

Fit criterion: Testing/issues can be resolved with the company via meet ups, and immediate email responses as they are within the same time zone.

Description: Completion by April 2015.

Rationale: The project has a deadline of April 2015, this allows time for testing and other post project examinations.

Fit criterion: The project will be live on the server by 30th April.

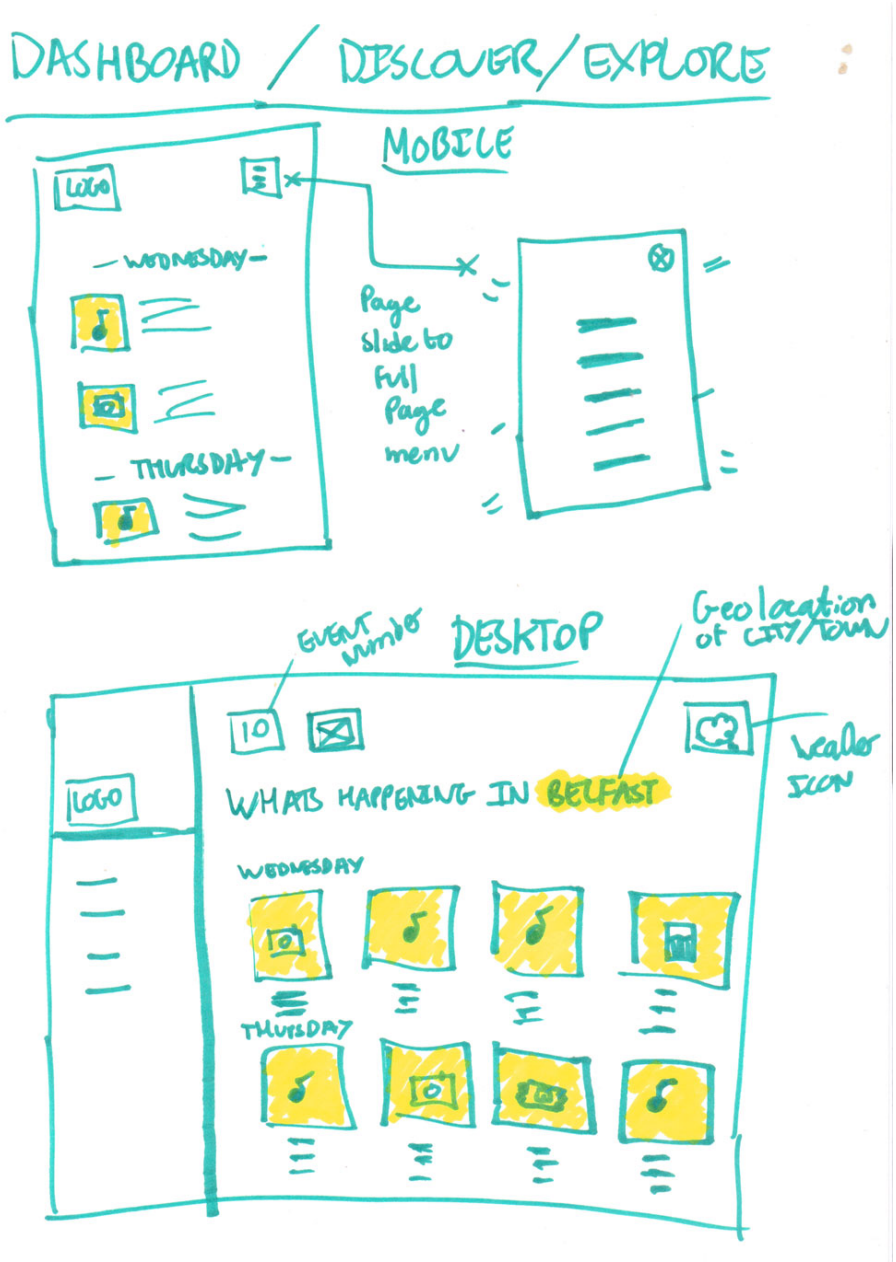
2.3 PAPER PROTOTYPING

2.3.1 Basic Sketches

The process began with basic sketches of the application. This involved developing simple ideals, such as positioning of page elements, rough ideas of what would appear on each page as well as potential interaction from users. Detail at this stage is minimal, getting a feel for how the application will work as opposed to detailed design. Considerations for both mobile and desktop are shown.

Below is an example of the dashboard/discover section for the application; the remaining basic sketches can be found in Appendix G.

Fig 2.1 – Explore Wireframe



2.3.2 High fidelity

High fidelity sketches provided refined prototypes of the initial basic sketches, as well as these more developed wireframes of the applications key pages. See appendix G under high fidelity for a full list of detailed wireframes.

Greater detail and description was needed for several pages:

- Login

In its simplicity, there still needed to be a more detailed description of how the basic layout will operate. Detail allowed description of the video background as well as subtle loading animations.

- Profile

Clear definitive layout post 6UP experimentation. Simple touches such as closed accordions on a mobile for clarity, and open on a desktop.

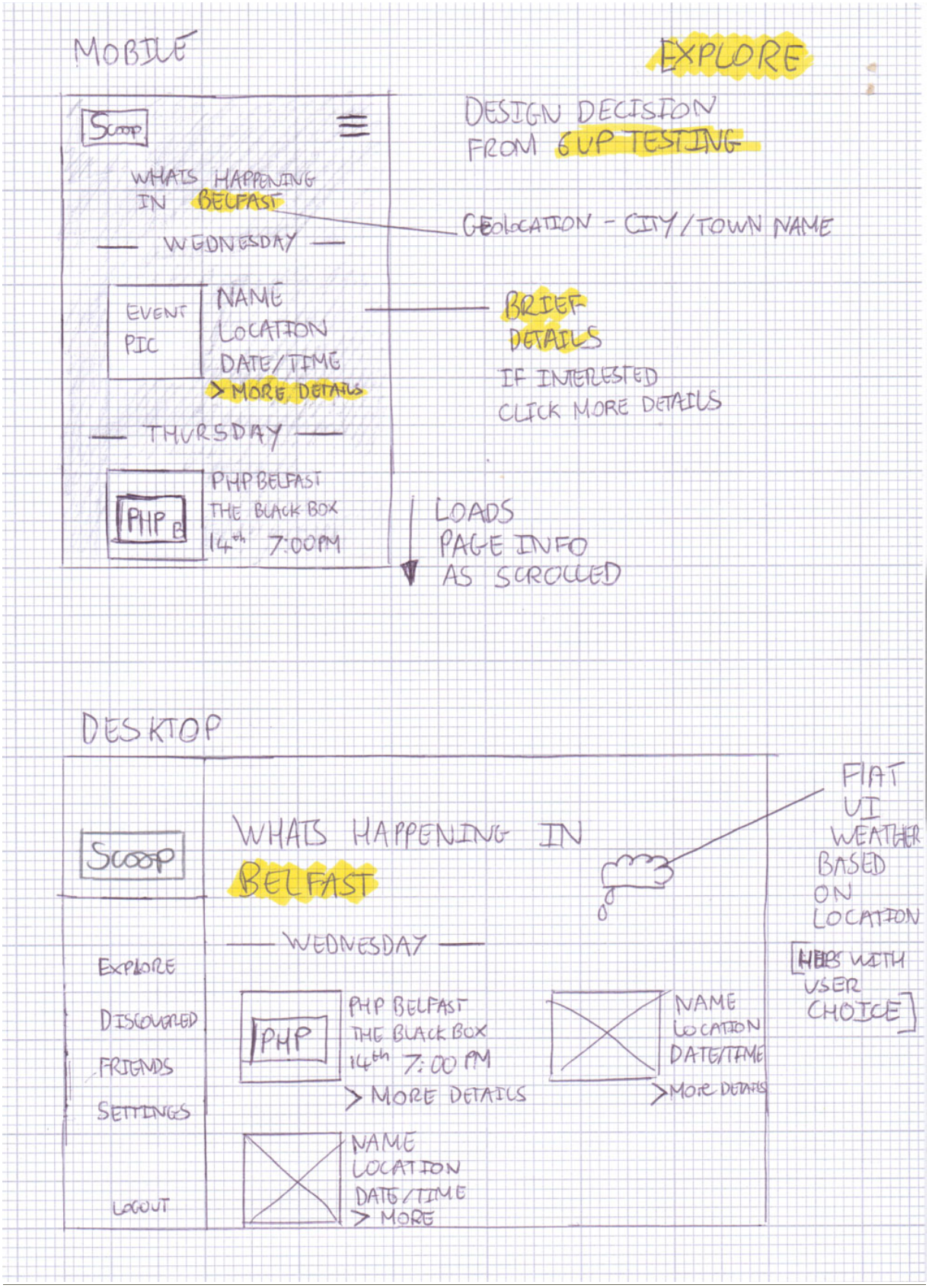
- Explore

Post 6UP design, showing the finalised layout for mobile. As well as how extra features can be added for desktop to increase UX.

- Discovered

In depth description of the interactions of this page helped refine the functionality of how this page would differ from desktop to mobile.

Fig 2.2 – Explore High Fidelity



2.3.3 - 6 UPS

Basic developments lead to several pages needing further thought. This process helped to eliminate potential issues that could be presented. The two following pages that were developed further were (Appendix G):

• Profile

This page would contain several pieces of information, settings, friend listings and statistics about the current user. From the 6UP you can see how what started as a tabbed display of information moved to clear cut accordions. Details of how the design moved are detailed at the side.

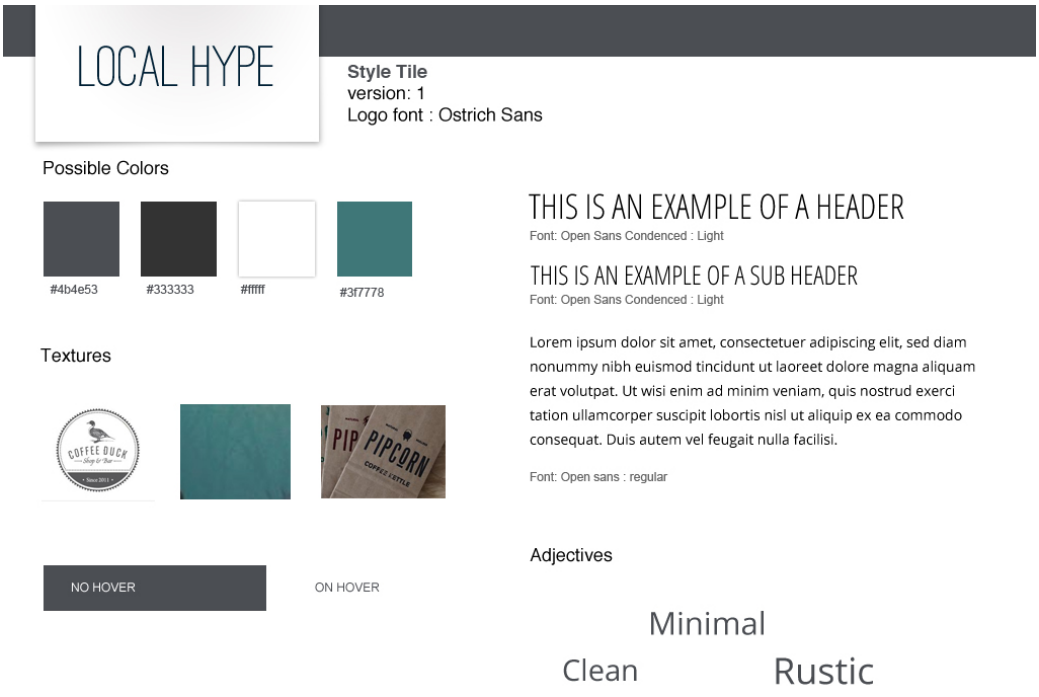
• Explore (dashboard)

The explore page shows how the events listing will be displayed to the user. A large part of this was trying to organise how the positioning and hierarchy of event dates would affect the layout. Both of these will have their final versions taken to the detailed prototypes.

2.3.4 Style tile

The initial approach to a brand the application was to explore the typography; colour themes and a general feel for the application. This was produced through style tiles; the finalized style tile is shown below. It provided a theme for the applications UX design to be based off.

Fig 2.3 – Style Tile



Two style tiles where created during the prototyping. Both are different representations of the application. You will notice that the name of the application is different on each style tile.

The two names generated for the application were felt to have very different personalities, and therefore styles. Applying each to a different style tile has shown me two very different tones for the project.

Each of them displays as the top a basic low-level logo possibility; the font type used is displayed to the right of the graphic. The other fonts mentioned are web fonts to be used within the application itself. Matched as best as possible to the corresponding typeface possibility for the projects branding. The rejected style tiled theme can be found in Appendix G.

2.4 FEASIBILITY TESTING

Determining the feasibility of a project can be done through a number of methods.
The first approach is to collect relevant facts based on user surveys. The developer, in addition to these user-based facts, can then generate a list of assumptions. Reverse engineering is then considered by the developer to get an accurate look at how the application could be built, functionally and considering technologies. After gathering this information if the project is feasible the scope and risk management will begin. Defining the scope of the project to spec if the time scale is sufficient and risk analysis to document any known risks for the project.

2.4.1 Relevant Facts

Relevant facts are driven from research preformed. A survey was generated to ask direct questions about the usage and functionality of the proposed application; the facts that came from this survey can be used to generate requirements for the project (Appendix D).

- Chrome is the most popular browser (74%), followed by Safari (16%). [1.2]
- 10 mile distance was preferred by survey takers, secondary was 25 miles. [1.1]
- 96% use their mobile device regularly to access the web. [1.3]
- Web apps are used on mobile (96%) more than tablet (32%) or desktop (65%). [1.3]
- Importance of information on invites, 43% feel date should be the most highlighted feature. [1.5]
- Importance to app features, the most important was finding local events (71%). [1.6]
- 51% of users feel that they would use the app weekly. [1.7]

2.4.2 Assumptions

- Access to server space will be available at all times.
- No money will be need to pay for API access.
- The developer will be working will a set list of technologies.
- The time frame for the project is agreed and will not change.
- Guidance will be available if an issue within the project arrises.
- Project scope will not be modified.
- Developers have access to add devices specified to test.
- Event API provided will be available via email for queries.

2.4.3 Reverse Engineering

Reverse engineering is another useful way to generate requirements for the project.

"Reverse engineering processes may ease the task of comprehending an existing application, providing useful insights into its architecture, low-level design, or the final behaviour offered to its users." (Tramontana, n.d).

Looking at similar systems can give clear indications of technologies used, practices and other qualities that may allow the developer to generate a more accurate time frame and scope for the project. Looking at an events web application "Eventful" gave an insight into key technologies that could be used or excluded from the development process. The application used a combination of JQuery and raw JavaScript for client side scripting and XHTML as mark up. The website is not responsive. It uses API libraries such as plusone.js to add additional share buttons to the website so that users may share the events.

Making use of technologies such as JQuery, is something intended for the application to be created, its huge development base eliminates risk, as most issues that could be encountered can be resolved with the community online. With the use of HTML5 and CSS3 becoming the new standard, moving away from XHTML is another clear decision to be made.

2.4.4 Scope

The scope of a project is variable, based on a number of factors. Qualities that can be affected by the scope of a project can be time, cost, resources and quality. Effectively mapping the scope is "To clarify the limitations or parameters of the project and clearly identify any aspects that are not to be included" (CIO staff).

The project in question has a reasonable duration for completion, though only key development functionality i.e. (user profiles, ability to invite friends & API connection to events based on location) will be implemented, as there will be only one active member on the project. Extra interactivity will only be permitted if the key functionality of the application is successfully working. The quality of the application will to be a high standard using new technologies and practices, such as HTML5 and CSS3. It will work across all modern web browsers and if the scope allows for it IOS devices. Though due to time limitations on the project there will be no ability to accommodate older browsers. Cost will not be an issue for this application as all technologies being used are free to do so. Any testing resources needed are also acquired (Appendix C). Google

maps to find the Geolocation has a free base rate of 25000 map request per day, this will be sufficient for the project and will only become an issue if the application gets commercially used post completion.

Scope of the Work

The user will interact with the application at its many different stages. Initial and enhanced user flow diagrams were formed from the original look at how these user actions will affect the system (Appendix O). It shows what actions the registered user can take once logged into the system.

- Find events

  The user can pass events into the "my events" by inviting a friend. The user selects the chosen friend from a generated list. If accepted the event then moves into the "my events" section.
- My events

  From the "my events" section the user can view all details of any of the events they have confirmed to be attending, as well as review any pending invitations.
- Profile

  From the profile the user can update information or search and add friends.

2.4.5 Risks

The main risk to this project is API access to an existing events library. Using real event data of a service that is already in use makes the credibility of the project more valid. Though this leaves room for questioning, the event listing plays a major part in the application. Relying on another service always has its issues; if the service stops working there is a risk of the project being redundant.

This will be avoided by promoting firstly by promoting the application/service to event API providers. As the application can be used with many events listing providers, it insures the application can only continue to grow and develop even at the unfortunate event that an event listing company has to stop operating. Secondly, as this was the highest risk of the project it was developer further as a prototype. Implementing the events API into a basic application frame insured that the greatest risk of the project was eliminated.

For a list of general and added functional project risks please refer to Appendix H risk tables.

2.4.6 Waiting Room

The scope of the project must be considered when looking at requirements; some requirements are not suitable for completion within the time frame specified. Additional requirements from the application developer will be considered for completion only if the core requirements of the project are completed. From the market research survey (Appendix D) a list of additional features was also generated, the selected few that could be beneficial to the application are listed below and will only be brought forward for consideration if all other requirements are completed.

Developer additional requirements:

- User to user chat application.
- Touch gestures on mobile.

Market research additional requirements:

- Notifications of approaching events.
- Exclusive generated hash tag for per event for social media use.

2.5 METHODOLOGY SELECTION

2.5.1 Planning

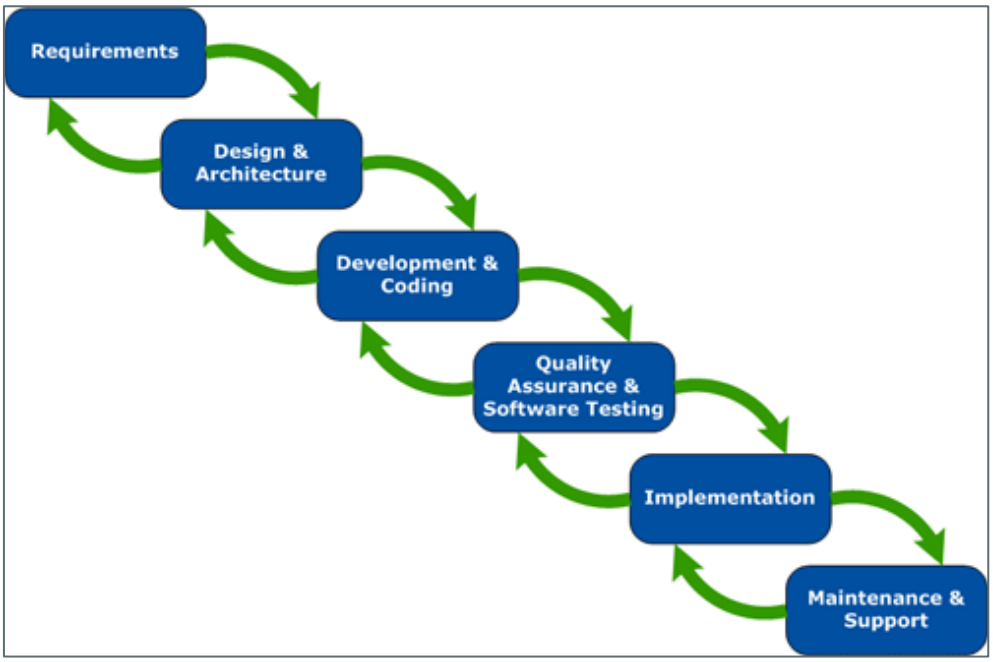
Project planning is key to a successful project, it monitors the schedule and duration of the tasks needed to achieve the project aim. To generate the projects tasks and subtasks the top down approach is the best suited. Taking the over all problem and driving different tasks from it, these tasks can then be managed carefully and applied to a chosen methodology to assure the aim is met.

The estimated duration of the project was decided by a comparison of two factors. Analogy, from previous projects that have included the same amount of complexity it can be assumed the project would take around 9 months to complete. Though with an expert opinion it was suggested two months could be reduced off this time, due to developing time management skills over the pervious year, which were not as disciplined previously.

2.5.2 Methodologies

Due to the nature of the project there were two methods that could be used.
The waterfall or prototyping methodologies were the chosen methodologies for comparison. (Appendix B). Both had their reasons for use, though in the end a compromise of using the modified waterfall was applied. Much like the original waterfall method in its approach of doing tasks chronologically, yet gives flexibility as you can return to a previous task for refining and improvement. This feature of refinement is what highlighted the prototyping method for the projects, after each planned section a refactor stage will take place for any amends needed. The modified waterfalls other quality, of high user engagement would not be suitable for this project as user testing will only occur after the implementation stage.

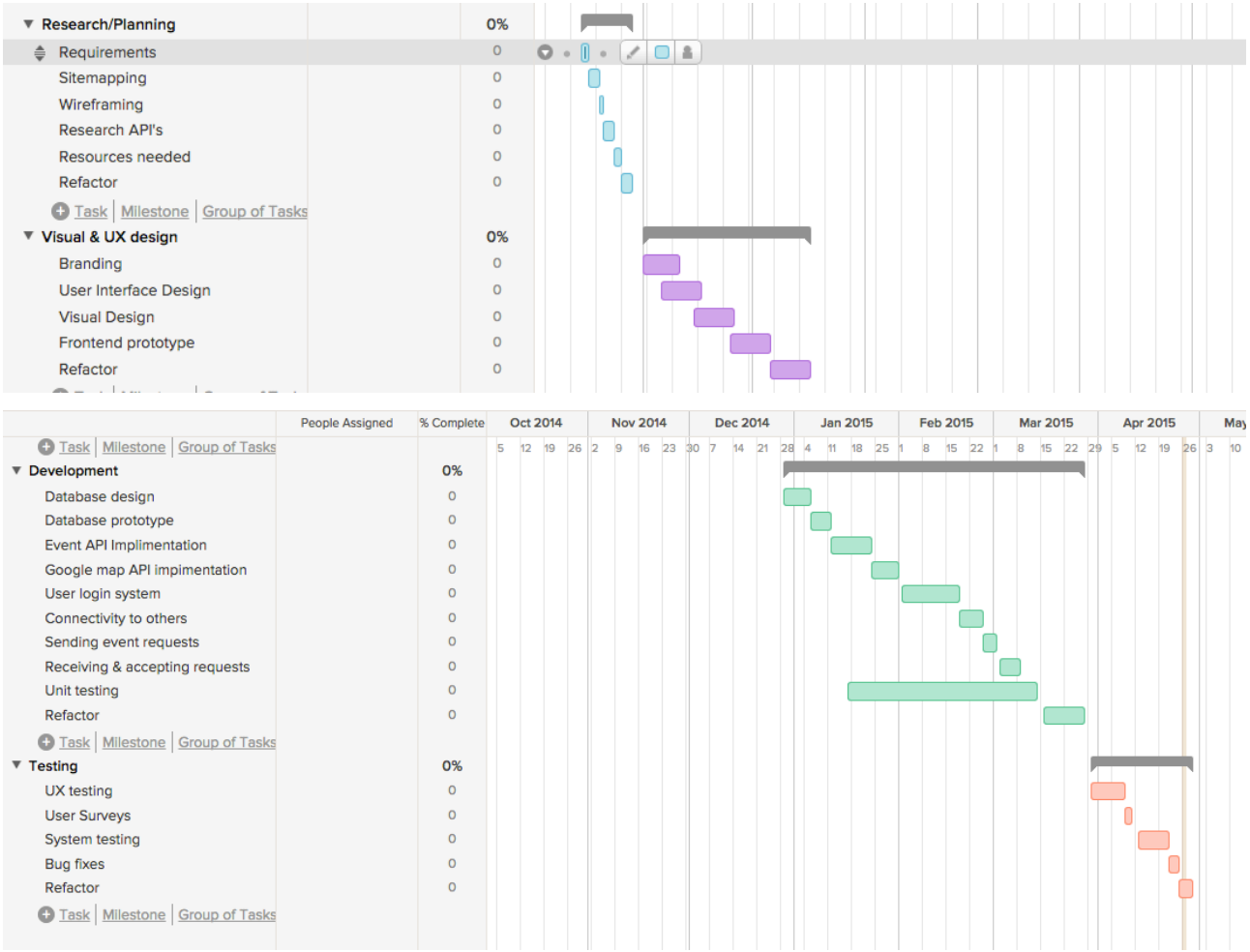
Fig 2.4 – Waterfall diagram



2.5.3 Management Tools

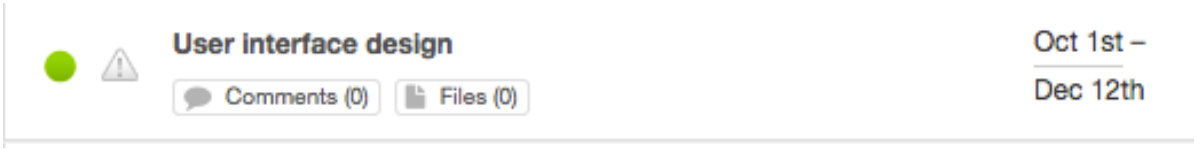
Two tools will be used to manage the tasks for the project; firstly a Gantt chart was created so an overview of deadlines could be visually represented. For Gantt deadline overview see Appendix M.

Fig 2.5 – Gantt Chart



Once the Gantt chart was clarified, these tasks were migrated into a program called Project bubble, see Appendix N. Project bubble is used to make sure that the appropriate amount of time is tracked on given tasks, as well as deadline management with pushing notification reminders. Comments and files can be attached to a project and its tasks, so the entirety of a project can be managed, not just the time frame.

Fig 2.6 – Project Bubble



3 - DESIGN

3.1 UX DESIGN

User experience design is fundamental to any project, insuring the application is usable. Functionality is a high priority, though without the ability for the user to understand and interact successfully with the system the entire project could become redundant.

