# Introduction

## Basic Concept

Wits marketing has asked us to create a mobile application which will allow the user to select on a map where they would like to go and from their current position to their destination, then it will calculate the shortest path and display it to the user.

## Hurdles

There are many hidden nuances involved in such an application. The main problem is converting raw GPS coordinates into image coordinates on the respective map, then mapping where the user currently is and drawing a pathway for them to follow which is then updated as they move along.

Another is how do we enable input of new maps, locations (with GPS) and location information with pictures.

## Some Ideas

I have some ideas to help solve these problems.

For the first hurdle, the plotting of GPS location data to image pixel coordinates is quite tricky. When the data is inputted from the input page, the user will have to input a map on one page. Perhaps even make the user input relative GPS coordinates for the points of the map so that we are able to calculate, with acceptable inaccuracy, what a pixel point on the map represents in GPS coordinates. Also the name of the Map, should be inserted. On another page the user will have a list of maps in the system and when one is selected, a picture of the map is displayed, zoomed in like on Google maps so that there will be buttons to enable the user to traverse the zoomed in map. The user will then be able to create points on the map. On the side of the page there will be form inputs for the point, where the user inputs the name, one or more code names (i.e. SH, but since there are some problems with Wits it needs to handle more than one- Do the same as you would for multiple emails by using delimiters.) and the GPS coordinates of the point. The third and final input page will show a list of points which are selectable and the user will be able to upload one image maybe more in a later version, and write a paragraph or two about the point/place to be displayed on the app to the user.

# Standards

## Security Issues

### HTML

Cross site scripting is an issue where instead of a user inputting plaintext into a String they instead insert HTML markup to allow the use of scripts such as JavaScript. This security flaw can be overcome by checking every string and using HTML & clause to handle all input as strings. This flaw would only be system local but should still be considered.

### MySQL and PHP

The main issue here is SQL injection. A user instead of entering plaintext may enter SQL statements to cause damage throughout the entire system. There are a few ways to negate this security flaw. One method is to add “” to the end of every string as it goes into the database. A better way would be to scrub the String so that no infected text goes into the database at all.

## Phonegap Updates

There are a few ways to handle updates to a phonegap app. It has a built in technology which can be added as an extra, but I don’t like it since it greatly decreases performance.

## Deliverables

### Milestone 1:

We need to create the database, some functional PHP and an input page. The page does not have to be complete but it must allow us to add data to the database. This is critical for future milestones.

I would also like to try and get the page which inserts points and locations, and images of the location.

For this milestone I the database and basic PHP should be handled by one person while the insert page and design be handled by another.

I would like Isaac to do the database and basic PHP code. The PHP for now must only allow for a site to upload a map image to the server, into a designated folder. It must also store data into the database about the map.

See Drawn UML Diagram

## Database:

MapTbl

* MapID (Primary Key and will be used as a foreign key later on) – Autonumber
* ImgLocation (Location of where the image of the map is stored) - String
* Corner1GPSLatitude (A Corner point latitude coordinate for the map) - double
* Corner1GPSLongitude (A Corner point longitude coordinate for the map) - double
* Corner2GPSLatitude (A Corner point latitude coordinate for the map) - double
* Corner2GPSLongitude (A Corner point longitude coordinate for the map) - double
* Corner3GPSLatitude (A Corner point latitude coordinate for the map) - double
* Corner3GPSLongitude (A Corner point longitude coordinate for the map) - double
* Corner4GPSLatitude (A Corner point latitude coordinate for the map) - double
* Corner4GPSLongitude (A Corner point longitude coordinate for the map) - double

LocationDataTbl

* LocationID (Primary Key and will be used as a foreign key later on) Autonumber
* TypeofPoint (There are three types of points. **Entrance/Exit**, to be connected to other maps ect, **Destination**, the buildings or destinations selected by the people inputting the data into the system and **NodalPathPoint**, which are the created points in the paths used in the adjacency list.) String
* MapID (Foreign key linking the point to a map)
* ImageCoordinateX (The coordinates of the point within the image of the map) Int
* ImageCoordinateY (The coordinates of the point within the image of the map) Int
* GPSLatitude (The GPS latitude coordinate of the point)
* GPSLongitude (the GPS Longitude coordinate of the point)
* Designation (Optional, will be NULL if the point is a **NodalPathPoint** or entrance/exit or there has