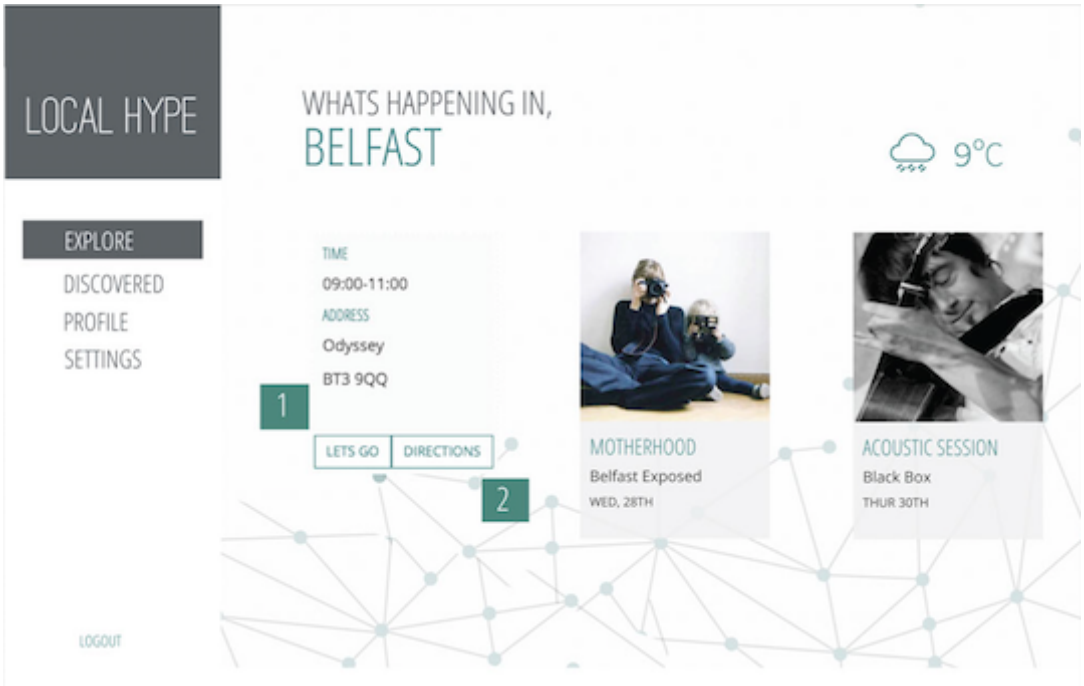
3.1.1 Brand Guidelines

A key part in the process is providing guidelines of how the brand should look. Implementation of brand guidelines guides this process, it defines colours to be used, how graphics should be displayed and typography that should be used. Insuring these rules are adhered to results in consistent designs for the application. Brand guidelines can be found in Appendix R under branding guidelines.

3.1.2 Interface Design

The interface design for the project required designs to be created for all of the pages of the application. Shown below in fig 3.1 is the Explore page of the application. This visual provides a realistic representation of what should be delivered by the developer of the project. For a full list of UX designs for desktop and mobile see appendix R.

Fig 3.1 – Explore UX Design

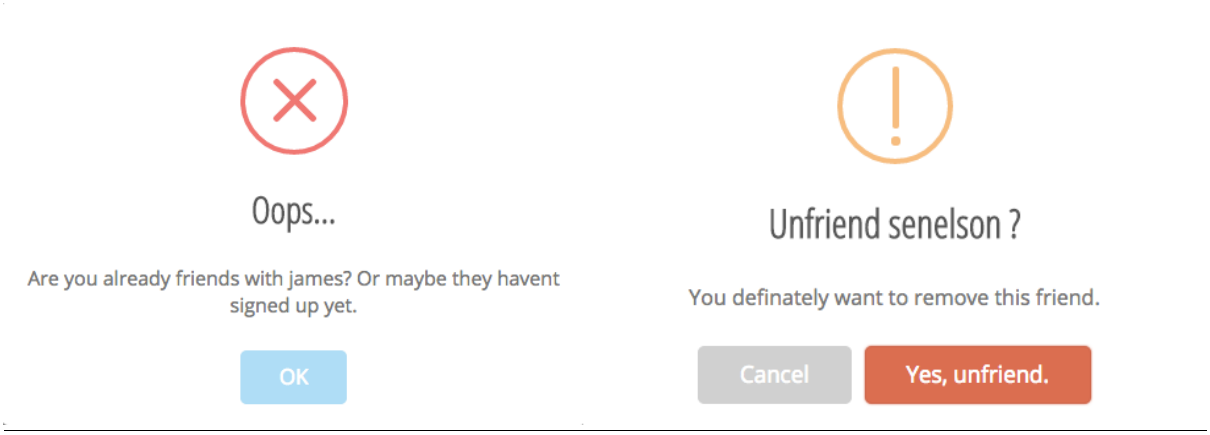


The developer will use these as a guide for how the system will appear to the user. Achieving these visual designs will help to fulfill non-functional requirements #6.1, #6.2, #6.3, #7.1 and #7.2. Once the design implementation is tested for approval these requirements will be successfully completed.

3.1.3 Interactions

As well as visual appearance of the site interactions are key elements to the interface design. Insuring the system implements appropriate responses can help a user interact with the application to a high standard, without these the visual design of the application is redundant, as it doesn't fulfill its requirement to the user.

Fig 3.2 – User interactions



The interactions shown above in fig 3.2 show examples of how the system engages the user. Responses to key tasks allow the user to make sense of the system and what is being asked of them. Insuring the user is aware at all times what action they have just, or are about to preform is as fundamental to the application design as the interface itself.

3.2 SYSTEM DESIGN

3.2.1 Client Server Model

It is important to see how the different aspects of the system will interact with discussed technologies, as these are the main building blocks for the application. See appendix F.

- Client side

  The display will be a browser window showing the application optimised for the discussed range of devices. Most of the returned data will be templated using client side scripting.

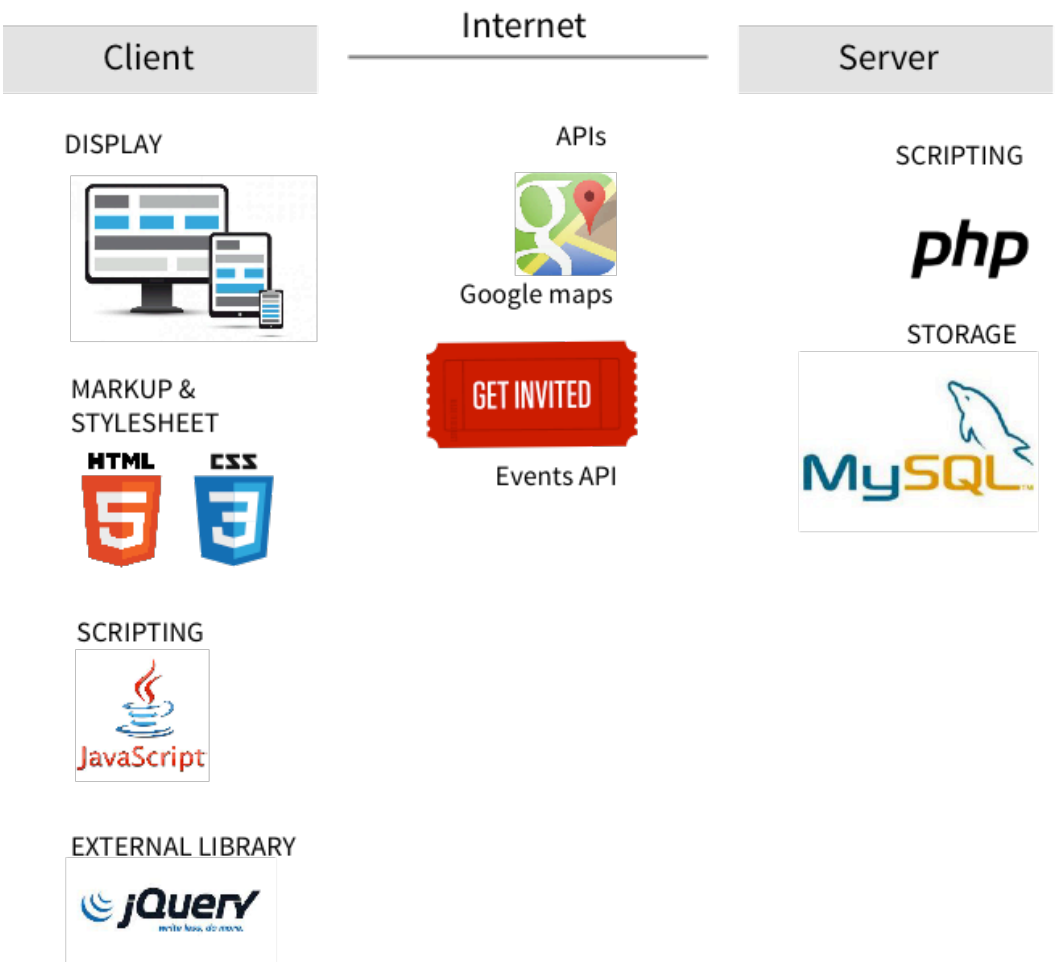
- Internet

  The main APIs of the system fall within the Internet category, as they are an external source. Pulling data into the system apposed to actually scripting on the server.

- Server

  The server side languages will interact with the APIs data it receives for use within the application. Storing relevant data within the MySQL database.

Fig 3.3 – client server model



3.3 LOGIC DESIGN

3.3.1 Model View Controller

From the above mentioned there are some clear technologies that can be associated with the different aspects of MVC. The controller can be several things, most likely JavaScript using event handlers to listen for information to be exchanged. The view is obviously the display screen but more importantly the HTML5 and CSS3 of how the information is displayed on this screen. The model will be the information held within the database and queried using PHP before results are passed back to the view.

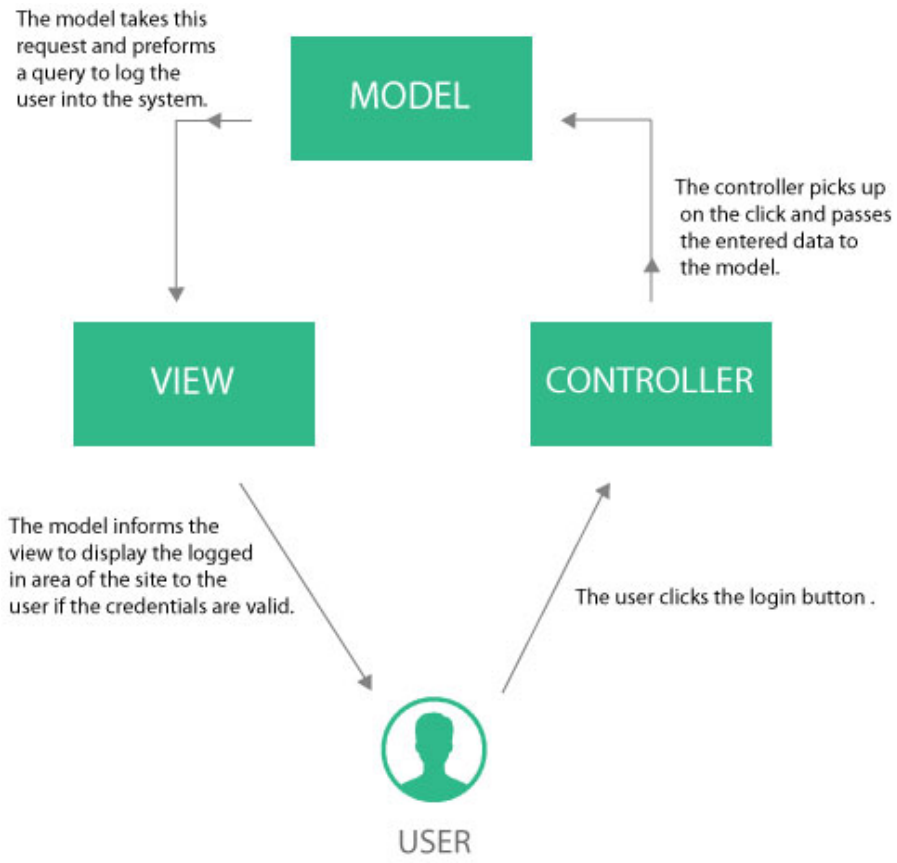
The greatest advantage of using this process within a project is the separation of tasks. It's easier to identify what went wrong and at what stage if the different actions and responsibilities have been place into these three areas.

3.3.2 Using MVC for main functionality

The MVC can help to identify how the functionality of the most important system tasks will operate. Using this method it provides a greater understanding to the developer and as mentioned above allows for a clear understanding of the separation of client and server side tasks. For further experimentation with MVC please refer to Appendix P.

Logging into the system

Fig 3.4 – model view controller



3.3.3 Design Patterns

"Consistency can be something that you can use to your advantage, rather than a straitjacket that constricts you unnecessarily." (Gerry Gaffney)

Design patterns are extremely useful within any project; this may appear in any aspect of the project not solely with the visual appearance of the application.

- General interface patterns

  After interacting with the application, the consistency in the interface will help teach the user what is expected of them. They will experience the same interactions, the same positioning of calls to action such as buttons, and a consistent colour scheme for what those appropriate actions mean. If the button to "add", whether it be adding a profile picture or adding a new invitation, the colour, shape and iconography should remain consistent. This way the user learns exactly what is expected of them when that button is clicked.

- Navigation patters general interface patterns

  Navigational structures within any application should be as minimal as possible and remain the same throughout the application. From position to how the user interacts with the navigation. The one tool a user will interact with the most will be the navigation as it is the easiest way for them to get exactly where they wish to be within the application.

- Header and footer content

  Header, and footer information usually remains the same on every page within an application. A file containing header information and the same for footer information can be included on each of the pages to save repetition of code. Not only does this increase performance of the application it also saves the developer time if there needs to be a change made as it only needs amended once using this method.

- Interactions and animations

  The application will display animations and interactions at certain stages of the user process, from success messages to notifications. Once a general frame is in place for these interactions they should remain the same. Consistency in what is for example a "success" interaction means that when it shows the user will only associate it with what they just did

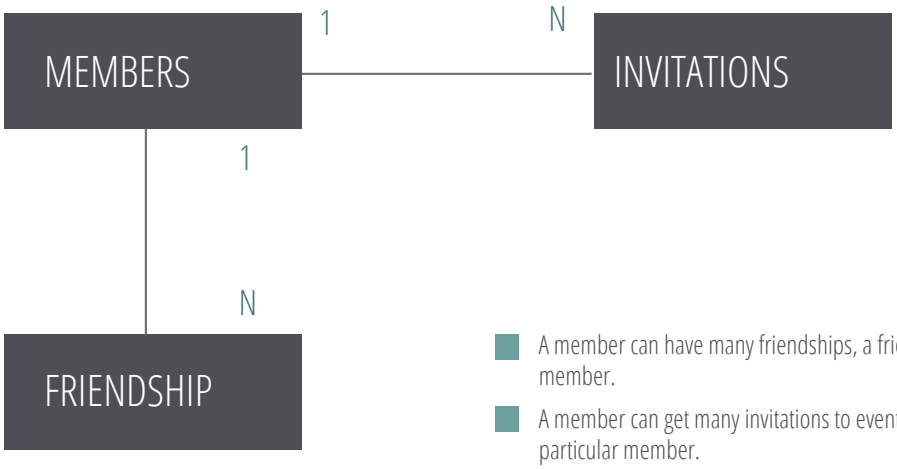
working correctly. This will be extremely useful as it can help teach the user how to use the application, without confusing them.

- Reusable functions and code

  As with the header and footer core functions of the application can be reused, instead of querying similar aspects generic code can be created and populated with the various variables it needs for questioning. This not only means a faster application but an easier editing process for the developer if an issue arises with the program.

3.4 DATA DESIGN

RELATIONAL SCHEMA



- A member can have many friendships, a friendship is with a particular member.
- A member can get many invitations to events, an event is sent to a particular member.

MEMBERS

membersID	INT, NOT NULL, Auto_Increment
username	VARCHAR
profilepic	VARCHAR
userlocation	VARCHAR
password	VARCHAR
email	VARCHAR
active	VARCHAR
resetToken	VARCHAR
resetComplete	VARCHAR

INVITATIONS

invite_id	INT, NOT NULL, Auto_Increment
eventname	VARCHAR
eventdate	VARCHAR
eventimg	VARCHAR
username	VARCHAR
userto	VARCHAR
status	INT

FRIENDSHIP

freindship_id	INT, NOT NULL, Auto_Increment
username	VARCHAR
userto	VARCHAR
status	INT

PK FK

4 - IMPLEMENTATION

4.1 TECHNOLOGY REVIEW

4.1.1 Mark-up Language

Use of HTML5 within the project as it has replaced XHTML. The developer has experience with HTML and XHTML for several years, with this background moving to HTML5 for the project won't be a risk. The next stage of web 3.0 is moving towards HTML5. Combined with the pre-processing stylesheet CSS3 there are some really interesting interactions to be used within web applications.

Outcome: HTML5

4.1.2 Stylesheet

Styling of the application will be done using CSS, more accurately CSS3 as it will style the selected HTML5 markup language.

While here have been developments towards modern pre-processors like Sass and LESS. Research shows that a decline in preference for these new pre-processors. Many feel they over complicate a once simplistic process, CSS was in nature designed to be simplistic.

Outcome: CSS3

4.1.3 Responsive Framework

The scope of the project has to be considered when deciding to implement framework to base the Application off. The survey taken as mentioned before showed a clear decision, with 96% of participants using their mobile device to access the web regularly, this indicated the need for a mobile site. The complexity of the product meant that the application could benefit from a framework. Foundation, Bootstrap and Skeleton were the final considerations. Bootstrap isn't near as customisable as you would expect a framework to be, while foundations had an apparent learning curve, which could compromise the project.

Outcome: Bootstrap

4.1.4 Client side Scripting

The choice of client side scripting is minimal; the only two real competitors are python and JavaScript. The developer on the project has extensive experience with JavaScript and its many libraries. One library in particular being jQuery, it will make the decrease development time of certain features of the site. Python is not a language known to the team and would create risk, as the developer does not have any familiarity with it. JavaScript being the web standard for client side scripting produces a large development following, this makes its choice more beneficial as there are many solutions to similar problems already.

Outcome: JavaScript & JQuery

4.1.5 Templating

Using templating is the most effective way to separate scripting from mark up. This is the chosen method for extracting information from Ajax functions communicating with the server side PHP scripts. The returned data is in json format, and can be interacted with using templating. Instead of appending the response data into the DOM directly templating allows for allocation of the response data to a chosen template. This template will then display out the selected data. The developer decided this would greatly improve the application, as separation of mark up from scripting is a valuable asset. No risk would be added as the developer has experience with templating. Other templating frameworks such as Mustache are available, though the developer has minimal experience with these in comparison to the experience level with jQuery templating. It was decided that as templating is an enhancement feature, that the developer would use the templating framework they are familiar with.

Outcome: JQuery Templating

4.1.6 Geolocation API

The application will require geolocation to target the users current position. There are different mapping applications that provide this as a part of their API. The two most commonly known are OpenStreetMap and Google maps. While OpenStreetMap has been favoured over Google's alternative the option for this project is limited. The data will be passed into the events API for the application and with OpenStreetMaps information being open for anyone to view it's a better choice to move towards Google maps as it integrates with the events API securely.

Outcome: Google maps API

4.1.7 Event listing API

The event listing API will provide a huge part of the applications content; this was the greatest consideration to be made for the project. The two choices boiled down to EventBrite and Get Invited. Both are event organising/ticketing companies. EventBrite while having detailed API documentation is still not a very easy to integrate application. Most feel it to be messy and of a bad structure. Get invited are local to the project developer and after many meetings with the Get Invited team it was decided that Get invited would be a greater suit. Not only as it is a far superior API but due to having the company local in case of queries.

Outcome: Get Invited

4.1.8 Server side scripting

The Server side scripting choice was based on the developer's knowledge. Personal experience of PHP insures the application will not develop a risk of not being completed on time; the main bulk of the applications functionality falls on the server side scripting and therefore cannot be risked. Not only is there a previous knowledge there but also PHP has a large global following insuring that guidance can be provided from looking at other solutions. Ruby on rails though powerful is still a young language and the learning curve would be too vast for this project. Object oriented PHP will also be used for aspects of the project, is more secure for login systems as it helps to prevent SQL injection, for this reasoning it will be used for this purpose and if the scope allows, for other aspects of the project.

Outcome: PHP & PDO PHP.

4.1.9 Server side storage

Server side storage will use MySql, it works efficiently with PHP which is the server side scripting model. Other databases such as MongoDB are becoming more popular yet as they are NoSQL, meaning it is modelled in means other than the tabular relations used in relational databases. This would be impractical to use as the developer only has experience in structuring databases as relational models.

Outcome: MySql

4.2 APPLICATION IMPLEMENTATION

4.2.1 Database Implementation

The first stage of the implementation involved the set up of the database and its tables, as well as considerations of how they interact with each other. The three tables required as defined in the data design were set up. A full list of SQL Queries used to generate the database can be found in Appendix I under database set up.

Fig 4.1 – db connection code

Database connection

The connection to the database was done using PDO PHP; this method is more secured than using the basic mysqli_connect() option. The connection is established and values are passed through, the database name, username, password and port number to be used.

```
//database credentials
define('DBHOST','10.169.0.3');
define('DBUSER','localhyp_lhdb');
define('DBPASS','Clara797');
define('DBNAME','localhyp_lhdb');

//application address
define('DIR','http://0.0.0.0:8888/localhype/');
define('SITEEMAIL','noreply@localhype.co');

try {

    //create PDO connection
    $db = new PDO("mysql:host=".DBHOST.";port=3306;dbname=".DBNAME, DBUSER, DBPASS);
    $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

} catch(PDOException $e) {
    //show error
    echo '<p class="bg-danger">'.$e->getMessage().'</p>';
    exit;
}
```

4.2.2 Registering to the application

Requirement : #1.1, #1.2, #1.3

PHP was the chosen scripting for server side; this was due to the developer's previous experience with this language. A more industry ready approach was suggested to the developer, recommending that using object orientated PHP would not only be a more secure method for logging into the system, but would also be a suitable for the project as the projects interactions already include physical objects: friends, events and invites. David Carr provided a tutorial on implementing an Object Orientated login system; this helped with the understanding of PDO PHP.

It was fundamental to the project to allow users to sign up to the application. The sign up process involved users entering a username, email address and password. If these credentials are valid, (see appendix I under sign up validation) the sign up process will continue. The database will be inserted with: (username, profilepic, userlocation, password, email, active). Username and email are taken from the posted values. The password is represented in the database as a hashed password for security and Profilepic and userlocation are given default values. Lastly the active is represented by a generated activation code for the activation email. See appendix I under sign up

script. An activation link is then sent to the users provided email; the account will only be accessible once it has been clicked. Setting the active to value to “YES”.

4.2.3 Logging in

Requirement :#2.1

Once the user has activated their account using the activation link in the email they are able to successfully log into the system. If the username and email provided match within the members table in the database the user has a session stored in the variable of username. They are then directed to the Explore page (memberpage.php). The full script can be found in appendix I under logging in.

```
if($user->login($username,$password)){

    $_SESSION['username'] = $username;
    header('Location: memberpage.php');
    exit;
```

Fig 4.2 – user login check

4.2.4 Add/change profile picture

Requirement ID: #3.1

When the user signs up to the application they may want to change the default picture being used, this proposed a few challenges. The solution desired would be to update into the image name to the database, while uploading the file into a directory for reference. The uploaded image is taken from the upload form Ajax is then used to pass the file information to the upload PHP script as shown in appendix I under Profile page – profile picture.

Fig 4.3 – Image file type requirement

```
// Allow certain file formats
if($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType != "jpeg") {
    echo "Sorry, only JPG, JPEG & PNG files are allowed.";
    $uploadOk = 0;
}
```

The first issue was if a user uploaded a profile picture with the same name as a previously uploaded image by another user. This would over write the previous image stored in the database causing the uploaded image to show on both accounts. To avoid this happening when the image is uploaded into the directory the file gets renamed to be the username of that user.
Simple image PHP is a script used to manipulate images. After certain validation requests are preformed, such as allowing certain formats shown in fig 4.3. The application will attempt to upload the image to the directory.

The selected image file from the upload form is then passed into the simple image API to preform changes. The image is cropped to be 150x150 and then saved into the selected directory as the username of the logged in user. The profile picture filename is then updated into the database replacing the default image. This is shown in fig 4.4.

Fig 4.4 – Image file type requirement

```
else {
    if (move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], $target_file)) {


        //create new simpleimage
        $img = new SimpleImage();
        // Get uploaded file, resize, crop and change name of the file.
        $img->load($target_file)->fit_to_height(150)->crop(0, 0, 150, 150)->save("../../img/profilepic/member_".$username.".jpg");

        //remove the old file
        unlink($target_file);

        echo "Profile Image Changed";

            try {
                $results = $db->query("UPDATE members SET profilepic='member_".$username.".jpg' WHERE username= '".$username."'");
                }

            catch(Exception $e) {
                echo $e->getMessage();
                die();
                }

                $results->execute();

    }
}
```

4.2.5 Edit personal information

Requirement ID: #3.2

It was key that once the user had signed up to the application that they could update certain profile details. These details can be changed on the profile page under the settings tab.
The values are first taken from the database and template out to the user. This is done using a simple SELECT query to retrieve the username, email and location of the logged in user. A templating, jQuery, PHP relationship overview can be seen in fig 4.7.

Fig 4.5 – Templating the user details

```
<h1>Personal details</h1>

 <script id="userdetails_tmpl" type="text/x-jQuery-tmpl">
            <input type="text" class="form-control" id="profileusername" value="${username}"/> <br/>
            <input type="text" class="form-control" id="profileuserlocation" value="${userlocation}"/> <br/>
```

The user details are retrieved and displayed to the user by placing the username, email and location into the input boxes using the Ajax. The values are set to be the responceData from this Ajax query. Once these boxes are populated with the correct information the ability to update these values is available.

Fig 4.6 – Updating profile details

```
//=======  UPDATE PROFILE DETAILS =======


    $("#profiledetails").on("click",".updatedetails", function(){ //on click

        email = $("#profileemail").val(); //gets the values from the named input boxes
        userprofilelocation = $("#profileuserlocation").val();



        $.ajax({
            url: "includes/profile/updateprofiledetails.php",
            type: "POST",
            data: { username : username, email : email, userprofilelocation: userprofilelocation },
            success: profileupdate,
            error: profileupdateError
        });



        function profileupdate(){
            sweetAlert("profile details have been updated", "Success"); //sweet alert for user confirmation
            $("#profiledetails").html(""); //empties
            getprofiledetails(); //runs for updated results

        }

        function profileupdateError(){
            sweetAlert("Oops...", "Cant update details.", "error");
        }

    });

//////////////////////////////// End
```

When the update details button is clicked from within the mentioned template area the values are taken of the update email and location. Username is retrieved on page load from targeting the session variable of the logged in user. This is key to all Ajax, PHP requests as username is the common value throughout the system. These values are passed to the update profile details script, to view this server side code see appendix I under update profile details. When the details are updated successfully a sweet alert is presented to the user to confirm the action is completed. The details will also be reloaded so they are up to date. This can be see in fig 4.6.

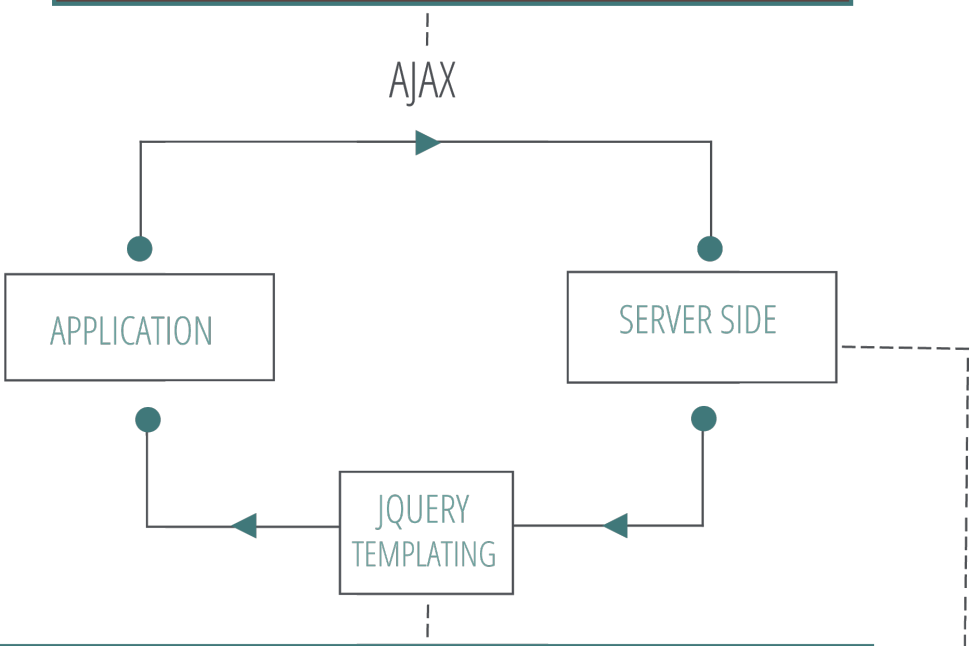
4.2.6 Profile Page – Friends

View friends list

Requirement ID: #3.3

Fig 4.7 – Ajax, PHP, Templating relationship

```
function getMyFriends(){ //get my friends function

    $.ajax({
        url:'includes/showfriends.php', //url the data is sent to
        type:'POST',
        data: { username : username }, //username is passed to this url to be used in query
        dataType:'json',
        success: showFriends,
        error: errorFriends
    });


    function showFriends(responseData){ //responseData from showfriends.php
        $("#friend_tmpl").tmpl(responseData).appendTo("#friend_list");//data passed to jquery template
    }

    function errorFriends(){
        sweetAlert("Oops...", "Cant generate friends list.", "error");  //error message as sweet alert
    }
}
```




```
<div id="tab-inner-panel">

    <div class="profilepanel">
        <h1>Friends</h1>

            <script  id="friend_tmpl" type="text/x-jQuery-tmpl">
                <div class="col-xs-6 col-lg-4">
                    <img class="profilepic" src="img/profilepic/${profilepic}" />
                    <p>${userto}</p>
                    <p>Location: ${userlocation}</p>
                    <button class="deletefriend" type="submit" value="${userto}">Delete</button>
                </div>
            </script>

        <div id="friend_list"></div>

    </div>
```

```
$username = $_POST["username"];


try {
    $stmt = $db->query('
    SELECT members.profilepic, members.userlocation, friendship.userto from friendship
    INNER JOIN members
    ON friendship.userto=members.username
    WHERE status = 1 AND friendship.username= "'.$username.'";
    ');

}


catch(Exception $e) {
    echo $e->getMessage();
    die();
}

$friends = $stmt->fetchAll(PDO::FETCH_ASSOC);
$friends = json_encode($friends);


echo $friends;
```

As shown above in fig 4.7 the users friends are listed out using templating, combined with Ajax and server side scripting. When the profile page is loaded the getMyFriends function is executed, it passes the logged in users username to the URL provided (showfriends.php). The server side script places this value into a variable called username and preforms a query to select the profile picture, location and username of the logged in users friends. Any results are encoded into json and are sent as responseData. This response data is then passed to the jQuery template called friend_tmpl. As shown the template displays the profile picture, username and location for each friend the user has. It also displays a delete button so the user can remove this friend if needed. Listing the users pending friends follows the same method as shown above in fig 4.7.

Search & add new friends

Requirement ID: #3.4

Searching for a friend provided a challenge in its creation. While implementation of searching the database for a user based on the input box value was simplistic the issue resided in not allowing the logged in user to search for a friend that they have already added.

Fig 4.8 – friend search script

```
$usersearch = $_POST["usersearch"];
$username1 = $_POST["username"];

    try {
            $stmt = $db->query('SELECT username, userlocation, profilepic from members
            WHERE username = "'.$usersearch.'"
            AND username NOT IN (SELECT userto from friendship WHERE username = "'.$username1.'")
            ');
        }

        catch(Exception $e) {
            echo $e->getMessage();
            die();
        }

        if ($stmt->rowCount() > 0) {

            $friends = $stmt->fetch(PDO::FETCH_ASSOC);
            $friends = json_encode($friends);
            echo $friends;
        }
```

The code shown above in fig 4.8 shows the finalised solution. Passing through the value that was searched for into usersearch while username is the logged in username. The query involves selecting the searched friends username, user location and profile picture where the username matches the searched for value. The added step is the nested query of “AND username NOT IN” this checks the userto within the friendship table is not equal to the logged in users username. The number of rows returned is then checked, if the value is greater than 0, results are fetched and encoded into json. This information is then template using the method shown in fig 4.7. The Jquery used to do this can be found in appendix I under searching for a friend.

Adding the returned friend

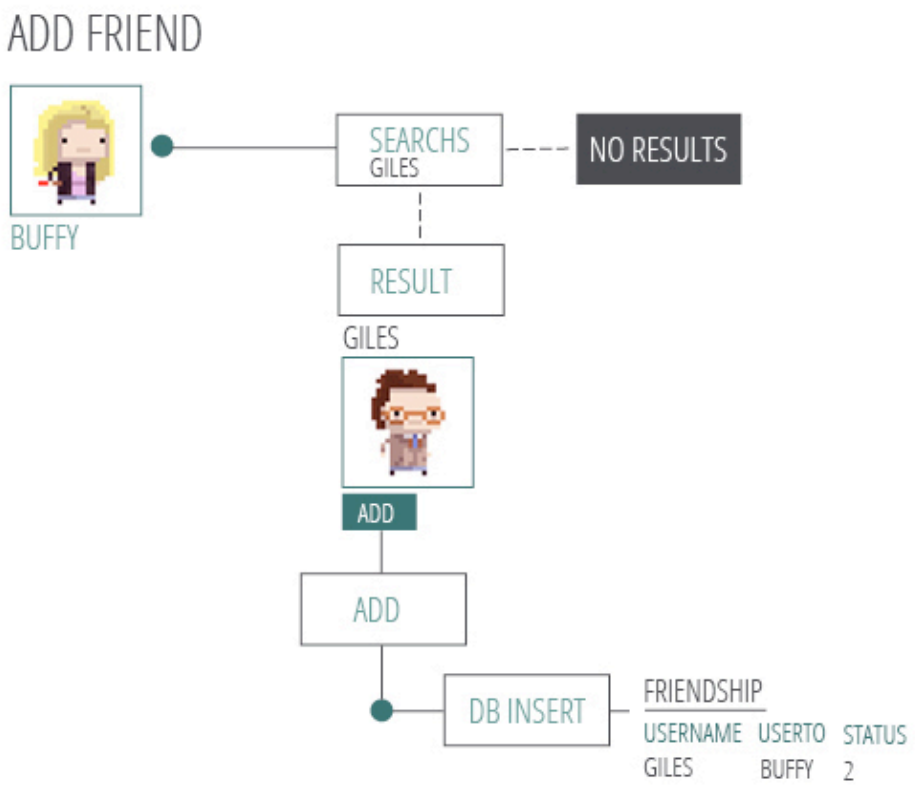
When the friend search returns the friend the user was searching for, the information is templated out again using jquery templating. As shown in fig 4.9 below you can notice how the value of the addfriend button is the username of the returned friend.

Fig 4.9 – templating the returned friend search

```
<script id="searchfriend_tmpl" type="text/x-jQuery-tmpl">
    <img class="profilepic" src="img/profilepic/${profilepic}" />
    <p>${username}</p>
    <p>Location: ${userlocation}</p>
    <button class="addfriend" type="submit" value="${username}">Add</button>
</script>
```

When this button is clicked that value is passed through using Ajax to a server side script that adds the information into the database. This can be seen in appendix I under adding a friend.
Below shown in fig 4.10 explains the user flow when a friend request is sent.

Fig 4.10 – Add friend system flow



Accepting or declining a friend request works similar to the diagram shown in fig 4.10.
A diagram to explain in detail how the accepting or declining of a friend request works is shown in appendix I under accept/decline invitation.

4.2.7 Explore page

Get invited API – Listing Events

Requirement ID: #4.1, #4.2

The Get Invited event listing was the vital part of the application and implementation would be key to the projects success. As mentioned in feasibility the API was used to make a prototype so that the risk of not being able to work with it was eliminated. However this was not the final process of implementing it into the application.

Fig 4.11 – Ajax for geolocation

```
//=======  GET GEOLOCATION FUNCTION =======

$(document).ready(function() {
    $('#listarea').html('<center><br><br><br><br><img src="img/loader.gif"></center>'); //uses a loader gif as the ajax runs
    if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(function (position) { //function to get position

    userlat = position.coords.latitude; //user latitude
    userlong = position.coords.longitude; //user longtitude
    usergeo = userlat +','+ userlong; //combined lat & lng

        $.ajax({
            type: "POST", //posts this user location to the events script to generate events based on location
            url: "events.php",
            data: {
                formlat: position.coords.latitude,
                formlng: position.coords.longitude
            },
            success: function (data) {
               $("#listarea").html(data); //displays the results to the div listarea

            }
            }); //end ajax request
```

As shown in fig 4.11 the first step was to retrieve the users geolocation to be used with the events script. These details are passed to events.php (the full events script can be found in Appendix I under events generation). These posted values are stored into variables and are used within the data array that is used with the Get Invited API class. The values are substituted for hardcoded latitude and longitude values so events are generated based on user location, shown in fig 4.12.

Fig 4.12 – geolocation results used in events.php

```
$lat = $_POST['formlat']; //places the geolocated latitude within variable lat
$lng = $_POST['formlng']; //places the geolocated longtitude within variable lng


//mydata variable includes an array of details needed to generate a list of events

$mydata = array (
  'data' => array (
  'lat' => $lat, //the latitude
  'lng' => $lng, //the longtitude
  'distance' => 10 //distance in miles from the geolocation
      )
  );
```

The returned results are then echoed out within front and back panels of flip cards that the user can interact with. Each event is returned as a result using a foreach loop:
"foreach ($events->results as $result)" Each element of an event result can then be targeted and returned as shown in fig 4.13. Other event results returned can be seen in Appendix I under events generation.

Fig 4.13 – Image result returned from get invited API

```
// IMAGE PLACEMENT
    if (strpos($result->event_logo,'http') !== false){  //checks if the returned result contains "http"
        echo '<img alt="" src=" '.$result->event_logo.'" />'; //outputs the event logo inside img tag
        }

    elseif ( $result->event_logo != '' ) { //if the event logo returned isnt empty place it within the specified file path
        echo '<img alt="" src="https://cdn.getinvited.to/eventlogos/'.$result->event_logo.'">';
        }

    else { //if the event logo returned is empty place the holder graphic
        echo '<img alt="" src="img/holder.png">';
     }
```

The information returned could be displayed easily for the user to interact with, however there were two elements that required greater attention. Fig 4.13 above shows the resolved issue regarding displaying the event image. Get Invited are currently migrating from using their own server to store event images to using Amazon. This above conditional argument searches to see if the $result->event_logo includes "http" if it does the image is echoed as shown. If not it moves on and tries the other server. As an added enhancement if no image exists for the event yet a holder image will be used.

Fig 4.14 – Date result returned from get invited API

```
// DATE CONVERSION
      $timestamp = $result->Start_Timestamp; //returnes the date from the saved time stamp
      echo '<p>' . date('jS F Y', $timestamp) . '</p><br />';
```

The other result returned that need enhancement was the date, fig 4.14. It is stored in the database as a Linux time stamp, which is not a human readable format. The result is stored in a variable and then converted to a human readable format for display to the user. This insures that the project usability of the project is high.

4.4.8 Invite friend to event

Requirement : # 4.3

After the events have been generated into the flip cards the user has the ability to invite a friend to a selected event when the lets go button is clicked. A modal box displays templated results of available friends you have, along with a button that says invite. When this button is clicked it takes the value, which is of the corresponding friends username.

A query is then run to check if the logged in user has already invited this friend to the event.

If no rows are returned from this query the invitation is placed into the database, otherwise an error is displayed to the user stating they have already invited the friend to that particular event.

The code can been found in appendix I under Invite friend to event.

4.2.9 Directions enhancement

Requirement : # 4.4

Google maps API to display directions to the user was a suggested enhancement to the project. Events are to be displayed to the user based on a 10-mile radius of where the user currently is. From here each event would have location details and other information. A directions feature will allow the user to click “directions” displaying a list of directions to the chosen event.

The developer decided that not only would it be appropriate to allow directions by default of driving, but to allow directions by car, bicycle and walking. This was important as the users might be accessing the applications from places that require different types of directions, for example in the city users may wish to walk. Users planning in advance from a more rural area will most likely want directions via car.

Fig 4.15 – CalcRoute function for directions

```
function calcRoute(start, end) {
  var start = usergeo; //sets the start point as the users location
  var end = directions; //sets the end point as the event location
  var selectedMode = document.getElementById('mode').value; //gets value of the selected mode
  var request = {
    origin: start,
    destination: end,
    travelMode: google.maps.TravelMode[selectedMode] //places the selected mode at the end of the travel mode query
  };
  directionsService.route(request, function(response, status) {
    if (status == google.maps.DirectionsStatus.OK) {
      directionsDisplay.setDirections(response); //responds with directions
    }
  });

}
```

The user location is used as a start point, and the end point the event latitude and longitude, which are placed into a directions variable. As you can see in fig 4.15 the value of mode is used to extend the travel mode option. This is selected from a select box on the user interface with the corresponding options having these values. For the full directions code see appendix I under

directions enhancement. There were many useful tutorials online for guidance regarding Google maps API.

4.2.10 Display users current city

Requirement : # 4.5

A personalisation feature of the app will display the current city/town to the user on the login/event listing page of the application. This city/town name will be based on where they are accessing the system. Originally the intention was to use Google maps reverse geocoding API, getting the users current location details and returning a list of string details such as the street name, town name, country etc. The issue with this method is Google maps APIs terms and conditions stipulate that the data returned from a query must be used to generate a map. This API will not be suitable for use within the application for this purpose; the alternatives available are to use another mapping API such as open street maps or some other location based API that can return the current city/town name. If the above alternatives are not available the last resort swill be to use a script that will display the current city name depending on the location the IP address is coming from, this method is the least favourable as it can be unreliable, the IP address is often provided from the Internet service provider, meaning the name it returns may not necessarily be where the application is being accessed from. The solution was reached from using the simple weather jQuery plugin; it accesses Yahoo's Weather API and returns values that can be used such as current temperature, and weather conditions. As it uses geolocation it can also display the city the user is accessing the application from see fig 4.16.

Fig 4.16 – City Name & Weather Display

```
  //========  WEATHER FUNCTION ========


   loadWeather(usergeo ,''); //@params location, woeid

   function loadWeather(location, woeid) { //gets the weather and city name based on geolocation
       $.simpleWeather({
           location: location,
           woeid: woeid,
           unit: 'c',
           success: function(weather) {

           html = '<h1>'+weather.city+'</h1><h2><i class="icon-'+weather.code+'"></i> '+weather.temp+'&deg;'+weather.units.temp+'</h2>';

           $("#weather").html(html);
           },
           error: function(error) {
           $("#weather").html('<p>'+error+'</p>');
           }
       });
     }

   });
  }
}); ////////////////////////////////// CLOSE DOCUMENT READY
```

4.2.11 Discovered – Attending & Invitations

Requirement ID: #5.1, #5.2

The discovered page displays out the events the user is attending and the invitations to events they are yet to confirm. Using the same method as shown in fig 4.7 this information is template out to the user after running a query to the database using server side scripts.

Displaying out the event image was achieved by retrieving the event image name from the database. Originally this proposes a problem, as storing images generated from the Get Invited API would break the terms and conditions, though as it was only the link that was stored this didn't break any conditions agreed to. This image retrieval was achieved by using the method shown in fig 4.13, combining this image name with the path names. See appendix I under event attending, and event invitation for the server side code.

4.2.12 UX features

Bootstrap as the framework

Requirement: #6.3

Frame working would be required as the time frame in which to complete the application was limited and focusing on functionality was more important to the projects success. Using bootstrap was the chosen course of action as it could be streamlined to be a minimal framework for responsive elements only. It was implemented at the beginning of the process and once the functionality above was completed it was customised to match branding guidelines.

4.2.13 API limitations & Design Changes

The original aim was for the application to display further event information on a separate page. This was suggested to remove all the information being displayed at once. The API being basic, and in its initial stages wasn't able to pass queries to display information on separate pages.

The developer discussed the issue with the project mentor and a few alternatives were suggested. An alternative that was experimented with was local storage, though research lead to the discovery that the Filesystem & Filewriter API, which provides a method of reading and writing files to a local file system was only useable in Chrome and newer versions on Opera.

After discussion it was felt users should be able to access all key information from the main page, and from there be able to invite friends.

This proposed a design challenge, as there needed to be a way to display this information to the user on the main event page, but only when this action was required. The first suggestion was to

use jQuery to toggle down a hidden panel that contains all the additional information, after changing the UX designs it was clear to see that this was moving towards the right solution, it was a better layout than the original proposed idea. The issue with this approach was the display to the user was in theory not as practical, the user experience suffered. The final idea was to use a CSS flip on the event information, allowing the event picture and title to be displayed and when hovered on the back panel would be reviled with the additional information. A method used by David Walsh. How this will be visually represented can be seen in appendix R under explore flip card.

4.3 FINAL REMARKS

The main achievement of the project for the developer is the use of the events API. This is the main achievement of the project the developer as they are the first individual to implement the Get Invited API.

Most APIs will contain extensive documentation online, as well as examples and other people developing using them. The Get Invited lead developer provided a document for guidance as well as being available for assistance when needed, though without the ability to just search online for solutions to questions there is a real sense of understanding. The API was integrated without this method of assistance. Not only was the general API implemented, but also adapted to allow for geolocation to generate the event listing. This is a key example of the developer's knowledge of the application they built. Applying the use of an API to a project also enhances the developers experience of developing within restrictions and licencing agreements.

5 - TESTING

5.1 INTRODUCTION

The key part of any application process is the testing that takes place pre and post implementation. Releasing a product without adequate testing could be a risk to the project, if the product is released to the public not responding as expected and the entire product could be discredited, as the end users will feel unconfident with the application. The other consideration is the stakeholders, as they have set key requirements that need to be achieved.
Unit testing was conducted during the implementation to insure the individual units behave as expected. Once the system has been completed and its unit tests reviewed a system test will be preformed to test the application as a whole. Once these developer driven tests have been completed the product will be tested with various user groups relating to the user personas specified.

5.2 TEST APPROACH SELECTION

The application will require several types of testing to successfully test the entire application at various points. Without testing the application would be unpredictable and not fit for release to the public.

5.2.1 Functional testing – Black box testing

Black box testing involves testing a system from an external perspective; this approach allows the application to be tested from an end user experience point of view. With no testing elements of small parts of the system (What is White Box Testing, youtube). System testing is a form of black box testing, as it is an overall system tests once the application is completed its unit tests. System tests will be undertaken for this project.

5.2.2 Structural – White box testing

White box testing, tests the systems internal coding and infrastructure. White box testing focuses mainly, on testing the flow of inputs and outputs through the applications. It is focused on the inner workings of the application and how they operate, insuring these internal parts are operating as expected (What is White Box Testing, youtube). During the implementation these structural tests will be completed.

5.2.3 Unit testing

Unit testing is the process of testing the smallest part of an application to insure that it operates as expected (Zilberfeld). Unit testing is used to test small fractions of your code. One of the big benefits of unit testing is the reassurance to the developer that functionality can be changed and enhanced with ease. As long as the new functionality passes the previous unit test. Unit testing will be carried out for this project at the implementation stage.

5.3 TEST PROCESS

5.3.1 Unit testing

Each small part of the application needed tested. As mentioned above the easiest way to implement this was to test each unit of a particular feature during the implementation process. If all tests were past the developer would move onto the next feature. An overview system test of the application, as a whole would be completed, though only once all unit tests were preformed to check the system is functioning properly as an application.

Fig 5.1 – Unit Case Template

Unit Test ID #: 000	
Component:	The unit of the system that is being tested
Purpose:	Statement of what the unit test is aiming to achieve
Inputs:	Any information used for the test
Expected Result:	What the expected output if test passes
Actual Result:	The actual output of the test
Pass/Fail:	Determines if the test was a success or failure

5.3.2 Usability testing

The usability testing involved getting a sample group of users that match the user personas established for the application. Each of them was required to walk through using the application. They were being monitored to see if the application was easily navigable. As well as being easy to use the interactions they experience needed tested to see if there was an effective method of communication to the user as to what was occurring. Knowing actions have been preformed successfully is of great importance. For the full survey results see Appendix L

User testing User surveys were conducted to gauge what users though of the application, the user groups that were targeted consisted of students, young professionals and parents. It was critical to

get feedback from these groups of individuals, as they reflected the user persona criteria for the applications key audiences. Two individuals were tested from each of the following groups, the survey responses can be found in appendix L.

Key requirements were assessed from user testing:

#6.1 - The app shall be attractive to a young audience

#6.2- The app interface design will be minimal.

#7.1 - The user can accurately operate the application

#7.2 - Consistency of the application

#8.1 - Application will run efficiently on the 3G network as well as Wi-Fi.

5.3.3 System testing

Testing the system commenced once the application has been finalized and both unit and usability tests has been completed. The process involved testing the entire system as a whole rather that individual functions/units or usability features. Testing the application to insure all functions were operating properly, combined with the appropriate UX responses that were to be expected. The database and interface interacting accurately with each other as apposed to assessing them as separate entities, this is the final phase of testing once the individual tests have been preformed.

Fig 5.2 – Test Case Template

Test Case ID #: 000	
Related Requirement:	Requirement that the test is for
Description:	Statement of what the test is aiming to achieve
Test Path:	Detailed instructions of how the test is preformed
Expected Result:	What the expected output if test passes
Actual Result:	The actual output of the test
Pass/Fail:	Determines if the test was a success or failure

Insert, update and delete queries to the database were preformed through the system to validate that the appropriate responses were being delivered. This was easily tested as throughout the project a model view controller approach was taken. Client side scripting interacts with the server side functionality, then returning the results to the user interface. A full list of test cases can be found in appendix K.

5.3.4 Browser Testing

Part of system testing was insuring that the application worked in all major browsers. The application was checked in Safari, Chrome and Firefox and operated successfully in all of the browsers to a high standard. Minor discrepancies such as font weight and size were noticeable but only to the developer and were not commended on in the user-testing period.

5.4 TEST RESULTS

The application operated as expected for most of the tests a full evaluation can be found in the following evaluation section. The three different fundamental tests can be found within the appendix. Evaluation of these test methods used provides proof to determine if the application was a success or not.

- Unit testing in appendix J
- Test Cases in appendix K
- Usability Testing in appendix L.

6 - EVALUATION

6.1 EVALUATE USER SURVEYS

The user surveys (appendix L) displayed some key outcomes that could be used to improve the application as well as prove non-functional requirements to be completed successfully.

6.1.1 – Parents

From the parents user testing there was a clear sense of uncertainty of using the application. Both Edith and John commented on not usually using applications and how it was out of their comfort zone. From the results shown, Edith was more critical with marks and coincidently remarks how she would not use the application again due to this apprehension.
John is more open to trying the technology and states how added support would convince him to sign up to the application.

6.1.2 – Students

Arguably the most critical user group is students, as they are the largest percentage user group as seen in the user persona section. From all questions asked both students didn't mark under 4/5. Requirements #6.1 and #6.2 were achieved as the design and interface interactions gained these high scores, the application was designed with the intention of being minimal and thus appealing to a young audience.

6.1.3 - Young professionals

This is the second most important user group, like the student group they marked the application high on all questions asked. The marks didn't fall under 4/5 apart from in the Efficiency for Clara's results. This was due to her using the application on a 3G network instead of Wi-Fi, which produced marginally slower results.

6.1.4 Overview evaluation

After in depth observations were made form the individual groups responses an overview of the results was undertaken. This was to evaluate the system as a whole instead of being evaluated within the smaller groups.

- From the testing results 5/6 users that tested the application said they would use the application again, requirement #7.1 is passed, as the application is highly useable.
  The consistency of the application did not fall under 4/5 for all users testing, passing requirement # 8.2.
- Comments for the interface design ranged from 3 - good to 5 - excellent.
- Functionality didn't fall under 4/5 for all 6 of the users testing
- The efficiency of the application was also impressive with the lowest mark being 3 – Good. Four of the users tested on a Wi-Fi connection giving the efficiency of the application a score of 4, however Clara and Joanne tested the application using their mobile 3G and they both rated the application 3/5. With the mobile connection still getting a score of good. This confirms requirement # 8.1 is complete.

6.1.5 Suggestions of future development and application enhancements

- Young professional Clara commented how a social media presence would increase the applications reach. Knowing that her friends are interested would make her more likely to use the application.
- Student Joanne mentioned a great idea for a future enhancement, she mentioned how when using her mobile she would only access applications in browser if necessary and there was no native app alternative. This would be a great future enhancement for the project.
- Parent John mentioned how he was slightly nervous when using the application, he suggested having a user guide that can be accessed. This was developed and can be found in Appendix Q.

6.2 EVALUATE TESTING RESULTS

Evaluation of the test results shows that the system is operating to a high standard. During the implementation stage unit tests were preformed. While all unit tests passed apart from unit test #013 it was still essential to undertake system tests once the implementation stage had concluded. As unit testing was carried out during the implementation stage for small units of the system as mentioned above, majority of the results passed. This does not insure that the system is working perfectly as an overall entity. (See appendix J for unit tests). System tests were carried out and highlighted two parts of the system that needed addressed. Once these were corrected the system was operating as expected and all functional requirements were signed off.

- Test case #007, relating to requirement #3.2 failed. Updating the users personal details
  While the server side was operating successfully, the client side scripting had the element of emptying the user details input boxes neglected. Clearing the inputs and running the function that retrieves the user details again on update success easily amended this.

- Test case #013, relating to requirement #4.4 failed. Displaying directions in modal box.
  The mapping element and directions were successfully being listing in the corresponding modal box when required. Though it was not showing correctly. Preforming a map resize when the modal box is activated amended this. Further details can be seen in Appendix K test case #013.

6.3 EVALUATE PROJECT OUTCOMES

The outcome of the project was a success. All requirements were achieved to a high standard, as well as implementation of system enhancements such as directions to selected events. The developer had little experience in understanding the time commitments a project of this scale would involve. With sufficient planning, guided by a chosen methodology the project was achieved with a high degree of satisfaction from all testing completed.

6.4 EVALUATE THE METHODOLOGY

The chosen methodology for the project was a modified waterfall method. This worked perfectly for the project, as unit testing was part of the implementation stage. It insured if something didn't operate as expected there was time allotted to refactor the code and insure that the elements were operating successfully.

The time scale was sufficient, allowing for required learning within the development stage. This insured the change to PDO PHP could be allowed. Without the ability to make suitable changes throughout the project the success rate would not have been as high. Certain changes that were made increased the value of the project, this would only have been possible with the modified waterfall method. Although certain elements of the project provided a technical challenge to the developer the assurance of sufficient time and planned refactor stages allowed for these expected challenges to be over come.

6.5 EVALUATE THE PLAN

The project was scoped sufficiently at the beginning, once feasibility of the application was confirmed a number of stages were planned with the help of the chosen modified waterfall methodology. The developer had little or no experience of planning a project as vast and complex as the one undertaken. Key skills were acquired through this process, such as the ability to scope the project and then insure that it will be completed on time. Each of the stages planned required the time scale to be tracked accurately, if a process exceeded its allotted time the project risk of completion was increased. Learning to work to strict deadlines was a key skill developed from the project, and the most credited towards the success of the application.
Insuring at an early stage to eliminate features that were out of scope was also a challenge to the developer. Many features could have been implemented though without a suitable time scale they would cause the project to suffer rather than excel. Planning involved having to make key decisions of what features were more important to the applications success as a minimum viable product.

6.6 EVALUATE TECHNOLOGIES USED

The technologies used for the development of the application were proven to be a success. The decision to use these specified technologies resulted in the developer being able to build the application to meet its requirements as well as complete the project on time.
The fundamental languages of the application, PHP, JQuery and HTML helped the development time vastly. The developer was able to make enhancements such as moving towards object orientated PHP, as the basics of the language were already acquired. The learning curve of enhancing previously gained knowledge on these technologies was not as large a risk to the project as learning a new development language would be. Increasing the complexity of how these technologies would be used insured the system was as robustly implemented for deployment as possible considering the timescale.

7 - CONCLUSIONS

7.1 INTRODUCTION

In this concluding chapter reflection on the entire project will be completed, assessing and highlighting the areas in which the developer has greatly succeeded. This review of the project will include a reflection from the developer's point of view of what happening during all aspects of the process. Reflection on the role the developer played and the key learning outcomes that have been taken away from completing such a vast and complex project. Lastly future enhancements will be discussed, outlining any functionality that could improve the system, taking it to the next stage of its development.

7.2 REFLECT ON WHAT HAPPENED

The project was a success as the aim and all requirements were achieved. The application was tested thoroughly at all stages to insure it operated as intended, this insured that the project could be deployed for a beta version. Management of the project ran smoothly as the planning stage at the start of the process was done in detail. With all projects there is always room for added enhancements and further validation though as the requirements were met and there was still sufficient time to improve with an added directions feature the project exceeded its expectations. While the developer faced challenges they were overcome with relative ease. No requirements were unfulfilled and a detailed list of future enhancements have been described to help the application reach its next stage of development.

7.3 REFLECT ON YOUR ROLE

The developer role for the project was not to be taken lightly the technical challenges were great, and although the developer had knowledge in most of the technologies used the learning curve was still vast for unknown elements. Taking the knowledge of already known technologies and applying them too more advanced methods has allowed the developer to grow within the industry. As well as developing and designing the project there was also the responsibility of managing the process. The product was achievable, but only if this process was managed effectively. Time management was a key part of the project, for all elements and not just the

development cycle. The developer has acquired key skills in project management, which are vital to industry enhancement. A key achievement was the ability to make decisions to move away from a technology if it surpassed the allotted time for it. Having to manage the process of finding an alternative method was one of the most demanding parts of the project management process.

This was achieved successfully as three fundamental changes were made to the project from the initial planning stage.

- Applying an object orientated PHP approach was beneficial but something the developer had not preformed before. This enhanced the project, but needed time allotted for learning the syntax.
- Issues with the Google API frameworks licence agreement resulted in needing to find an alternative technology/approach to returning the users city location.
- Use of templating to display results from the client side, in response to requests made to the server side technology. This applied a model, view controller approach that is beneficial to the project for maintainability.

7.4 SUGGEST FUTURE WORK

With an application that has a potential for marketability the developer has several points that would need addressed to take this minimum viable product to market.

- The issues discussed within the waiting room would be key features to implement. Developing the ability to have a live chat, so friends that have successfully connected on the system could discuss more event details in greater detail. This feature was out of scope but would add great value to the project.

- Currently the system operates as a web application, the developers skill set made this the best option for developing the application as a minimal viable product. If the application is proven to be a success, consideration of a mobile application to accompany the web app would be appropriate. Depending on the popularity and success of the application as mobile apps cost significantly more to produce

The application functions best currently at desktop size, and although the application is responsive for a range of screen sizes a mobile application alternative would be a great enhancement for the Local Hype brand. As the application is not an informational based website, but rather an application its interactions would operate to a higher standard as a mobile application. This was taken from user surveys as a suggested future enhancement.

This would involve a different skillset to which the developer of the project had. The mobile app would need developed for IOS and android amongst other devices. Other social sites move to developing their applications for mobile once the success rate has been proven.

REFERNECES

Prince Ea - If This Video Doesn't Convince You To Put Down Your Phone, Nothing Probably Will - The Meta Picture. 2014. If This Video Doesn't Convince You To Put Down Your Phone, Nothing Probably Will - The Meta Picture. [ONLINE] Available at: http://themetapicture.com/if-this-video-doesnt-convince-you/. [Accessed 08 October 2014].

Hype! - The Pulse of Your City. 2014. Hype! - The Pulse of Your City. [ONLINE] Available at: http://hypeapp.co/. [Accessed 08 October 2014].

Tramontana, P. (n.d.). Reverse Engineering of Web Applications. Ph.D. UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II.

UWE - Examples. 2014. UWE - Examples. [ONLINE] Available at: http://uwe.pst.ifi.lmu.de/examplePhiloponella.html. [Accessed 06 November 2014].

Introduction to Gathering Requirements and Creating Use Cases. 2014. Introduction to Gathering Requirements and Creating Use Cases. [ONLINE] Available at: http://www.codemag.com/Article/0102061. [Accessed 06 November 2014].

Responsive CSS Framework Comparison: Bootstrap, Foundation, Skeleton. 2014. Responsive CSS Framework Comparison: Bootstrap, Foundation, Skeleton. [ONLINE] Available at: http://responsive.vermilion.com/compare.php. [Accessed 07 November 2014].

Why We No Longer Use SASS or LESS | Notebook | Dumbwaiter | A Full-Service Web Design & Development Firm | Rochester, NY. 2014. Why We No Longer Use SASS or LESS | Notebook | Dumbwaiter | A Full-Service Web Design & Development Firm | Rochester, NY. [ONLINE] Available at: http://dwaiter.com/notebook/11/why-we-no-longer-use-sass-or-less/. [Accessed 20 November 2014].

Gerry Gaffney. Why Consistency is Critical. 2014. Why Consistency is Critical. [ONLINE] Available at: http://www.sitepoint.com/why-consistency-is-critical/. [Accessed 20 November 2014].

Gil Zilberfeld - Why you need unit testing in web development | Web design | Creative Bloq. 2015. Why you need unit testing in web development | Web design | Creative Bloq. [ONLINE] Available at: http://www.creativebloq.com/web-design/testing-times-10134991. [Accessed 02 March 2015].

CIO Staff -How to define the scope of a project - CIO. 2015. How to define the scope of a project - CIO. [ONLINE] Available at: http://www.cio.com.au/article/401353/how_define_scope_project/. [Accessed 25 November 2014].

What is White Box Testing - YouTube. 2015. What is White Box Testing - YouTube. [ONLINE] Available at: https://www.youtube.com/watch?v=3bJcvBLJViQ. [Accessed 02 March 2015].

CSS Flip Animation. 2015. CSS Flip Animation. [ONLINE] Available at: http://davidwalsh.name/css-flip. [Accessed 30 March 2015].

Login and Registration system with PHP - David Carr | Web Developer. 2015. Login and Registration system with PHP - David Carr | Web Developer. [ONLINE] Available at: http://daveismyname.com/login-and-registration-system-with-php-bp#.VUGCjNpVikp. [Accessed 14 February 2015].

Travel modes in directions - Google Maps JavaScript API v3 — Google Developers. 2015. *Travel modes in directions - Google Maps JavaScript API v3 — Google Developers*. [ONLINE] Available at:https://developers.google.com/maps/documentation/javascript/examples/directions-travel-modes. [Accessed 30 March 2015].

Showing Google Map inside a Bootstrap modal window | Netgloo Blog. 2015. Showing Google Map inside a Bootstrap modal window | Netgloo Blog. [ONLINE] Available at: http://blog.netgloo.com/2014/06/02/showing-google-map-inside-a-bootstrap-modal-window/. [Accessed 02 April 2015].

APPENDICES

APPENDIX SECTION A: COMPETITOR RESEARCH

Competitor	Features present	Features not present	Overview
HYPE hypeapp.co	Ability to find events by user location.	Ability to discuss these events.	The application features work well in a large city like London. Within Belfast events occurring within hours would not be suitable. Lack of user interactions supports theory for all in one app.
	Events displayed by specific time frame.	User system to interact with friends.	
	Curates trusted reviews of places.	Connected friends have no ability to review places themselves.	
Eventful eventful.com	Search facility to define the users location.	Tickets purchasing feature is secondary, redirected to other websites.	Application is useful for searching and buying, but not on a social level. No means for discussion with a very formal feel.
	Community page for information on what people are visiting. Feels very forum like.	No real sense of a user journey. Lists of information with no interaction	
Scout scout.me	Search by categories of the type of event you wish to go to.	New product outside of USA. Not as usable.	The integration with native app to get to an event is nice but the event finding service doesn't work as smoothly. Making the location application redundant.
	Integration with native app, directions to the event that you select on the web app.	Key functionality is featured around getting to events. Not finding.	

APPENDIX SECTION B: METHODOLOGY REVIEW

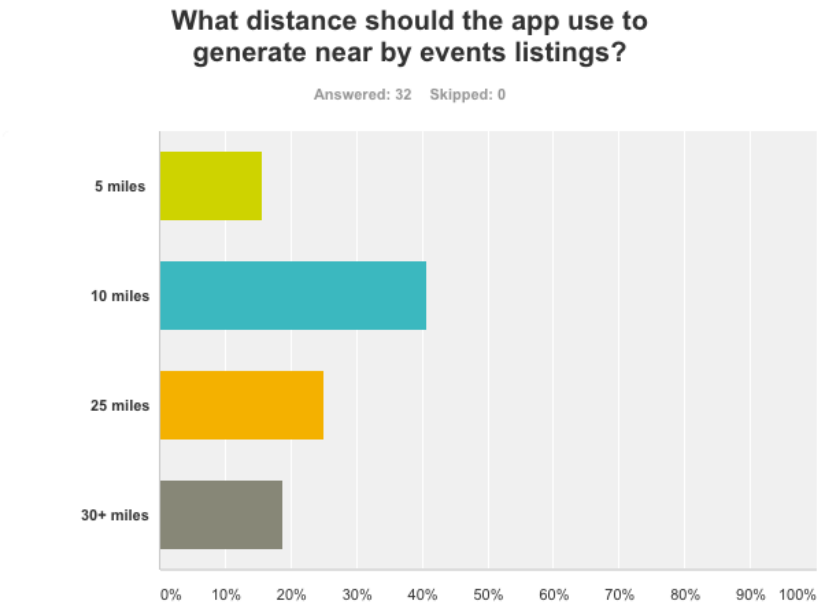
METHOD	PRO	CON
Prototype	Flexible	Complex
	Helps to identify a projects requirements	Managing scope becomes more complex
	Focused around customer satisfaction with end product	Needs high end user engagement which is time consuming
Waterfall	Works well for small projects	Rigid with steps
	Easy to manage	Needs well defined requirements
	Straightforward process	Lack of refining can lead to a unusable product

APPENDIX SECTION C: RESOURCES

Resources	Acquired
Browsers	Safari, Opera, Chrome, Internet Explorer.
Devices	iPhone, iPad, Android Tablet, MacBook pro, 13" & 15", iMac. Windows OS.
Development tools	Photoshop, Illustrator, InDesign, Coda, MAMP.
Other	Sample testing groups.

APPENDIX SECTION D: INITIAL USER SURVEYS

1.1



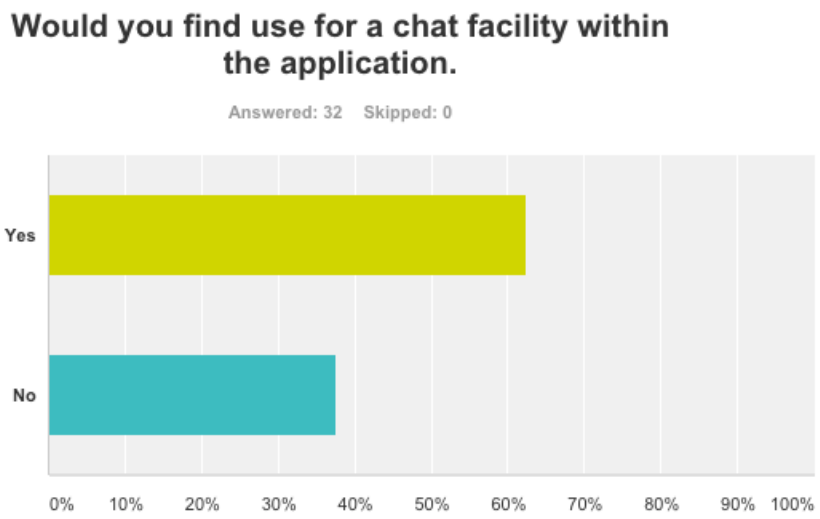
1.2

Answer Choices	Responses	
Safari	**16.13%**	5
Chrome	**74.19%**	23
Internet Explorer	**6.45%**	2
Firefox	**3.23%**	1
Total		31

1.3

	Regularly	Sometimes	Never	Total	Average Rating
Mobile device	**96.88%** 31	**3.13%** 1	**0.00%** 0	32	1.03
Tablet device	**32.26%** 10	**32.26%** 10	**35.48%** 11	31	2.03
Desktop computer or laptop	**65.63%** 21	**34.38%** 11	**0.00%** 0	32	1.34

1.4



1.5

Answer Choices	Responses	
Name of event	**25.00%**	8
Location of event	**31.25%**	10
Date of event	**43.75%**	14
Total		32

1.6

Answer Choices	Responses	
Finding local events	**71.88%**	23
Inviting friends to local events	**28.13%**	9
Organising details about events your attending	**31.25%**	10
Total Respondents: 32		

1.7

Answer Choices	Responses	
Hourly	**3.23%**	1
Daily	**12.90%**	4
Weekly	**51.61%**	16
Monthly	**32.26%**	10
Total		31

1.8

Exclusive promos/special offers for the event through the app.
11/17/2014 3:36 PM View respondent's answers

Instant sharing
11/16/2014 10:56 PM View respondent's answers

Photo sharing
11/16/2014 5:48 PM View respondent's answers

A filter for different typeS of events e.g. Music, party etc
11/16/2014 5:39 PM View respondent's answers

Availability or location of parking
11/16/2014 5:14 PM View respondent's answers

A hashtag feature for content written on photos taken at the event, so you could look back at everything that was written about the event using the hashtag.
11/16/2014 4:38 PM View respondent's answers

notification of event you are attending

1.9

Share to facebook
11/16/2014 4:00 PM View respondent's answers

special offers, which websites to visit in order to get the most out of your visit to the location
11/16/2014 3:57 PM View respondent's answers

lets you add an event your organising
11/16/2014 3:43 PM View respondent's answers

Links to sold out ticket ads
11/16/2014 3:14 PM View respondent's answers

Updates on the event (e.g. If the event time changes or the event is cancelled I would like a notification)
11/16/2014 1:46 PM View respondent's answers

A calander notification
11/16/2014 1:33 PM View respondent's answers

APPENDIX SECTION E: REQUIREMENTS SPEC

FUNCTIONAL REQUIREMENTS

Requirement ID: 1.1

Description: Creation of username

Rationale: Generation of a username to allow user to access the application and their account. A unique username is required to prevent other users information to be displayed.

Fit Criterion: Validation against other username to insure a unique choice is provided.

Conflicts: None

Requirement ID: 1.2

Description: Creation of password

Rationale: Password creation to be used with username to login to account and provide access to the application.

Fit Criterion: Password must be of 8 characters and entered twice for validation.

Conflicts: None

Requirement ID: 1.3

Description: Entry of email for user creation.

Rationale: Email provided on sign up so users can reset password if forgotten.

Fit Criterion: Validation of email address to include an @ symbol and at least one ".". As well as characters and numbers.

Conflicts: None

Requirement ID: 1.4

Description: Creation of default image and location name.

Rationale: Creation of a default image and location name of earth allows the user to register to the application without having to enter excessive details. These can be altered once logged into the system.

Fit Criterion: Validation to insure correct details are uploaded as defaults.

Conflicts: None

Requirement ID: 2.1

Description: Login to application

Rationale: Entry of username combined with a password gives user access to the application and their profile details.

Fit Criterion: When details are submitted check if entered username and passwords match and are present within the user database.

Conflicts: #1.1, #1.2

Requirement ID: 3.1

Description: Add**/**Change profile picture

Rationale: Ability to add a profile picture to an account, and then edit/update said picture.

Fit Criterion: Upload graphic of any size or format to database.

Conflicts: #2.1

Requirement ID: 3.2

Description: Edit personal information

Rationale: Ability to add/edit all person information displayed on the profile page to insure data is up to date.

Fit Criterion: Test update function on database to overwrite pervious data held.

Conflicts: #2.1

Requirement ID: 3.3

Description: View friends list

Rationale: Ability to view friends list, friends can then be added to associated users and used to interact with inside the application.

Fit Criterion: Database query to display a list of friends the logged in user is currently connected with.

Conflicts: #2.1

Requirement ID: 3.4

Description: Search & add new friends

Rationale: Search within the application and add friends to the friends list, these friends can then be interacted with to attend events.

Fit Criterion: Display any profiles matching searched usernames or names that appear within the user database.

Conflicts: #2.1

Requirement ID: 4.1

Description: Generate listing of events

Rationale: Display a list of events for the user to interact with.

Fit Criterion: Access the Events API to generate a list of events based on location.

Conflicts: #2.1, #4.2

Requirement ID: 4.2

Description: Determine users Geolocation

Rationale: Locating the users location will be used to display events from the area they are in.

Fit Criterion: Browser gets geolocation, the users coordinates are passed to the events API.

Conflicts: #2.1, #4.1

Requirement ID: 4.3

Description: Invite friend to event

Rationale: Friends/connected users can be sent invitations to attend events with the logged in user.

Fit Criterion: Event details will be sent to connected users.

Conflicts: #2.1, #3.4, #4.1

Requirement ID: 4.4

Description: Directions to events

Rationale: A key featured would be the ability to allow users to not only view information of events but also get directions to the selected event.

Fit Criterion: Usability feature of not needing to leave the application to get directions

Conflicts: #4.2

Requirement ID: 4.5

Description: Display users current city name

Rationale: A personalisation feature of the application is to display the users current city name when they are on the explore screen.

Fit Criterion: Displaying the users current city will provide an added user experience

Conflicts: #4.2

Requirement ID: 5.1

Description: My events section - Attending

Rationale: Ability to view the events the logged in user has confirmed to attending.

Fit Criterion: Query and display a list from database of confirmed events to the user.

Conflicts: #2.1, #4.1, #4.3

Requirement ID: 5.2

Description: My events section - Invitations

Rationale: Ability to view the events the logged in user has pending for approval.

Fit Criterion: Query and display a list from database of pending events to the user.

Conflicts: #2.1, #4.1, #4.3

NON- FUNCTIONAL REQUIREMENTS

A. APPEARANCE REQUIREMENTS

Requirement ID: 6.1

Description: The app shall be attractive to a young audience.

Rationale: The main target audience is students and young professionals, insuring the application appeals to them can help towards the applications success.

Fit Criterion: Over 70% of young users after testing the application will feel the appearance to be attractive.

Requirement ID: 6.2

Description: The app interface design will be minimal.

Rationale: Tourist users will be able to interact with the application, as they will have only the key/minimal information that is needed.

Fit Criterion: Testing with a large range of users over 60% will feel they can operate the application successfully.

Requirement ID: 6.3

Description: The app will match branding standards

Rationale: Consistency between the logo and graphic branding of the application and the interface itself will provide a rounded experience of the product.

Fit Criterion: With the first impression of the application over 70% of users will feel the application is reliable and trustworthy.

B. EASE OF USE REQUIREMENTS

Requirement ID: 7.1

Description: The user can accurately operate the application

Rationale: A minimal interface will insure users with any technical ability will be able to operate the application easily.

Fit Criterion: Testing user walk through with the application to ensure the actions are clear and concise.

Requirement ID: 7.2

Description: Consistency of the application.

Rationale: A consistent application will engage users to continue using the app as they find it easy to navigate around, with consistent interactions.

Fit Criterion: After the first use of the application the users feel confident in the layout of the interface.

Requirement ID: 8.1

Description: Application will run efficiently on the 3G network as well as Wi-Fi.

Rationale: Minification of CSS will insure the applications load time will be more efficient, allow for a more effective experience for users.

Fit Criterion: Program will be used to preform minification of CSS before application is launched.

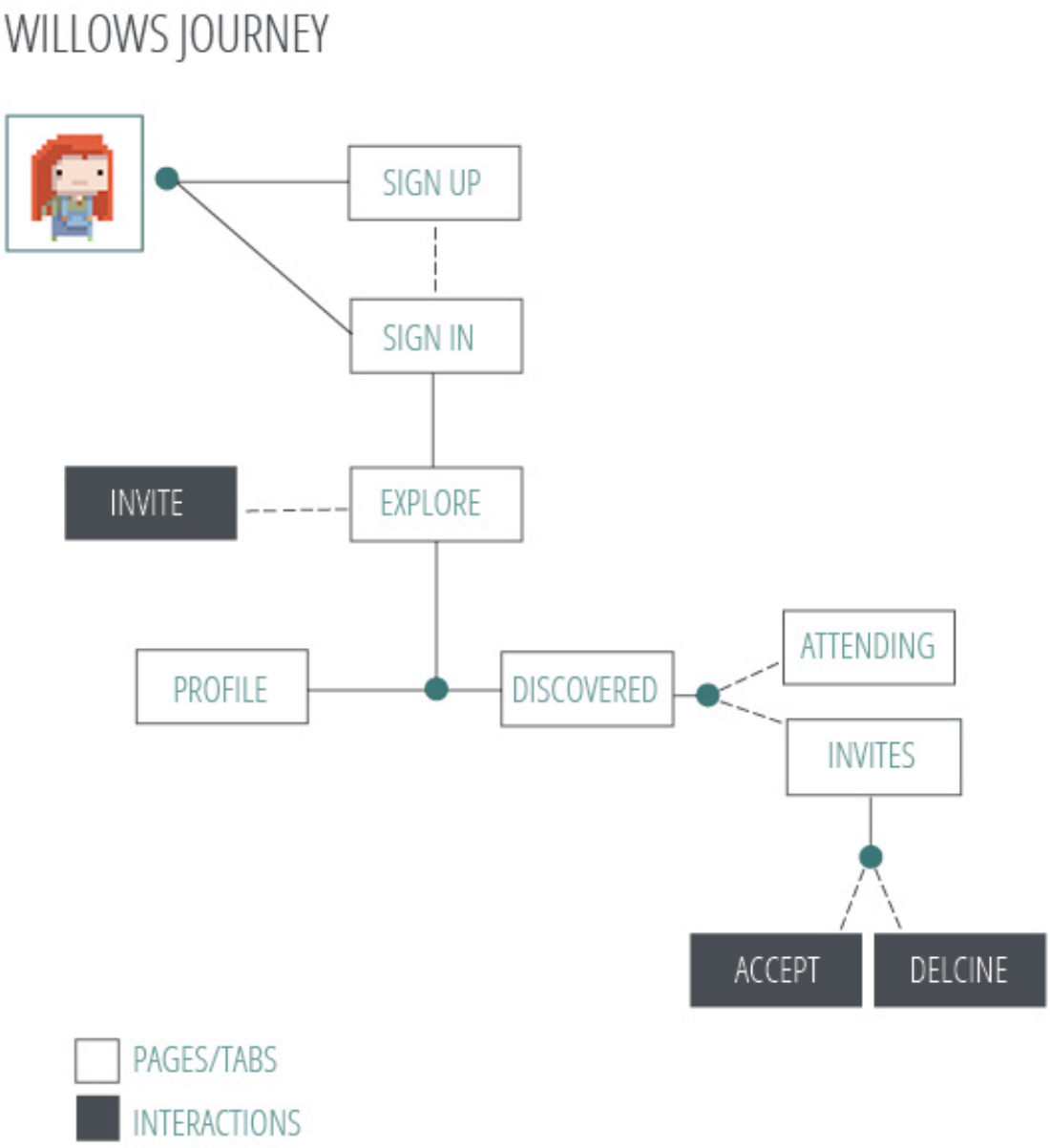
Requirement ID: 8.2

Description: Images will be optimised.

Rationale: Optimised images will decrease the load time of the application and therefore increase user support.

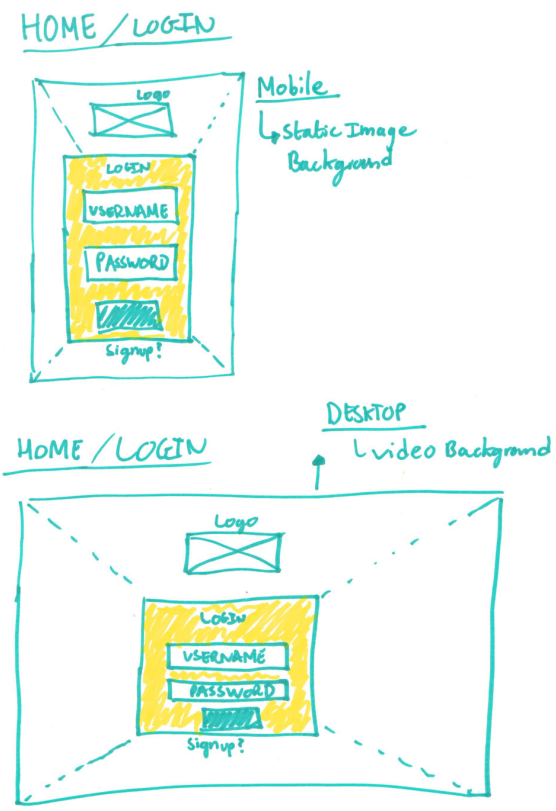
Fit Criterion: Optimisation of images through a compression programme will increase loading time in comparison to uncompressed images.

APPENDIX SECTION F: SITE MAP

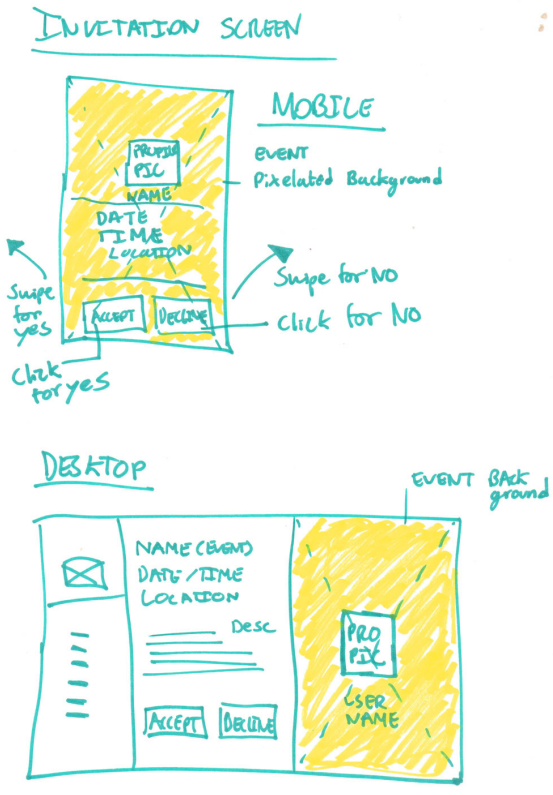


APPENDIX SECTION G: INITIAL UX PROTOTYPING

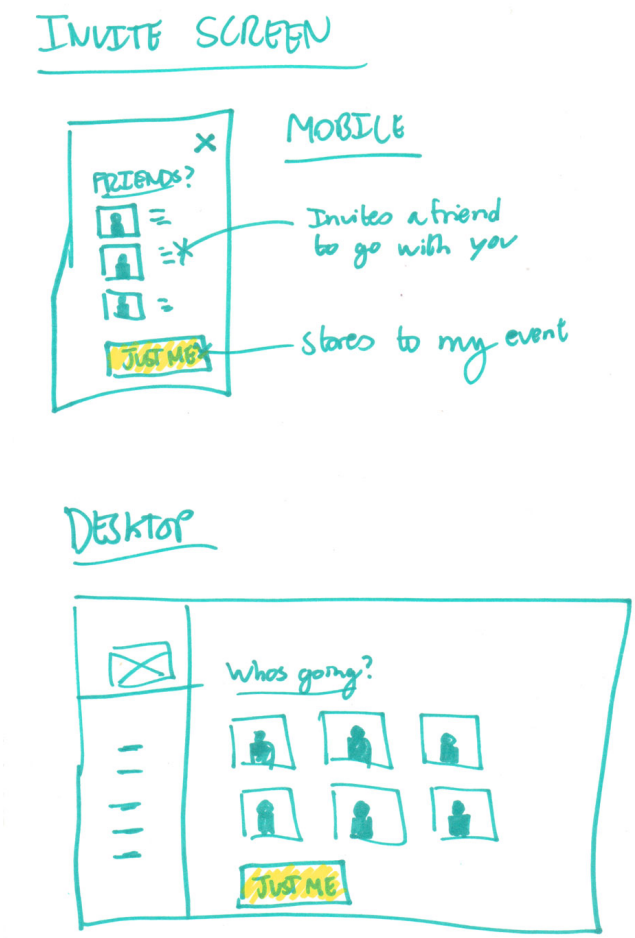
LOGIN PAGE



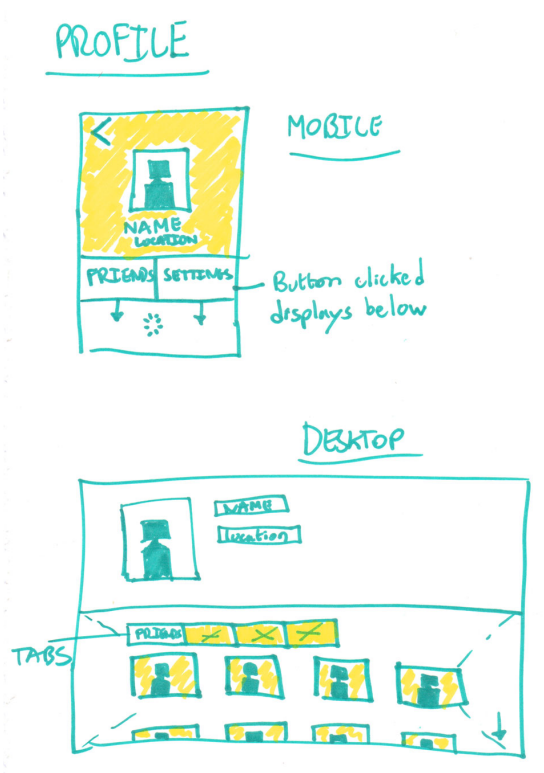
INVITATION SCREEN



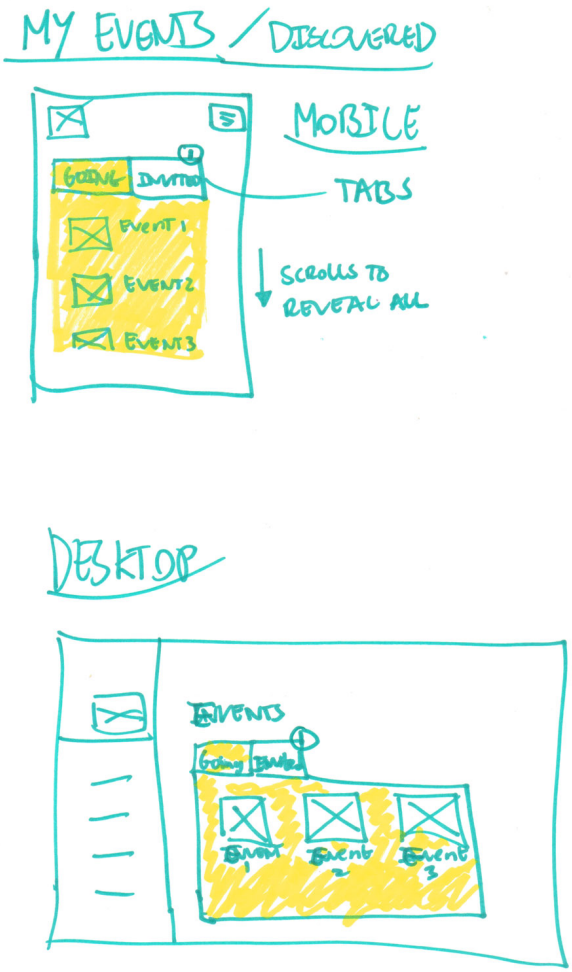
INVITE SCREEN



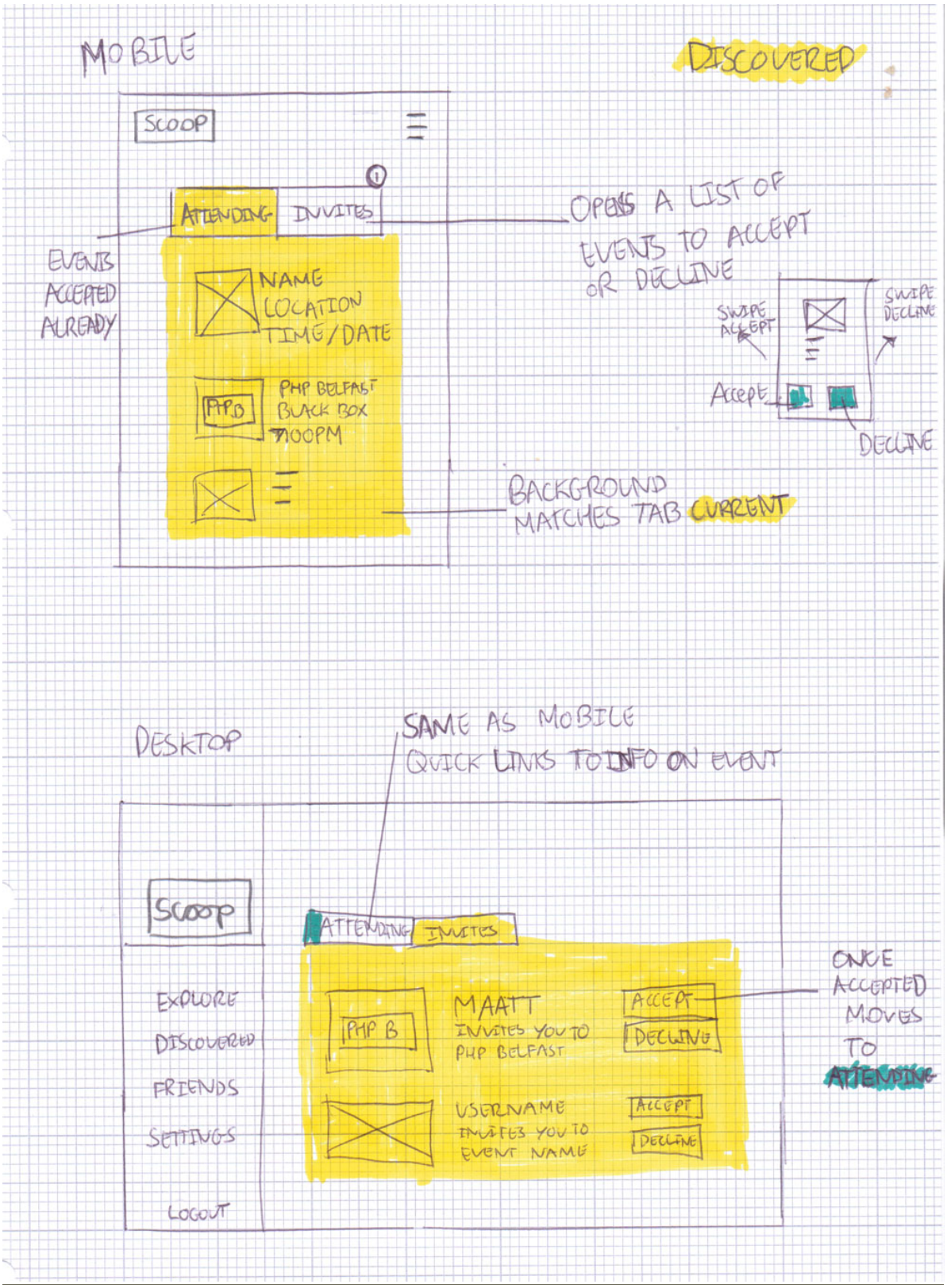
PROFILE PAGE

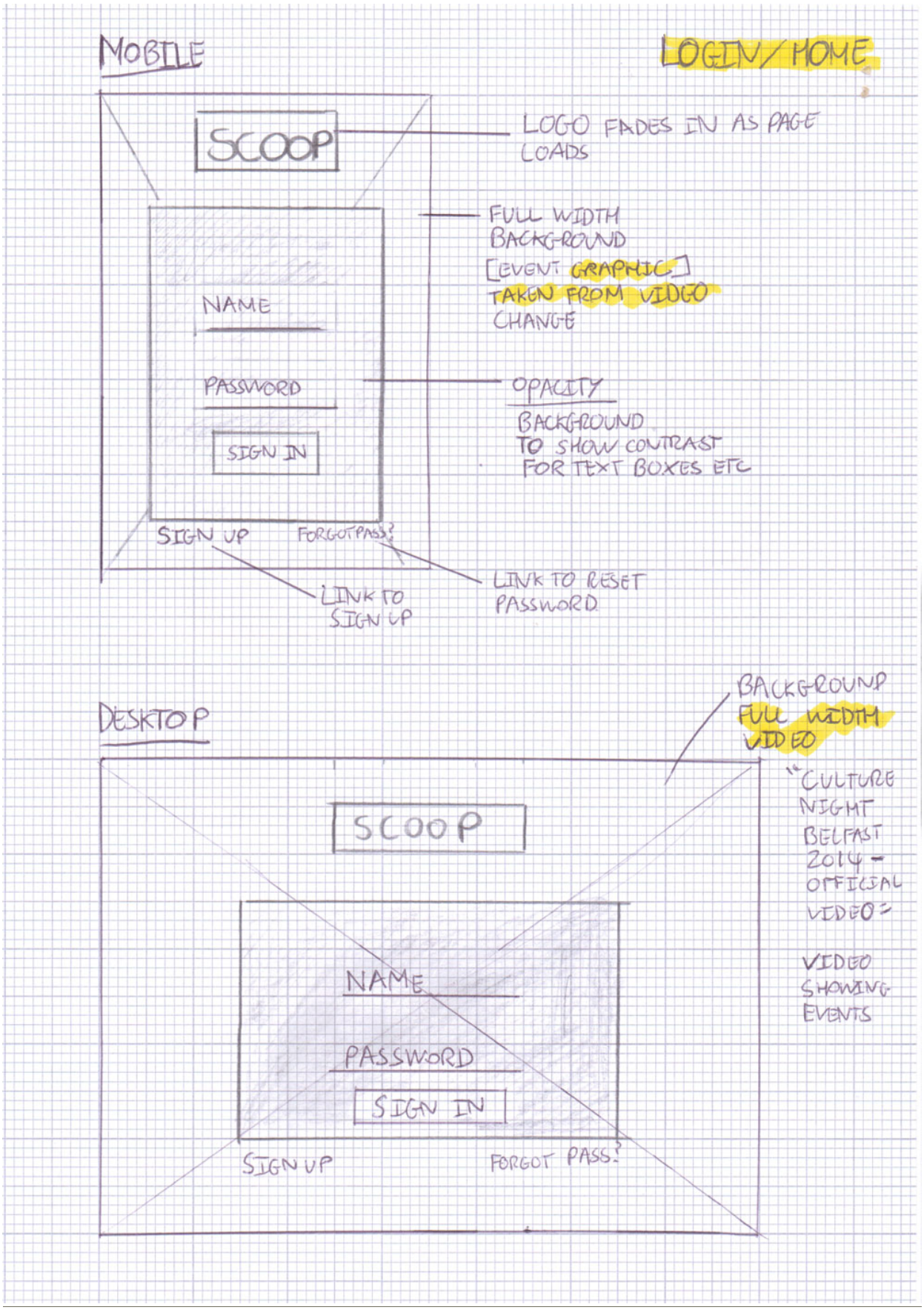


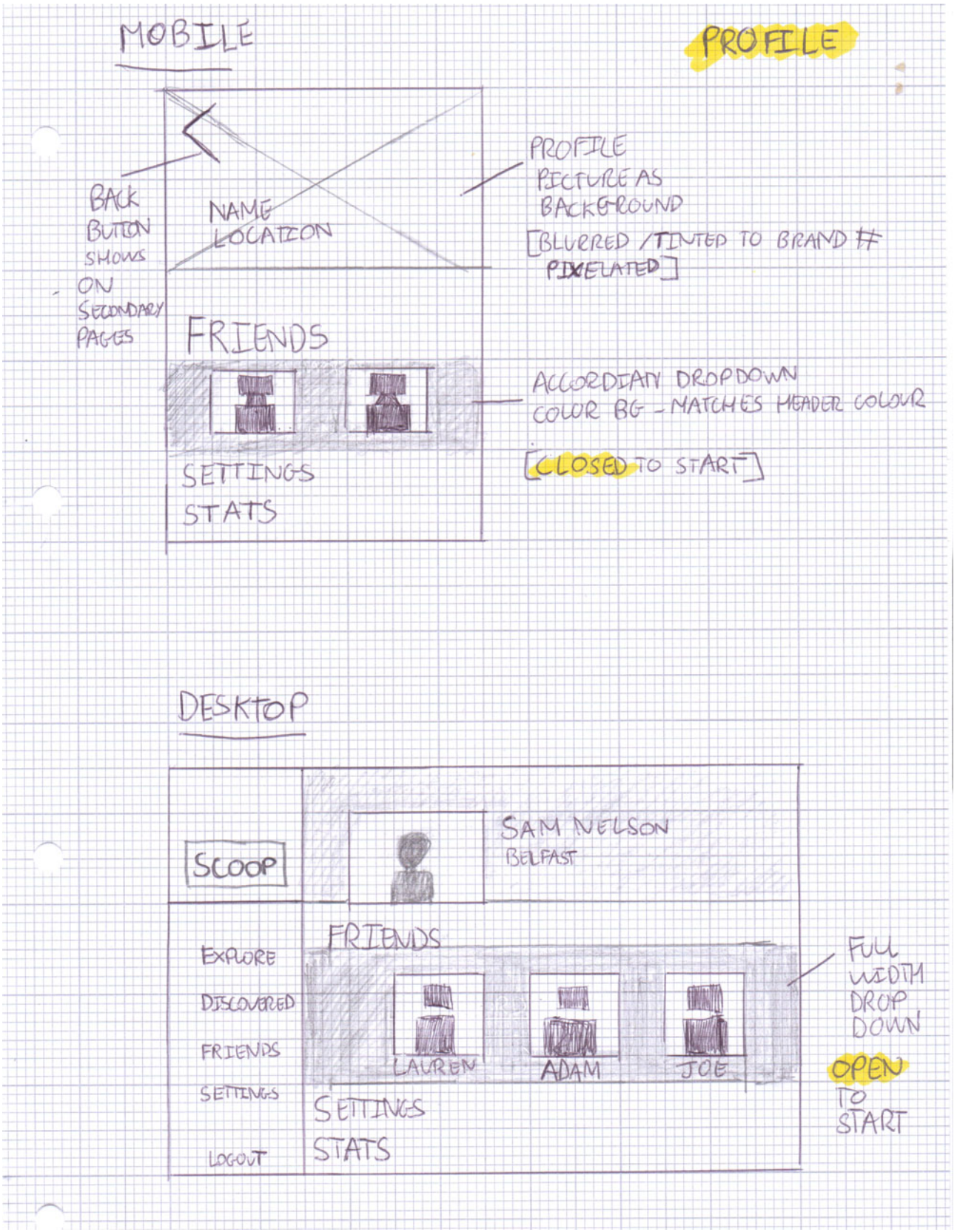
DISCOVERED/MY EVENTS PAGE



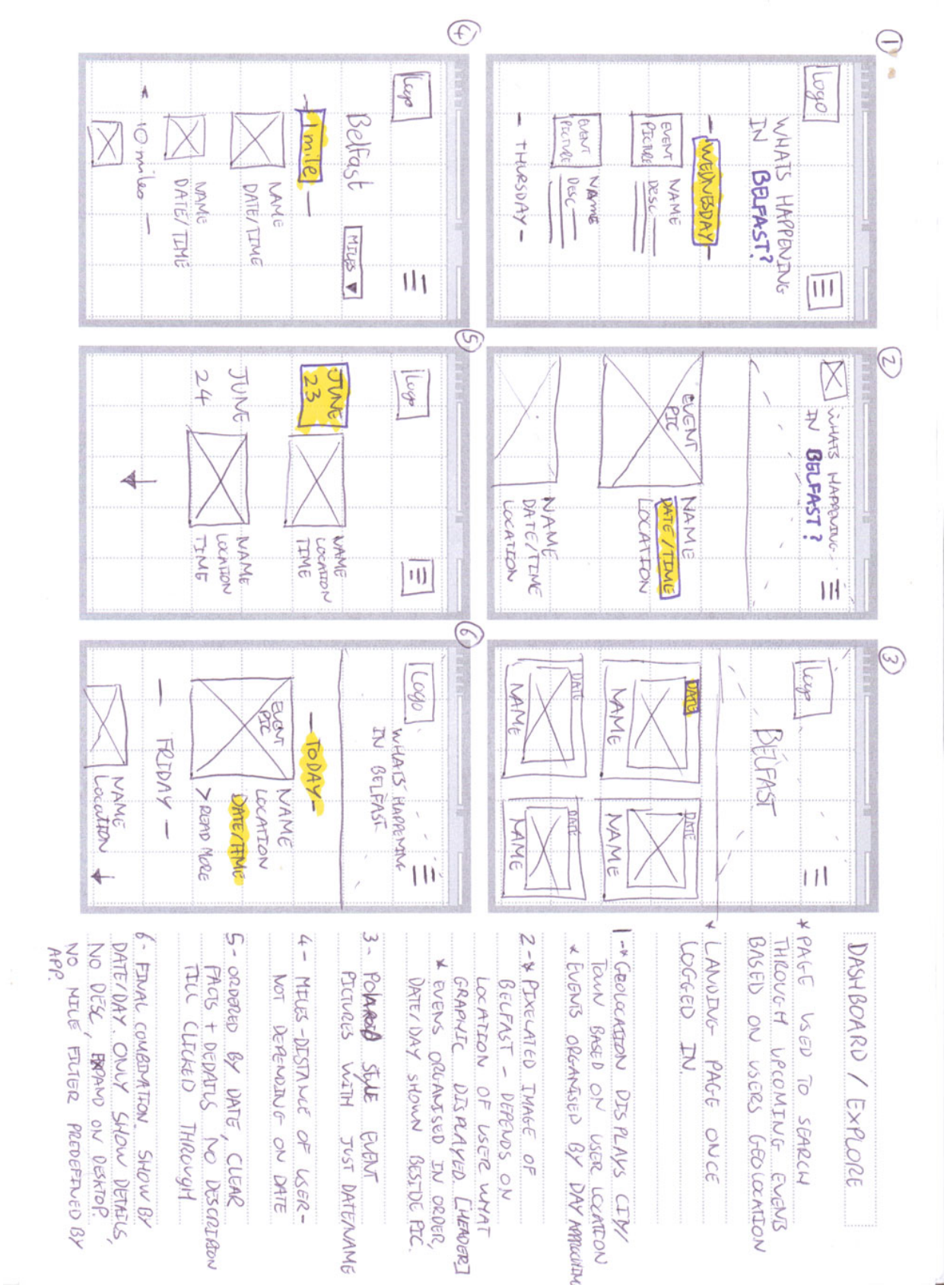
HIGH FIDELITY




MOBILE
LOGIN/HOME
SCOOP
LOGO FADES IN AS PAGE LOADS
FULL WIDTH BACKGROUND [EVENT GRAPHIC] TAKEN FROM VIDEO CHANGE
NAME
PASSWORD
SIGN IN
OPACITY
BACKGROUND TO SHOW CONTRAST FOR TEXT BOXES ETC
SIGN UP
FORGOTPASS?
LINK TO SIGN UP
LINK TO RESET PASSWORD
DESKTOP
BACKGROUND FULL WIDTH VIDEO
"CULTURE NIGHT BELFAST 2014 - OFFICIAL VIDEO"
VIDEO SHOWING EVENTS
SCOOP
NAME
PASSWORD
SIGN IN
SIGN UP
FORGOT PASS?


MOBILE
PROFILE
BACK BUTTON SHOWS ON SECONDARY PAGES
NAME
LOCATION
PROFILE PICTURE AS BACKGROUND
[BLURRED / TINTED TO BRAND # PIXELATED]
FRIENDS
ACCORDIAN DROPDOWN COLOR BG - MATCHES HEADER COLOUR
[CLOSED TO START]
SETTINGS
STATS
DESKTOP
SCOOP
SAM NELSON
BELFAST
EXPLORE
DISCOVERED
FRIENDS
SETTINGS
LOGOUT
FRIENDS
LAUREN
ADAM
JOE
SETTINGS
STATS
FULL WIDTH DROP DOWN
OPEN TO START

6 UP - EXPLORE PAGE



6UP - PROFILE PAGE

SCOOP STYLE TILE

Scoop

Style Tile
version: 2
Logo font : Akadora

Possible Colors

#45403f #ffffff #d95b44

Textures

NO HOVER

ON HOVER

This is an example of a header
Font: Didact : Regular

This is an example of a sub header
Font: Didact : Regular

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel feugait nulla facilisi.

Font: Source sans pro : Light

Adjectives

Clean Flat
Quirky Bold

APPENDIX SECTION H: RISKS

GENERAL RISKS

RISKS	STEPS TO AVOID
Going over time	Time management software in place to track tasks and make sure nothing runs over time.
Unusable product	User requirements survey and user testing of final product to insure the system will be simple and effective for end users.
API integration too complex	Time is dedicated at the research stage to familiarize how API's will be used later in the project.
Resource Shortfalls	Having a predefined list of resources already acquired and only provide initial testing on them.
Architecture is not fit for purpose	Having planned technical research, such as database set up and development methods. This should insure the system is designed in an effective usable way.

FUNCTIONAL RISKS

RISK	LEVEL	ELIMINATION	BACKUP
Get Invited API	10/High	This risk is the first to be tackled, as it will be used for the applications main purpose of pulling in events. A prototype will be developed using the Get Invited API, thus removing any concerns about developing with it. Without this functionality the application will not be functional.	If the risk can't be eliminated, and the API becomes too complex to work with there are many alternatives. EventBrite is the most common with many online tutorials so it will be used as a suitable replacement.
Secure login	8/High	Having a secure login for the application not only provides the platform in which the app will operate but also provides user trust for the application. Apps without secure logins do not have a user base. The developer has developed many secured login systems therefore having a reference point as well as sufficient	A basic login system can be used as a back up until a more secure login can be developed. The application can be made as a minimum viable product and therefore not require an extremely secure interface.

		knowledge.	
Ability to add friends	6/ Intermediate	Searching and interacting with friends is another key element of the project. Creating user connections has been done previously. With the chosen language of PHP not only will there be a base knowledge but a large community online of people making similar functionally.	With a scheduled time frame based on a similar task on a different project this functionality will be achieved. Allowing for extra time will insure that it is completed.
Inviting friends to events.	5/ Intermediate	Once the previous risk of being connected to friends within the application is completed the next most important feature is the ability to invite these friends to the chosen event. Tutorials online of how many people have implemented this functionality exist.	This risk needs to be developed and an alternative can't be suggested. Allowing suitable time to develop this will help eliminate the risk.
Profile Creation	4/Low	Profile creation will allow the user to personalise their experience within the application. Queries to a database, to update user information has been completed on previous projects.	If the profile creation falls out of scope with time spent on higher risks the alternative will be to provide a more basic profile system with less customisation for the user.
Google map API	3/Low	Working with the Google map API to preform reverse geocoding will provide the welcome message that states what city the user is currently in. Tests have been preformed with the API and working examples exist already.	If the functionality can't be developed uniquely there are many plugins that exist already that provide this functionality. If any of the higher risks require extended development time a plugin can be used to eliminate the risk.
Bootstrap Framework	2/Low	The Bootstrap framework provides a platform that the application will be built on. It uses HTML5 and CSS, which have been used by the developer before. There is nearly no risk element to using this framework.	With all new technologies it is still important to plan for the event of it not being useable. If the framework proves to be to complex to use a basic HTML5 and CSS application will be developed.

APPENDIX SECTION I: IMPLEMENTATION

DATABASE SET UP

```
--
-- Database: `localhype`
--
```

```
--
-- Table structure for table `friendship`
--

CREATE TABLE `friendship` (
`friendship_id` int(11) NOT NULL,
  `username` varchar(255) NOT NULL,
  `userto` varchar(255) NOT NULL,
  `status` int(10) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=40 DEFAULT CHARSET=latin1;
```

```
--
-- Table structure for table `members`
--

CREATE TABLE `members` (
`memberID` int(11) NOT NULL,
  `username` varchar(255) NOT NULL,
  `profilepic` varchar(255) NOT NULL,
  `userlocation` varchar(255) NOT NULL,
  `password` varchar(60) NOT NULL,
  `email` varchar(255) NOT NULL,
  `active` varchar(255) NOT NULL,
  `resetToken` varchar(255) DEFAULT NULL,
  `resetComplete` varchar(3) DEFAULT 'No'
) ENGINE=InnoDB AUTO_INCREMENT=15 DEFAULT CHARSET=latin1;
```

```
--
-- Table structure for table `Invitations`
--

CREATE TABLE `Invitations` (
`invite_id` int(11) NOT NULL,
  `eventname` varchar(225) NOT NULL,
  `eventdate` varchar(225) NOT NULL,
  `eventimg` varchar(500) NOT NULL,
  `username` varchar(225) NOT NULL,
  `userto` varchar(225) NOT NULL,
  `status` int(11) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=56 DEFAULT CHARSET=latin1;
```

```
--
-- Indexes for table `friendship`
--
ALTER TABLE `friendship`
 ADD PRIMARY KEY (`friendship_id`);

--
-- Indexes for table `Invitations`
--
ALTER TABLE `Invitations`
 ADD PRIMARY KEY (`invite_id`);

--
-- Indexes for table `members`
--
ALTER TABLE `members`
 ADD PRIMARY KEY (`memberID`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `friendship`
--
ALTER TABLE `friendship`
MODIFY `friendship_id` int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=40;
--
-- AUTO_INCREMENT for table `Invitations`
--
ALTER TABLE `Invitations`
MODIFY `invite_id` int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=56;
--
-- AUTO_INCREMENT for table `members`
--
ALTER TABLE `members`
MODIFY `memberID` int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=15;
```

SIGN UP VALIDATION

```
//if form has been submitted process it
if(isset($_POST['submit'])){


    //USERNAME VALIDATION
    if(strlen($_POST['username']) < 5){ //if username is less than 5 show error
        $error[] = 'X Username Must Be Over Five Characters.';
    } else {
        $stmt = $db->prepare('SELECT username FROM members WHERE username = :username'); //else query if username exists
        $stmt->execute(array(':username' => $_POST['username']));
        $row = $stmt->fetch(PDO::FETCH_ASSOC); //fectes any rows that match the query

        if(!empty($row['username'])){   //if any rows are returned the username is already in use
            $error[] = 'X Username provided is already in use.';
        }
    }


    //PASSWORD VALIDATION
    if(strlen($_POST['password']) < 7){ //if password is less than 7 display error
        $error[] = 'X Password Must be Over 7 Characters.';
     }

    if(strlen($_POST['passwordConfirm']) < 7){ //if confirm password is less than 7 display error
        $error[] = 'X Confirm password Must Be Over 7 Characters.';
     }

    if($_POST['password'] != $_POST['passwordConfirm']){ //password and confirm password must match
        $error[] = 'X Passwords do not match.';
     }


    //EMAIL VALIDATION
    if(!filter_var($_POST['email'], FILTER_VALIDATE_EMAIL)){ //email must be a valid email address
        $error[] = 'X Please enter a valid email address';
    } else {
        $stmt = $db->prepare('SELECT email FROM members WHERE email = :email'); //queries if entered email is in the database
        $stmt->execute(array(':email' => $_POST['email']));
        $row = $stmt->fetch(PDO::FETCH_ASSOC);

        if(!empty($row['email'])){ //checks if email is already in use
            $error[] = 'Email provided is already in use.'; //if it is error is displayed
        }
    }

//End Validation
```

SIGN UP SCRIPT

```
    if(!isset($error)){

        //hash the password
        $hashedpassword = $user->password_hash($_POST['password'], PASSWORD_BCRYPT);

        //create the activation code and places into a variable
        //md5 - hashes
        //uniqid- generates a unique identifier
        //rand - generates a random idenitifer
        $activasion = md5(uniqid(rand(),true));

        try {

            //insert into database with a prepared statement
            $stmt = $db->prepare('INSERT INTO members (username,profilepic,userlocation,password,email,active) VALUES
(:username, :profilepic, :userlocation, :password, :email, :active)');
            $stmt->execute(array(
                ':username' => $_POST['username'], //posted through username
                ':profilepic' => "default.jpg", //sets the profile pic string value to "default.jpg"
                ':userlocation' => "Earth", //sets the user location toa default of Earth
                ':password' => $hashedpassword, //passes the hashed passedword into the database for security
                ':email' => $_POST['email'], //passes the posted email
                ':active' => $activasion //passes the activation code generated
            ));
            $id = $db->lastInsertId('memberID'); //creates memeber ID from the last member added



            //send activation email to user
            $to = $_POST['email']; //sends to the email address entered at registration
            $subject = "Registration Confirmation";
            $body = "Thank you for Signing up to LOCAL HYPE .\n\n To activate your account, please click on this link:\n\n ".DIR."activate.php?x=
$id&y=$activasion\n\n Then Go Explore Your Local Events \n\n"; //places a link in that matches the activation code in database
            $additionalheaders = "From: <".SITEEMAIL.">\r\n"; //from defined site email in config file
            $additionalheaders .= "Reply-To: $".SITEEMAIL.""; //from defined site email in config file
            mail($to, $subject, $body, $additionalheaders);

            //redirect to index page
            header('Location: index.php?action=joined');
            exit;

        //else catch the exception and show the error.
        } catch(PDOException $e) {
            $error[] = $e->getMessage();
        }

    }
```

LOGGING IN

```php
//process login form if submitted
if(isset($_POST['submit'])){

    $username = $_POST['username'];
    $password = $_POST['password'];

    if($user->login($username,$password)){

        $_SESSION['username'] = $username;
        header('Location: memberpage.php');
        exit;

    } else {
        $error[] = ' X Wrong username or password, or your account has not been activated.';
    }

}//end if submit
```

PROFILE PAGE - PROFILE PICTURE

```php
//=======  UPDATE PROFILE PIC =======

$("#uploadpic").on('submit',(function(e){
    e.preventDefault();


           $.ajax({
               url: "includes/profile/upload.php", //passes the value to uploaded.php
               type: "POST",
               data:  new FormData(this), //gets the form data
               contentType: false,
               cache: false,
               processData:false,
               success: function(data){
                   $("#targetLayer").html(data);
                   location.reload();
                   },
               error: function(){}


           });

  }));

////////////////////////////////// End
```

UPDATE PROFILE DETAILS

```php
$username = $_POST["username"];
$email = $_POST["email"];
$userprofilelocation = $_POST["userprofilelocation"];

        try {
             $updateprofile = $db->query('UPDATE members
             SET username = "'.$username.'", email = "'.$email.'", userlocation = "'.$userprofilelocation.'"
             WHERE username = "'.$username.'"');

            }


         catch(Exception $e) {
             echo $e->getMessage();
             die();
         }


         $updateprofile->execute();
```

SEARCHING FOR A FRIEND

```
//=======  Friend Search ========


    $("#friendsearch").on("click", function(){ //when clicked

    usersearch = $('#friendsearchinput').val(); //gets in value of the input box that was submitted

        if(usersearch == username){

            $("#searchfriend_list").html(""); //clear the listing area
            $("#friendsearchinput").val(""); //empty the search box
            swal("Cant search for yourself!") //cant search for yourself message
        }

        else{

            $("#searchfriend_list").html(""); //empty the search friend list

            $.ajax({
                url:'includes/searchfriends.php',
                type:'POST',
                data: { usersearch : usersearch, username : username }, //pass data to search friends script
                dataType:'json',
                success: showSearch,
                error: errorSearch
                });

            }

         });


    function showSearch(responseData){
            $("#searchfriend_tmpl").tmpl(responseData).appendTo("#searchfriend_list");  //appends the data from templating
            $("#friendsearchinput").val(""); //reset search box to empty
        }

        function errorSearch(){
            sweetAlert("Oops...", "Already Friends.", "error"); //error if already friends
        }
```

ADDING A FRIEND

```
    $username = $_POST["username"];
    $useradd = $_POST["useradd"];

        try {
             $addresult = $db->query('INSERT INTO friendship (username, userto, status)
             VALUES ("'.$useradd.'", "'.$username.'", "2")');
         }


          catch(Exception $e) {
              echo $e->getMessage();
              die();
          }
```

EVENTS GENERATION

```
<?php

    include 'classes/get.invited.api.php'; // INCLUDES THE GET INVITED API

     use getinvited\api; // USE THE METHODS WITHIN THE APIS SCRIPT

     $Vdata = file_get_contents('includes/key.txt'); //calls in the api key stored in a seperate text file for security

     $query = new api($Vdata); //places the API key within a variavle query


              $lat = $_POST['formlat']; //places the geolocated latitude within variable lat
              $lng = $_POST['formlng']; //places the geolocated longtitude within variable lng


              //mydata variable includes an array of details needed to generate a list of events

              $mydata = array (
                'data' => array (
                'lat' => $lat, //the latitude
                'lng' => $lng, //the longtitude
                'distance' => 10 //distance in miles from the geolocation
                    )
                );



                $events = $query->get('nearby', $mydata); //places the events returned from the queried data into a variable events


                foreach ($events->results as $result) { //creates a loop to display they array of events

                echo '<div class="col-xs-6 col-lg-4 events">';
                 echo '<div class="flip-container" ontouchstart="this.classList.toggle("hover");">';
                    echo '<div class="flipper">';
                     echo '<div class="front">'; //front pannel for the flip cards
```

```
// IMAGE PLACEMENT
    if (strpos($result->event_logo,'http') !== false){  //checks if the returned result contains "http"
        echo '<img alt="" src=" '.$result->event_logo.'" />'; //outputs the event logo inside img tag
        }

    elseif ( $result->event_logo != '' ) { //if the event logo returned isnt empty place it within the specified file path
        echo '<img alt="" src="https://cdn.getinvited.to/eventlogos/'.$result->event_logo.'">';
        }

    else { //if the event logo returned is empty place the holder graphic
        echo '<img alt="" src="img/holder.png">';
     }


// DETAILS

    echo '<h1>'.$result->Title.'</h1>';//echos out the event title as heading
    echo '<p>'.$result->Venue;          //echoes out concatinated venue and town
    echo ' ' . $result->Town .'</p>';



// DATE CONVERSION
     $timestamp = $result->Start_Timestamp; //returnes the date from the saved time stamp
     echo '<p>' . date('jS F Y', $timestamp) . '</p><br />';

 echo '</div>';

 echo '<div class="back">';


 // DISTANCE
      $latevent = $result->lat;     //gets event latitude
      $lngevent = $result->lng;     //gets event longtitude
      $latlngevent = $latevent .',' . $lngevent;  //combines these together into a string


      $starttime = $result->Start_Timestamp; //gets the start time
      $endtime = $result->End_Timestamp; //gets the end time

      echo '<h5>Time</h5>';
      echo '<p>' . date('H:i', $starttime)  .'-' . date('H:i', $endtime) . '</p>'; //concatinates the varibles for time duration

      echo '<h5>Address</h5>';
      echo '<p>' .$result->Address .'</p>' ;
      echo '<p> ' . $result->Postcode .'</p>';
```

```
              echo '<button class="btn btn-primary btn-sm letsgo" data-toggle="modal" data-target="#letsgomodel">Lets go</button>';
              echo '<input type="hidden" class ="invitetitle" value="'.$result->Title.'" />';
              echo '<input type="hidden" class ="invitedate" value="'.date('jS F Y', $timestamp).'" />';
              echo '<input type="hidden" class ="inviteimg" value="'.$result->event_logo.'" />';

              echo '<button class="btn btn-primary btn-sm directions" data-toggle="modal" data-target="#myModal">Directions</button>';
              echo '<input type="hidden" value="'.$latlngevent.'" />';


        echo '</div>';


       echo '</div>';

    echo '</div>';
  echo '</div>';

     } //end for each loop
 ?>
```

INVITE FRIEND TO EVENT

```
       try {

           $stmt = $db->query('SELECT eventname, eventdate, username, userto, status from Invitations
           WHERE eventname = "'.$eventname.'" AND eventdate = "'.$eventdate.'"  AND username = "'.$userinvite.'" AND userto = "'.$username.'" AND
status = 2 ');

        }


        catch(Exception $e) {
            echo $e->getMessage();
            die();
        }


        $rows = $stmt->fetchAll(PDO::FETCH_ASSOC);


           if (count($rows)== 0) {

             $addresult = $db->query('INSERT INTO Invitations (eventname, eventdate, eventimg,  username, userto, status) VALUES ("'.$eventname.'",
"'.$eventdate.'", "'.$eventimg.'", "'.$userinvite.'", "'.$username.'", "2")');


            echo "<p>Invitation sent to ".$userinvite." </p>";

           }



           if (count($rows) == 1){

           echo "<p>You have already invited ".$userinvite." to this event!</p>";

           }
?>
```

DIRECTIONS ENHANCEMENT

```
var directionsDisplay;
var directionsService = new google.maps.DirectionsService(); //sets up a directions service variable

function initialize() { //initialise function
  directionsDisplay = new google.maps.DirectionsRenderer();
  var mapOptions = {
    zoom: 17, //sets initial zoom level
    center: new google.maps.LatLng(54.787714900000000000, -6.492314500000020000), //centeres the map
    scrollwheel: false, //sets the scroll with track pad to false
    style: [
```

```
  var map = new google.maps.Map(document.getElementById('map-canvas'), //map variable is filled with a new google map inside map-canvas
      mapOptions);
  directionsDisplay.setMap(map);
  directionsDisplay.setPanel(document.getElementById('directions-panel')); //places directions into directions panel

  var control = document.getElementById('control');
  map.controls[google.maps.ControlPosition.TOP_CENTER].push(control);

 ////////////////////////////////////////////////////////////////////////////
 ////////////////////////// FIX FOR BOOTSTRAP MODAL ////////////////////////////
    $("#myModal").on("shown.bs.modal", function(e) {
      google.maps.event.trigger(map, "resize");
      return map.setCenter(new google.maps.LatLng(userlat, userlong));
    });


} //end initialize function


var directions; //difines the variable of directions



$(document).on("click",".directions",function() { //when the directions button is clicked
    $('#directions-panel').html(""); //the pannel that contains the directsions is cleared, this insures a fresh set of directions displays
    directions =  $(this).next().val(); //gets the value of the directions from the events listing
    calcRoute(); //runs the calc route function when the directions button is clicked
});
```

```
function calcRoute(start, end) {
  var start = usergeo; //sets the start point as the users location
  var end = directions; //sets the end point as the event location
  var selectedMode = document.getElementById('mode').value; //gets value of the selected mode
  var request = {
    origin: start,
    destination: end,
    travelMode: google.maps.TravelMode[selectedMode] //places the selected mode at the end of the travel mode query
  };
  directionsService.route(request, function(response, status) {
    if (status == google.maps.DirectionsStatus.OK) {
      directionsDisplay.setDirections(response); //responds with directions
    }
  });

}



google.maps.event.addDomListener(window, 'load', initialize); //when the docuement is loaded the initalize function will run
```

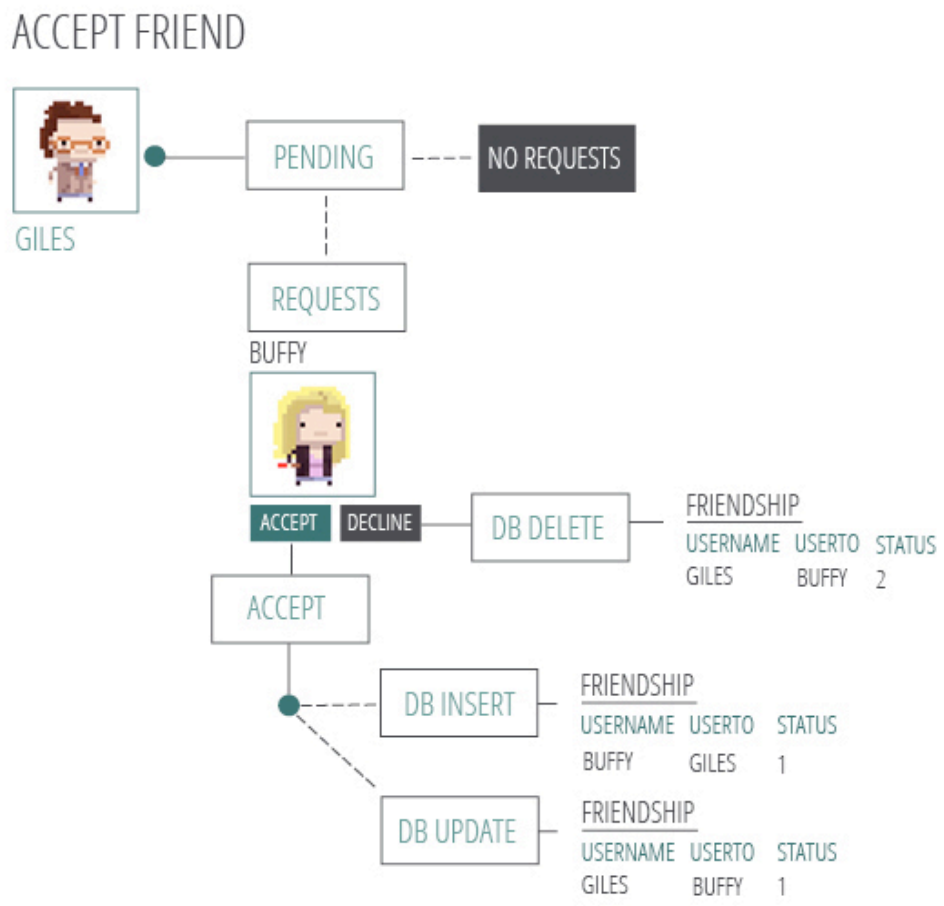
EVENT INVITATION

```
try {
    $stmt = $db->query('SELECT * from Invitations
    WHERE status = 2 AND username= "'.$username.'"');
}


catch(Exception $e) {
    echo $e->getMessage();
    die();
}

$invites = $stmt->fetchAll(PDO::FETCH_ASSOC);
$invites = json_encode($invites);

echo $invites;
```

EVENT ATTENDING

```
try {
    $stmt = $db->query('SELECT * from Invitations
    WHERE status = 1 AND username= "'.$username.'"');
}


catch(Exception $e) {
    echo $e->getMessage();
    die();
}



$data = $stmt->fetchAll();
foreach ($data as $row)
{
    echo'<div class="attending holder">';
     echo "<h2>" . $row['eventname'] . "</h2>" ;


        if (strpos($row['eventimg'],'http') !== false){ //checks if the returned result contains "http"
        echo '<img width="200" alt="" src=" '. $row['eventimg'] .'" />'; //outputs the event logo inside img tag
    }

    elseif ($row['eventimg'] != '' ) { //if the event logo returned isnt empty place it within the specified file path
        echo '<img width="200"alt="" src="https://cdn.getinvited.to/eventlogos/'. $row['eventimg'] .'">';
     }

    else { //if the event logo returned is empty place the holder graphic
            echo '<img width="200" alt="" src="img/holder.png">';
        }


     echo "<br><p>". $row['eventdate'] . "</p>";
     echo "<p>Going with: ". $row['userto'] . "</p>";
    echo'</div>';
}
```

ACCEPT/DECLINE FRIEND



APPENDIX SECTION J: UNIT TESTS

Unit Test ID #: 001	
Component:	User Register
Purpose:	Ensuring the Sign up process
Inputs:	User details in registration from. Email & username.
Expected Result:	Users attempted registration. Activation should be sent to email account. User details should be inserted into the members table.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 002	
Component:	User Register with password greater than 7
Purpose:	Sign up process validation check
Inputs:	User details in registration from. Email & username.
Expected Result:	Users attempted registration returns error
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 003	
Component:	User Register with a taken username
Purpose:	Sign up process validation check
Inputs:	User details in registration from. Email & username.
Expected Result:	Users attempted registration returns error
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 004	
Component:	User Register with an invalid email
Purpose:	Sign up process validation check
Inputs:	User details in registration from. Email & username.
Expected Result:	Users attempted registration returns error
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 005	
Component:	Activation Email
Purpose:	Check activation email link successfully actives account
Inputs:	n.a
Expected Result:	The activation email sets the user account to activate
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 006	
Component:	Customer Login
Purpose:	Insuring a registered user can sign in
Inputs:	Email & username
Expected Result:	Signs the user into the system and directs to the Explore page.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 007	
Component:	Customer Login with wrong username/password
Purpose:	Sign up process validation check.
Inputs:	Email & username.
Expected Result:	Users attempted registration returns error
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 008	
Component:	Forgot Password
Purpose:	Check user can reset password
Inputs:	Email
Expected Result:	Reset email sent to users email with reset link
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 009	
Component:	Generate Events
Purpose:	Insure the Get Invited API is operating with the system
Inputs:	Geolocation & API file.
Expected Result:	Page will list out any event results in a 10-mile radius.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 010	
Component:	Display User City Name
Purpose:	Insure the city named displayed is accurate and operating properly.
Inputs:	Geolocation & Simple weather plugin script.
Expected Result:	The city name of the current location of app access will be displayed.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 011	
Component:	Display weather of current location
Purpose:	Insure the weather results returned are accurate and operating properly.
Inputs:	Geolocation & Simple weather plugin script.
Expected Result:	The temperature and weather symbol will be loaded for users location.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 012	
Component:	Profile Page user details displayed
Purpose:	Insure the user details are being displayed from the database accurately.
Inputs:	Username of signed in user
Expected Result:	The user details of signed in user will be displayed into input boxes for update.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 013	
Component:	Update user details
Purpose:	Insure user details get updated when the update button is clicked.
Inputs:	Username, email, location.
Expected Result:	Ajax clears and places the new details into input. Database update complete.
Actual Result:	Database function runs, Ajax does not place in new details
Pass/Fail:	Fail

Unit Test ID #: 014	
Component:	Upload new profile picture
Purpose:	Insure user can upload and update old profile pictures
Inputs:	Selected image file from computer.
Expected Result:	The new image renamed and placed into directory. Database profile pic updated.
Actual Result:	As expected
Pass/Fail:	Pass

Unit Test ID #: 015	
Component:	Search for a user
Purpose:	Users have the ability to search for friends.
Inputs:	Inputted search string from user
Expected Result:	Returns profile picture, location and user name of user. Button to add
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 016	
Component:	Search for yourself
Purpose:	Search for friend process validation check.
Inputs:	Inputted search string from user
Expected Result:	Sweet alert message “You cant search for yourself”
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 017	
Component:	Search for a used your already friends with
Purpose:	Search for friend process validation check.
Inputs:	Inputted search string from user
Expected Result:	Sweet alert error message saying “Your already friends with “search string”?.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 018	
Component:	Send Friendship invitation to user
Purpose:	Test that database inserts with pending friendship details.
Inputs:	Username & userto from “add” button value.
Expected Result:	Database inserts a row with status of 2 meaning pending.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 019	
Component:	Accept Friend Invitation
Purpose:	Test database updates and inserts the correct friendship information.
Inputs:	Username, userto
Expected Result:	Database updates friendship details status to 1. Inserts reverse friendship.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 020	
Component:	Decline Friend Invitation
Purpose:	Test database deletes correct friendship information.
Inputs:	Username, userto
Expected Result:	Database deletes friendship status where status is 2 with username and userto.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 021	
Component:	Update Friend Count
Purpose:	Check the friend count updates when a new friend Is accepted
Inputs:	n.a
Expected Result:	The friend count increases by 1 when the user accepts a friend request
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 022	
Component:	Update Event Count
Purpose:	Check the event count updates when user accepts friend invitation
Inputs:	n.a
Expected Result:	The event count increases by 1 when the user accepts a friend request
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 023	
Component:	Display Users Friends
Purpose:	Ensure the friends of the logged in user displays
Inputs:	username
Expected Result:	The profile picture, location and username of friends will appear in friends section
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 024	
Component:	Display Users Pending Friend Invitations
Purpose:	Ensure the pending friends of the logged in user displays
Inputs:	username
Expected Result:	The profile picture, location and username of friends will appear in pending section
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 025	
Component:	Get Directions to events
Purpose:	Check the Map and direction details are accurate from point A to B.
Inputs:	Geolocation, event location latitude & longitude
Expected Result:	The correct directions details appear when directions button is clicked.
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 026	
Component:	List invitations to events
Purpose:	List the invitations to events of the logged in user
Inputs:	Username
Expected Result:	Invitations and accept/decline will appear in the invitations tab on discovered page
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 027	
Component:	List Events users attending
Purpose:	List the events the logged in user is attending
Inputs:	Username
Expected Result:	List of events displayed to the user in the attending tab on discovered page
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 028	
Component:	Accept Invitation to Event
Purpose:	Insure accepted invitation correctly update and insert database information
Inputs:	Username ,userto, eventname, eventimg, eventdate
Expected Result:	Database row status will update to 1, new row inserted for userto acceptance
Actual Result:	As expected
Pass/Fail:	Pass

Unit Test ID #: 029	
Component:	Decline Invitation to event
Purpose:	Insure decline invitation correctly deletes invitation from database
Inputs:	Username ,userto, eventname, eventimg eventdate
Expected Result:	Query will select the row with details and status of 2 and remove it from database
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 030	
Component:	Send friend an invitation to event
Purpose:	Insure sending an invitation places details into the database successfully
Inputs:	Eventname, eventdate, eventpic, username, userto
Expected Result:	Inserts above inputs and sets status to 2 for pending into database
Actual Result:	As Expected
Pass/Fail:	Pass

Unit Test ID #: 031	
Component:	Send friend second invitation to same event
Purpose:	Insure you cant invite a friend twice to the same event
Inputs:	Eventname, eventdate, eventpic, username, userto
Expected Result:	Error messages displays informing user the friend has already been invited.
Actual Result:	As Expected
Pass/Fail:	Pass

APPENDIX SECTION K: TEST CASES

Test Case ID #: 001	
Related Requirement:	Requirement #1.1
Description:	Test to see if user can create account with unique username
Test Path:	1 - Navigate to Localhype.co 2 - Sign up to the application with a unique username 3 – Verification message informing user to check email
Expected Result:	The application will send an activation email to users email specified. Database members table will be updated with values.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 002	
Related Requirement:	Requirement #1.2
Description:	Test to see if user can create account with valid password
Test Path:	1 - Navigate to Localhype.co 2 - Sign up to the application with a password no longer than bla 3 – Enter the same password again for confirmation 4 – Verification message informing user to check email
Expected Result:	The application will send an activation email to users email specified. Database members table will be updated with values.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 003	
Related Requirement:	Requirement #1.3
Description:	Test to see if user can create account with valid email
Test Path:	1 - Navigate to Localhype.co 2 - Sign up to the application with accurate email address 3 – Verification message informing user to check email
Expected Result:	The application will send an activation email to users email specified. Database members table will be updated with values.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 004	
Related Requirement:	#1.4
Description:	Verification that on account creation default image and location name is inserted into members table
Test Path:	1 - Navigate to Localhype.co 2 - Sign up to the application, see Test Case IDs – #001, #002, #003 3 – Verification message informing user to check email
Expected Result:	Members table will have updated with profilepic set as default.jpg and userlocation set to Earth.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 005	
Related Requirement:	Requirement #2.1
Description:	Verification that user can successfully login to application with correct details
Test Path:	1 - Navigate to Localhype.co/login.php 2 – Enter username and password from sign up
Expected Result:	Application should redirect upon successful login to /memberpage.php
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 006	
Related Requirement:	Requirement #3.1
Description:	Logged in user can add/change profile picture
Test Path:	1 – Follow test case #005 2 – Navigate to profile using the side bar 3 – Click the settings tab 4 – Click choose file under “upload profile pic” 5 – Once selected click upload image
Expected Result:	The user profile picture will change. Database updated with new image name. Profile picture uploaded to directory with username as the filename.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 007	
Related Requirement:	Requirement #3.2
Description:	Logged in user can edit personal information
Test Path:	1 – Follow test case #005 2 – Navigate to profile using the side bar 3 – Click the settings tab 4 – Update any user information 5 – Click the update button
Expected Result:	When the update button is clicked the user details will dynamically change on the screen to the updated results. The database will be updated with the new information.
Actual Result:	The database information updated but the information didn't repopulate
Pass/Fail:	Fail
Correction Action:	Values in input boxes needed cleared before inserting new values. Passed on retest.

Test Case ID #: 008	
Related Requirement:	Requirement #3.3
Description:	Logged in user can view friends list
Test Path:	1 – Follow test case #005 2 – Navigate to profile using the side bar 3 – The open tab is Friends 4 – Locate the second panel labeled friends
Expected Result:	The panel should be filled with usernames, profile pictures and locations of the logged in users friends if any exist. As well as a button to delete that friend.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 009	
Related Requirement:	Requirement #3.4
Description:	Logged in user can search and add new friends
Test Path:	1 – Follow test case #005 2 – Navigate to profile using the side bar 3 – The open tab is Friends 4 – Locate the search for friend panel 5 – type the name of the user you wish to search for and click "find"

Expected Result:	If the user exists the profile picture, username and location of the searched for user will be displayed, along with an “add” button if you wish to add that user.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 010	
Related Requirement:	Requirement #4.1
Description:	Generate listing of logged in users friends
Test Path:	1 – Follow test case #005 2 – Navigate to profile using the side bar 3 – The open tab is Friends 4 – Locate the panel labeled pending
Expected Result:	The panel should be filled with usernames, profile pictures and locations of the logged in users pending friends if any exist. As well accept and delete buttons for that friend request.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 011	
Related Requirement:	Requirement #4.2
Description:	Determine users Geolocation
Test Path:	1 – Follow test case #005
Expected Result:	The events listing will generate as will the weather and city name. All these functions use the users geolocation.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 012	
Related Requirement:	Requirement #4.3
Description:	Logged in user can invite their friend to an event
Test Path:	1 – Follow test case #005 2 – Hover on an event and select lets go 3 – Choose from the list of friends, click “invite”
Expected Result:	A message will appear showing the friend request has been sent. The database will place the username, userto, eventname, eventdate and eventpic

	into the Invitations table with a status of 2.
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 013	
Related Requirement:	Requirement #4.4
Description:	Get directions to events
Test Path:	1 – Follow test case #005 2 – Hover on an event and select directions
Expected Result:	A modal box will appear with a generated map and directions list
Actual Result:	The modal box contained the directions list but there was style errors with the map
Pass/Fail:	Fail
Correction Action:	Netgloo blog explains that if Google maps are accessed within a modal box you need to preform additional functionality. A resize needs to be triggered when the modal window is opened (shown.bs.modal) so the map resizes to the loaded map container.

Test Case ID #: 014	
Related Requirement:	Requirement #4.5
Description:	Display users current city name
Test Path:	1 – Follow test case #005 2 – As test case #011 has passed
Expected Result:	The logged in users location will display on the explore page
Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 015	
Related Requirement:	Requirement #5.1
Description:	My events section - Attending
Test Path:	1 – Follow test case #005 2 – Navigated to the discovered page using the sidebar 3 – The tab open will be attending
Expected Result:	A list of events the user has confirmed to be attending will be generated. Event name, event image, event date and name of user your attending with will be displayed.

Actual Result:	As expected
Pass/Fail:	Pass

Test Case ID #: 016	
Related Requirement:	Requirement #5.2
Description:	My events section - Invitations
Test Path:	1 – Follow test case #005 2 – Navigated to the discovered page using the sidebar 3 – Navigate to the Invites tab
Expected Result:	A list of events the user is attending will be generated. Event name, event image, event date and name of user your attending with will be displayed.
Actual Result:	A list of events the user is attending will be generated. Event name, event date and name of user you're attending with will be displayed. Buttons to accept or decline that event will also be generated.
Pass/Fail:	Pass

APPENDIX SECTION L: UX TESTING

STUDENT USER

USER EVALUATION

HIGHLIGHT THE MOST APPROPRIATE NUMBER THAT YOU FEEL REFLECTS THE USER INTERFACE OF LOCAL HYPE

1 - POOR 2- AVERAGE 3- GOOD 4- VERY GOOD 5- EXCELLENT

	POOR		GOOD		EXCELLENT
INTERFACE DESIGN	1	2	3	4	(5)
EASE OF USE	1	2	3	(4)	5
EASE OF NAVIGATION	1	2	3	4	(5)
CONSISTENCY	1	2	3	(4)	5
FUNCTIONALITY	1	2	3	4	(5)
USEFUL RESPONSE	1	2	3	(4)	5
EFFICIENCY	1	2	3	(4)	5
OVER ALL SATISFACTON	1	2	3	(4)	5

WOULD YOU USE LOCAL HYPE?

OF COURSE [X] ITS NOT FOR ME [ ]

OTHER COMMENTS OR SUGGESTIONS, IF YOU HAVE ANY :

The app is great, though the mobile version would work better as a mobile app. I never use browsers on my phone to use an application.

SIGNATURE : Joanne Cuffey

USER EVALUATION

HIGHLIGHT THE MOST APPROPRIATE NUMBER THAT YOU FEEL REFLECTS THE USER INTERFACE OF LOCAL HYPE

1 - POOR 2- AVERAGE 3- GOOD 4- VERY GOOD 5- EXCELLENT

	POOR		GOOD		EXCELLENT
INTERFACE DESIGN	1	2	3	(4)	5
EASE OF USE	1	2	3	(4)	5
EASE OF NAVIGATION	1	2	3	(4)	5
CONSISTENCY	1	2	3	(4)	5
FUNCTIONALITY	1	2	3	4	(5)
USEFUL RESPONSE	1	2	3	4	(5)
EFFICIENCY	1	2	3	(4)	5
OVER ALL SATISFACTON	1	2	3	(4)	5

WOULD YOU USE LOCAL HYPE?

OF COURSE [X] ITS NOT FOR ME [ ]

OTHER COMMENTS OR SUGGESTIONS, IF YOU HAVE ANY :

It was extremely minimal and easy to use, this was good as most event applications are very clutted which puts me off them.

SIGNATURE : Matt Clarke

USER EVALUATION

HIGHLIGHT THE MOST APPROPRIATE NUMBER THAT YOU FEEL REFLECTS THE USER INTERFACE OF LOCAL HYPE

1 - POOR 2- AVERAGE 3- GOOD 4- VERY GOOD 5- EXCELLENT

	POOR		GOOD		EXCELLENT
INTERFACE DESIGN	1	2	3	4	(5)
EASE OF USE	1	2	3	(4)	5
EASE OF NAVIGATION	1	2	3	(4)	5
CONSISTENCY	1	2	3	(4)	5
FUNCTIONALITY	1	2	3	4	(5)
USEFUL RESPONSE	1	2	3	(4)	5
EFFICIENCY	1	2	(3)	4	5
OVER ALL SATISFACTON	1	2	3	(4)	5

WOULD YOU USE LOCAL HYPE?

OF COURSE [X] ITS NOT FOR ME [ ]

OTHER COMMENTS OR SUGGESTIONS, IF YOU HAVE ANY :

I think the application would benefit from having a social media presence. I would use it but I would like to have my friends know about it so that it could be used to full effect.

SIGNATURE : Clara Thorniley

USER EVALUATION

HIGHLIGHT THE MOST APPROPRIATE NUMBER THAT YOU FEEL REFLECTS THE USER INTERFACE OF LOCAL HYPE

1 - POOR 2- AVERAGE 3- GOOD 4- VERY GOOD 5- EXCELLENT

	POOR		GOOD		EXCELLENT
INTERFACE DESIGN	1	2	3	(4)	5
EASE OF USE	1	2	3	4	(5)
EASE OF NAVIGATION	1	2	3	4	(5)
CONSISTENCY	1	2	3	(4)	5
FUNCTIONALITY	1	2	3	4	(5)
USEFUL RESPONSE	1	2	3	4	(5)
EFFICIENCY	1	2	3	(4)	5
OVER ALL SATISFACTON	1	2	3	4	(5)

WOULD YOU USE LOCAL HYPE?

OF COURSE [X] ITS NOT FOR ME [ ]

OTHER COMMENTS OR SUGGESTIONS, IF YOU HAVE ANY :

I think the idea is great and I would definately sign up.
Its key that it works on mobile so I can use it on the go.

SIGNATURE : Peter Scott

USER EVALUATION

HIGHLIGHT THE MOST APPROPRIATE NUMBER THAT YOU FEEL REFLECTS THE USER INTERFACE OF LOCAL HYPE

1 - POOR 2- AVERAGE 3- GOOD 4- VERY GOOD 5- EXCELLENT

	POOR		GOOD		EXCELLENT
INTERFACE DESIGN	1	2	(3)	4	5
EASE OF USE	1	2	(3)	4	5
EASE OF NAVIGATION	1	2	(3)	4	5
CONSISTENCY	1	2	3	(4)	5
FUNCTIONALITY	1	2	3	(4)	5
USEFUL RESPONSE	1	(2)	3	4	5
EFFICIENCY	1	2	3	(4)	5
OVER ALL SATISFACTON	1	2	(3)	4	5

WOULD YOU USE LOCAL HYPE?

OF COURSE [ ] ITS NOT FOR ME [X]

OTHER COMMENTS OR SUGGESTIONS, IF YOU HAVE ANY :

I found the application confusing as I dont use any apps.
It looked great and I liked the idea but I feel I would be more comfortable sticking to offline planning.

SIGNATURE : Edith Minford

USER EVALUATION

HIGHLIGHT THE MOST APPROPRIATE NUMBER THAT YOU FEEL REFLECTS THE USER INTERFACE OF LOCAL HYPE

1 - POOR 2- AVERAGE 3- GOOD 4- VERY GOOD 5- EXCELLENT

	POOR		GOOD		EXCELLENT
INTERFACE DESIGN	1	2	(3)	4	5
EASE OF USE	1	2	3	(4)	5
EASE OF NAVIGATION	1	2	3	(4)	5
CONSISTENCY	1	2	3	(4)	5
FUNCTIONALITY	1	2	3	(4)	5
USEFUL RESPONSE	1	2	(3)	4	5
EFFICIENCY	1	2	3	(4)	5
OVER ALL SATISFACTON	1	2	3	(4)	5

WOULD YOU USE LOCAL HYPE?

OF COURSE [X] ITS NOT FOR ME [ ]

OTHER COMMENTS OR SUGGESTIONS, IF YOU HAVE ANY :

It would be a very useful app though it would be good to have some sort of user guide. I dont use applications very much im not even on facebook I would only use it if there was somewhere I could check for guidance.

SIGNATURE : John Campbell

APPENDIX SECTION M: MANAGEMENT - GANTT CHART

Gantt chart deadline view

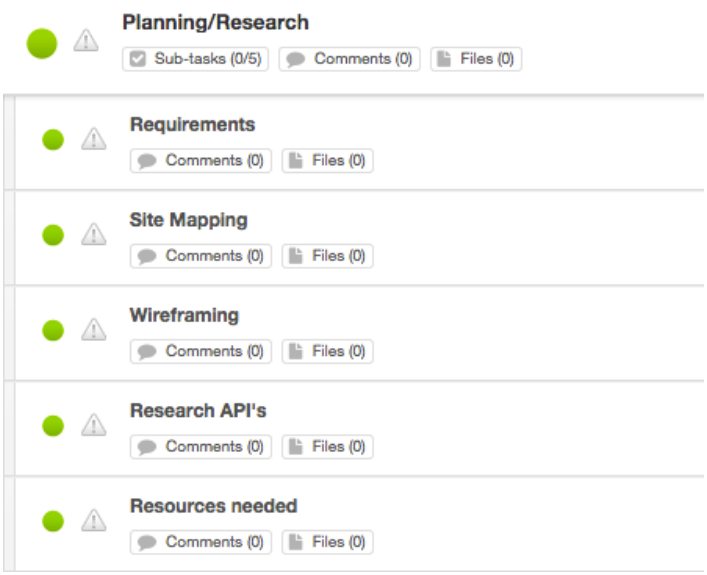
IMD Major Project

▾ **Research/Planning**	0%		Start	Due
Requirements	0%		Oct 15, 2014	Oct 16, 2014
Sitemapping	0%		Oct 17, 2014	Oct 19, 2014
Wireframing	0%		Oct 20, 2014	Oct 20, 2014
Research API's	0%		Oct 21, 2014	Oct 23, 2014
Resources needed	0%		Oct 24, 2014	Oct 25, 2014
Refactor	0%		Oct 26, 2014	Oct 28, 2014
Task | Milestone | Group of Tasks				
▾ **Visual & UX design**	0%		Start	Due
Branding	0%		Nov 1, 2014	Nov 10, 2014
User Interface Design	0%		Nov 6, 2014	Nov 16, 2014
Visual Design	0%		Nov 15, 2014	Nov 25, 2014
Frontend prototype	0%		Nov 25, 2014	Dec 5, 2014
Refactor	0%		Dec 6, 2014	Dec 16, 2014
Task | Milestone | Group of Tasks				

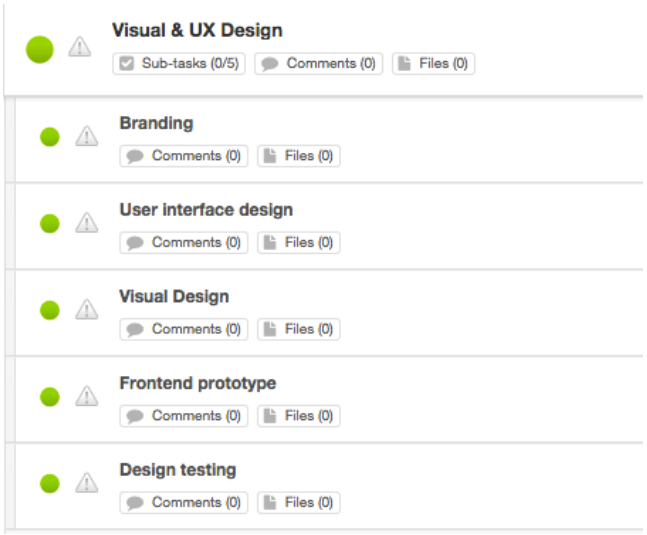
▾ **Development**	0%		Start	Due
Database design	0%		Dec 29, 2014	Jan 5, 2015
Database prototype	0%		Jan 6, 2015	Jan 11, 2015
Event API Implimentation	0%		Jan 12, 2015	Jan 23, 2015
Google map API impimentation	0%		Jan 24, 2015	Jan 31, 2015
User login system	0%		Feb 2, 2015	Feb 18, 2015
Connectivity to others	0%		Feb 19, 2015	Feb 25, 2015
Sending event requests	0%		Feb 26, 2015	Mar 1, 2015
Receiving & accepting requests	0%		Mar 3, 2015	Mar 8, 2015
Unit testing	0%		Jan 17, 2015	Mar 13, 2015
Refactor	0%		Mar 16, 2015	Mar 27, 2015
Task | Milestone | Group of Tasks				
▾ **Testing**	0%		Start	Due
UX testing	0%		Mar 30, 2015	Apr 8, 2015
User Surveys	0%		Apr 9, 2015	Apr 10, 2015
System testing	0%		Apr 13, 2015	Apr 21, 2015
Bug fixes	0%		Apr 22, 2015	Apr 24, 2015
Refactor	0%		Yesterday	Tuesday

APPENDIX SECTION N: MANAGEMENT - PROJECT BUBBLE

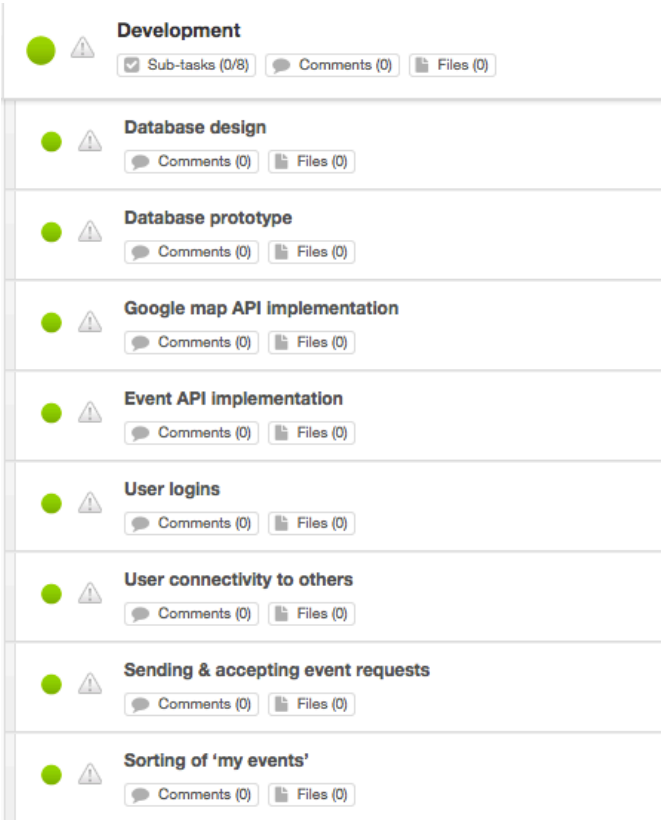
PROJECT BUBBLE INTERFACE



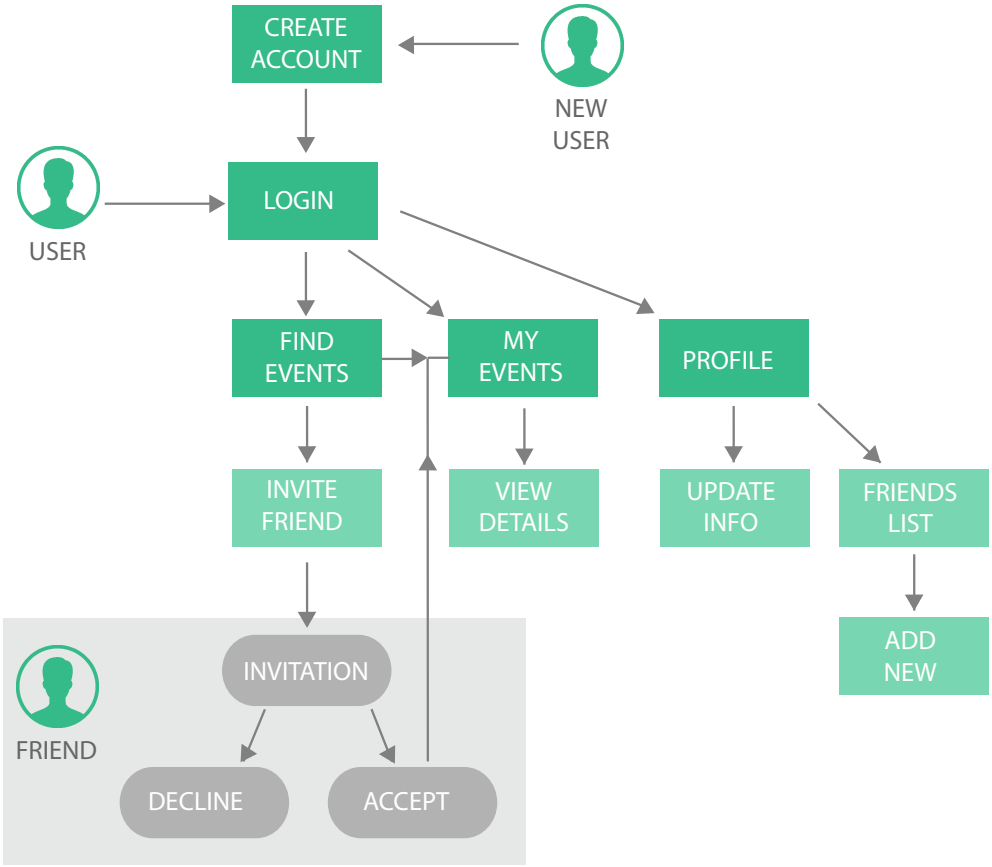
VISUAL STAGE WITH SUBTASKS



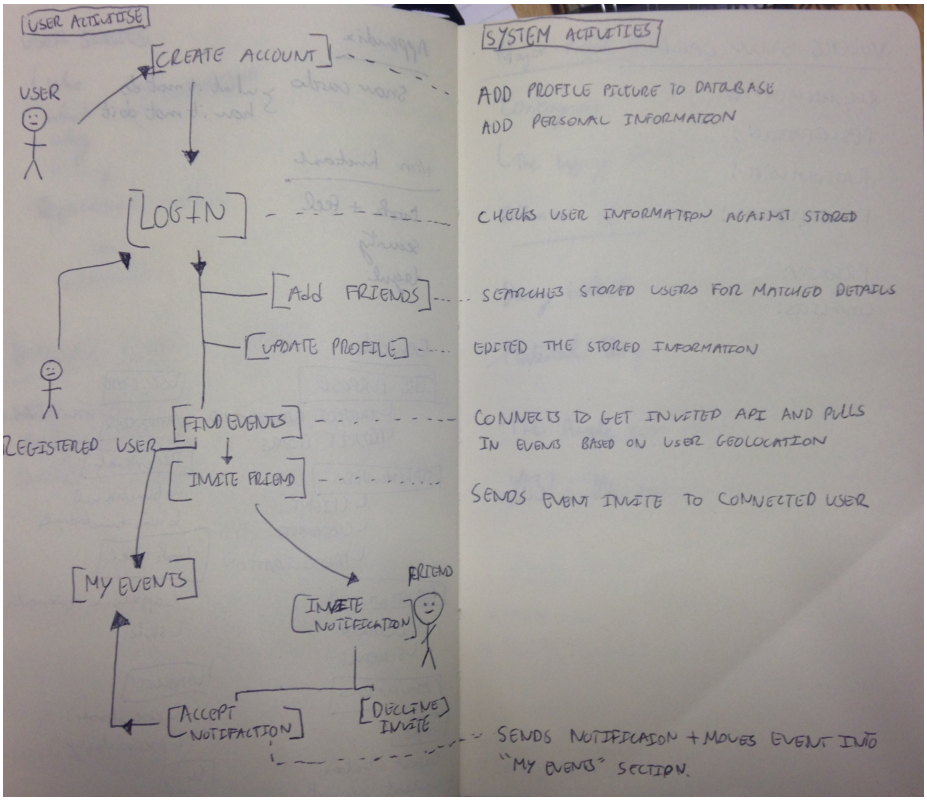
DEVELOPMENT STAGE WITH SUBTASKS



APPENDIX SECTION 0: USER FLOW

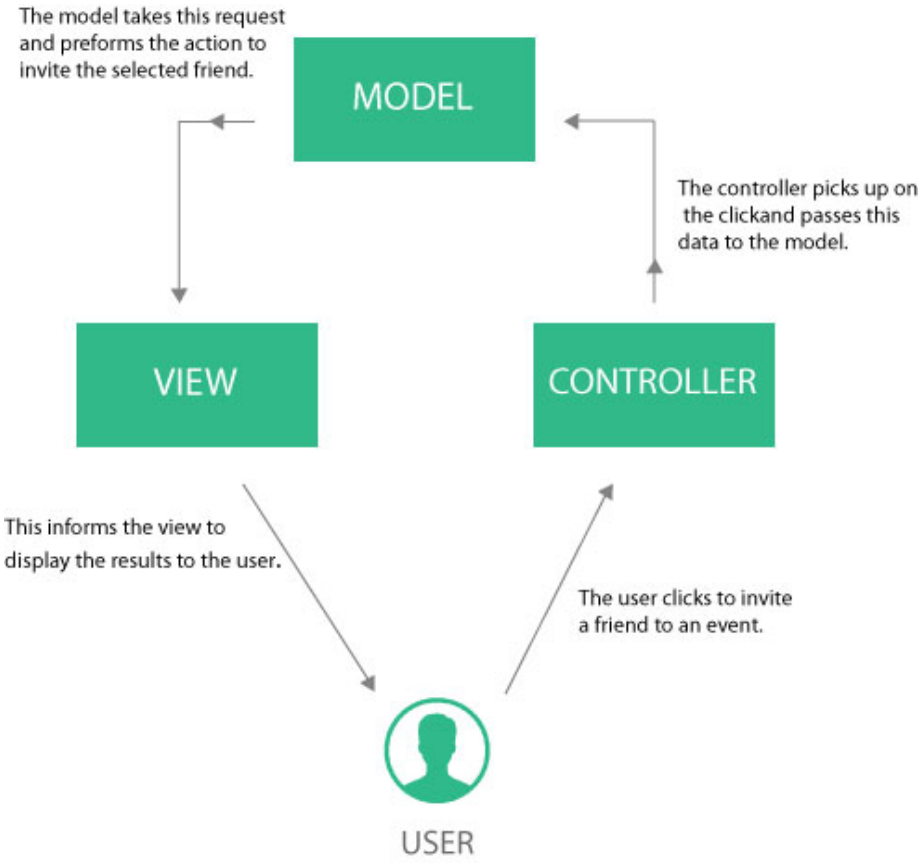


INITIAL SKETCH

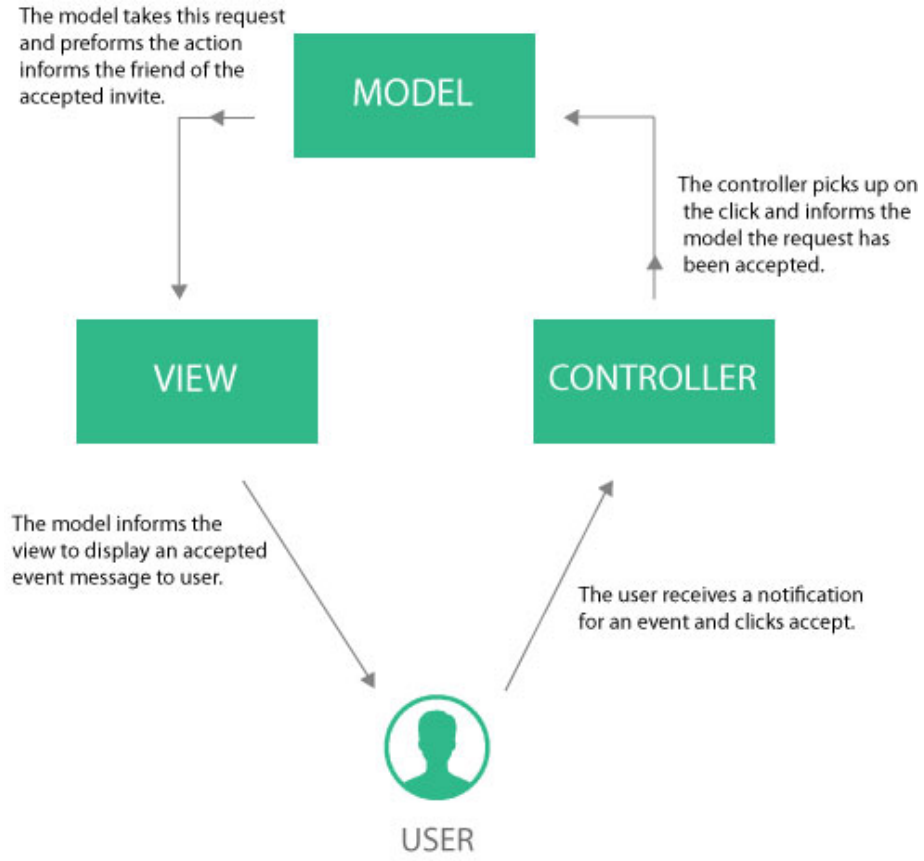


APPENDIX SECTION P: MODEL VIEW CONTROLLER

Inviting a friend to an event



Accepting an event invitation



LOCAL HYPE
USER GUIDE

BRAND

LOCAL HYPE

WORDMARK

LOCAL HYPE LOCAL HYPE

LOCAL HYPE LOCAL HYPE LOCAL HYPE

LOGO

CROWD GRAPHICS

Graphics that include more than one individual when being used for advertising purposes should used the wordmark. Any of the approved graphics will be permitted, providing they use a 80% opacity.

INDIVIDUAL GRAPHICS

Graphics featuring one invididual should be branding using the logo in the bottom right corner. Any of the above approved logos are permitted.

TYPOGRAPHY

HEADING TYPE

Aa

ABCDEFGHIJKLMNOPQR
STUVWXYZ 1234567890

OPEN SANS
CONDENCED LIGHT

BODY TYPE

Aa

ABCDEFGHIJKLMNOPQR
STUVWXYZ 1234567890

OPEN SANS
REGULAR

COLOURS

#FFFFFF #6BA19D #3F7778 #4B4E53 #333333

SIGN UP

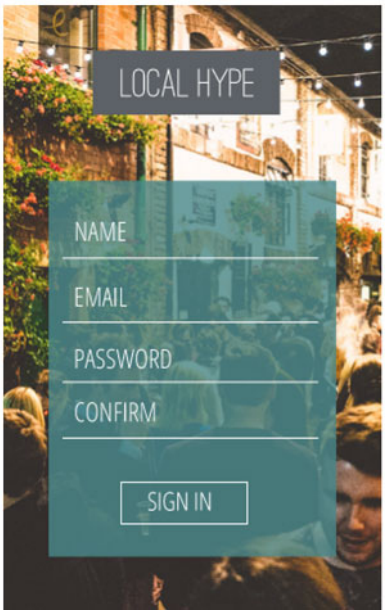
1. Clicking on the sign up button will submit the information entered, it will not submit untill it has been entered correctly.

2. Users will have validation checks while entering the data. The user can clearly see this way why the information can't be submitted.

MOBILE

Much like the sign in page a randomly generated graphic will fill the background.

If all the information is correct the user will be logged into the system to start setting up their profile.



SIGN IN

1 Clicking on the sign up button will allow users to register for the application on a different page.

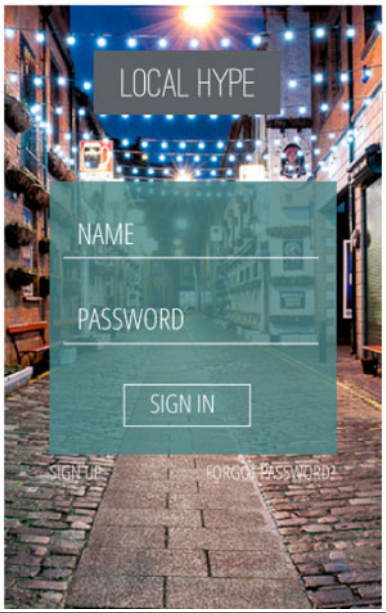
2 If a user forgets their password they can click on "forgot password" which will display a modal box.

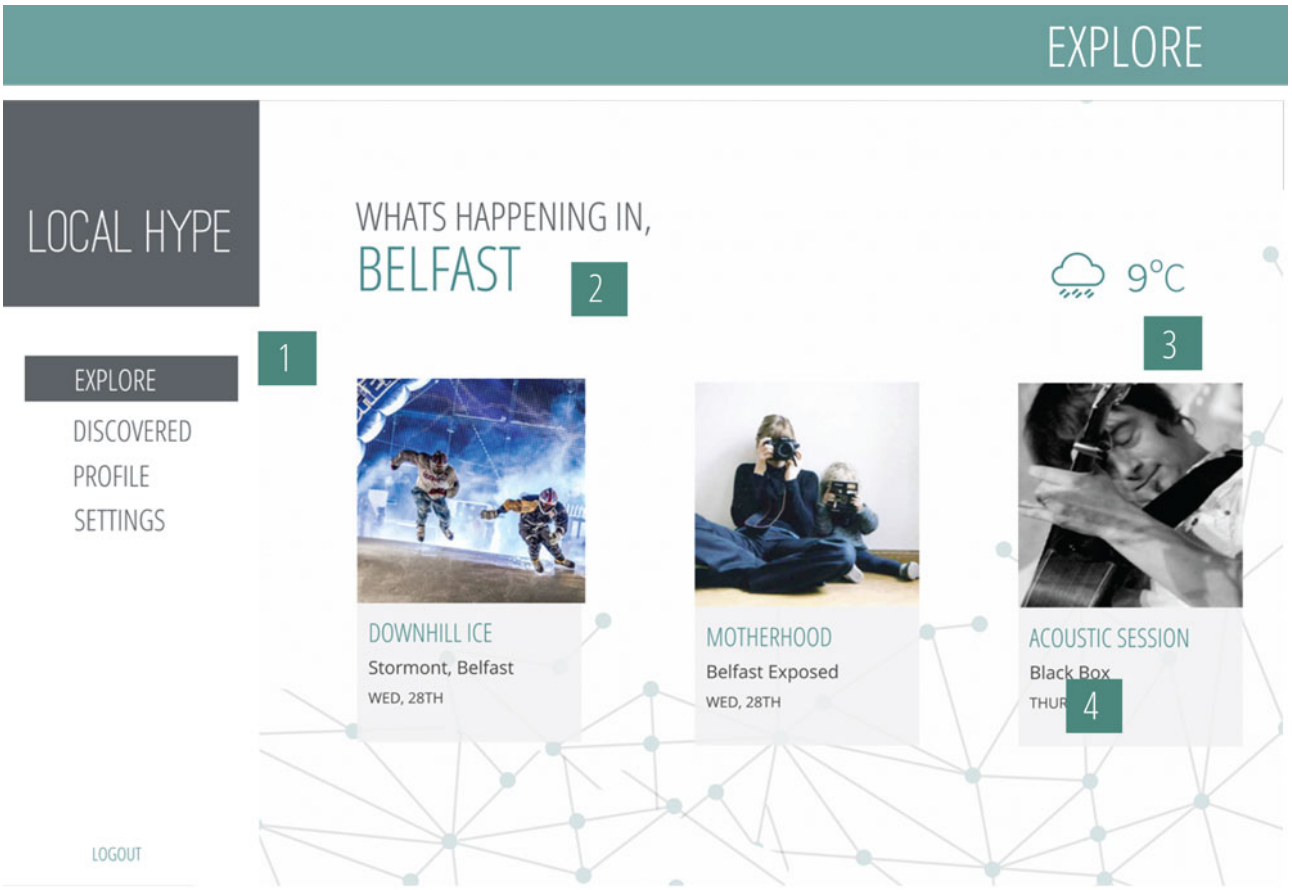
3 The desktop version contains a full video background that plays on a loop with no audio.

MOBILE

The mobile breakpoint will not make use of a video background.

On page load a random image will be placed in as the full page background. These images will be taken from the video used for desktop view





1 The current page will be highlighted to the user.

2 The location will change based on the users geolocation.

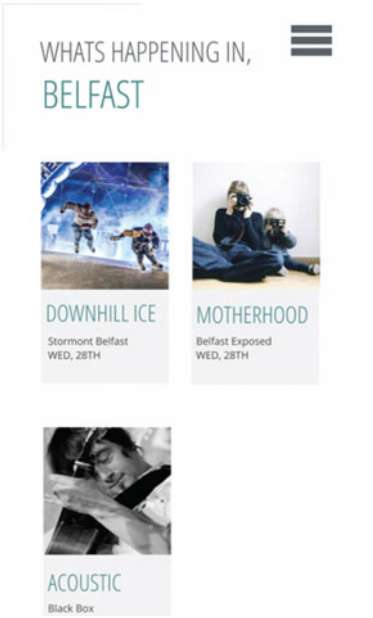
3 The current temperateure and weather type will be displayed dynamically based on location of user

4 Hovering on the panel will cause the it to flip 180 degrees and show the back facing panel.

MOBILE

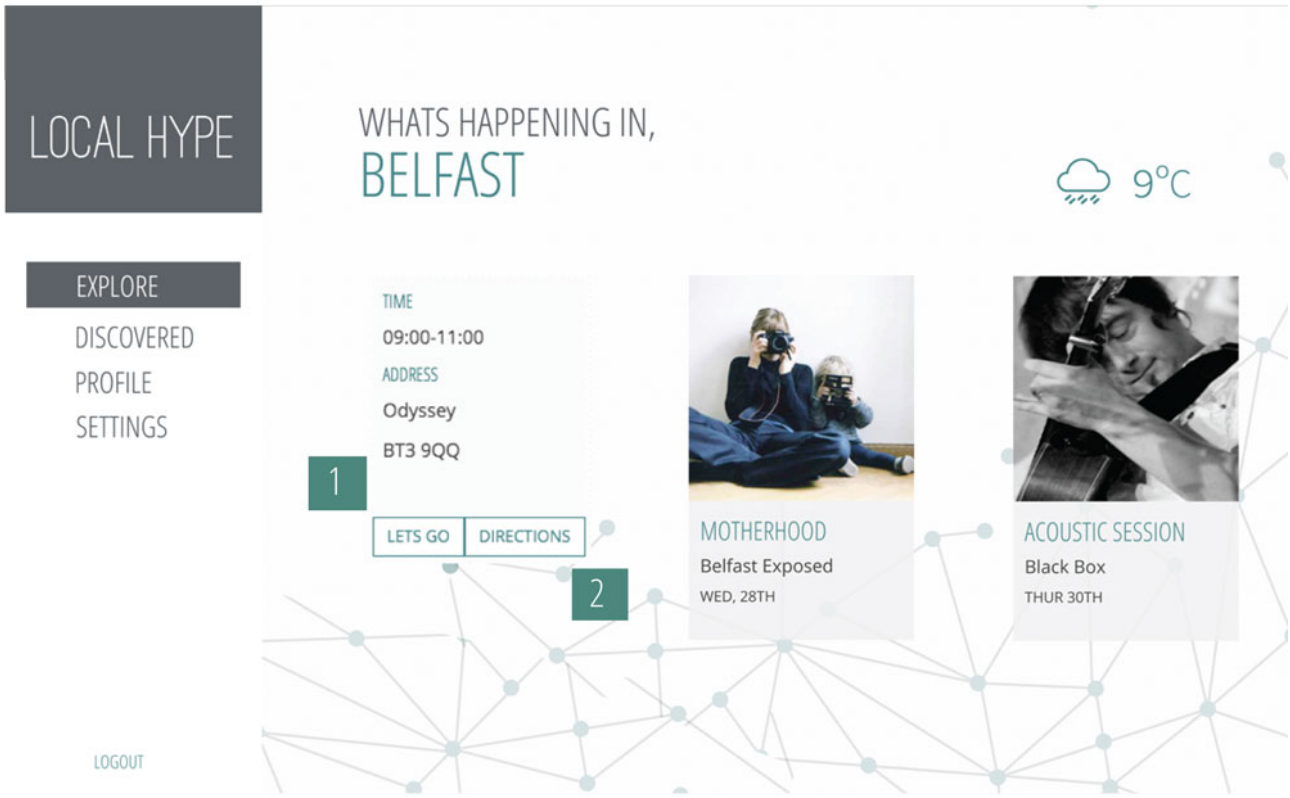
The mobile version places the naviagtion into a side menu. When clicked the menu will slide in from the left side.

The events will drop one below another so that the user can still see clearly the information and event graphic.



EXPLORE FLIP CARD

EVENT

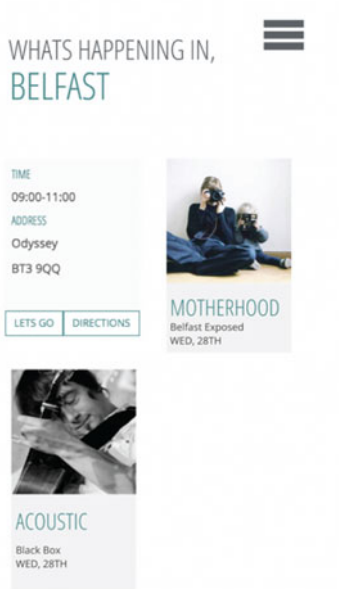


1 When the panel is flipped users will have the ability to click “lets go” and invite friends to the event selected.

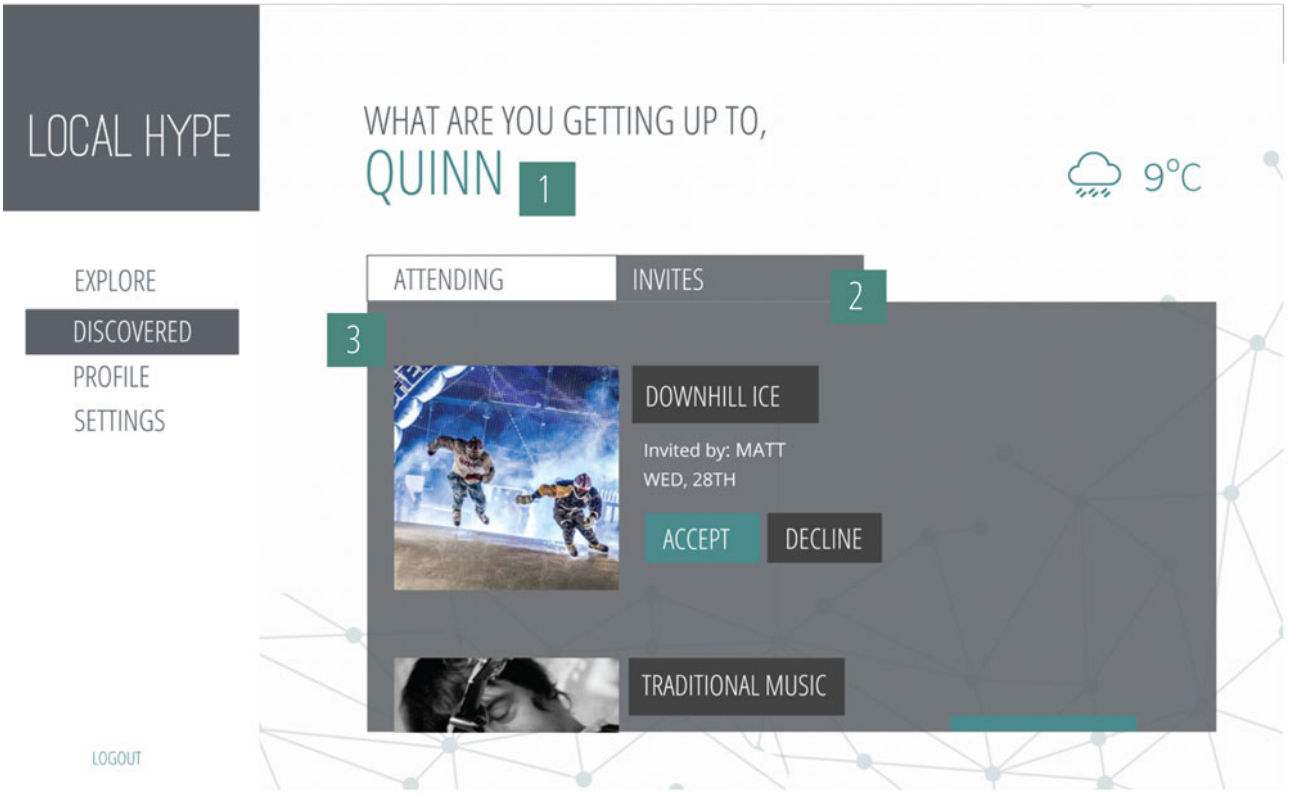
2 The user will get event directions using the users current location in relation to the event.

MOBILE

Events will be under one another by the date in which they are starting.



DISCOVERED



1 The users name will be displayed for the feel of personalisation

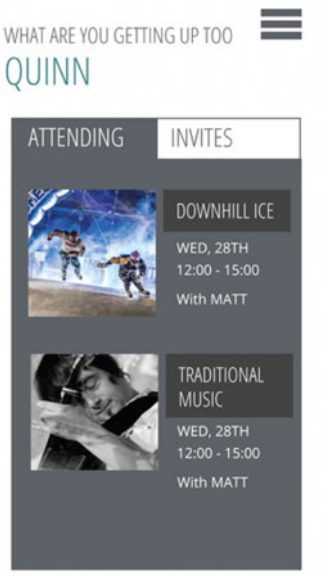
2 The invite tab shows a listing of events the user has been invited to. They remain there untill accepted or declined.

3 If an event is accepted it moves into the attending tab. The event once accepted fades out.

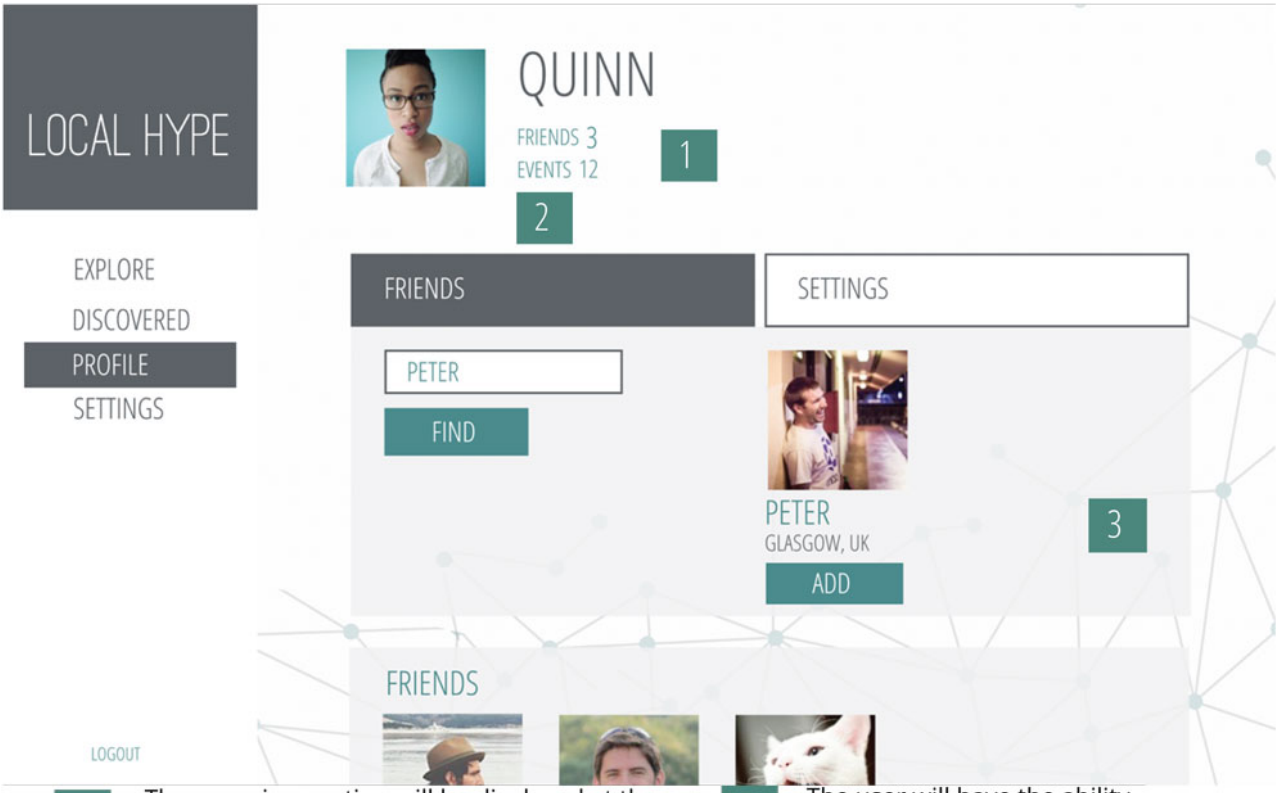
MOBILE

The tabs will decrease in width causing the information to drop one below another.

This listing of information insures the user can see all details easily.



PROFILE



1 The users inormation will be displayed at the top of the profile page.

2 The users info is tracked. The number of linked friends, and also the number of events attended.

3 The user will have the ability to search for a user name.
If a user exsists it will show to the right of the search box.
The abbility to add the user to the friends will send a friend request to that user.

MOBILE

When any of the buttons are clicked the page will toggle down showing information on the screen.

The users profile page also becomes full width of the page, placing the information on top of the graphic.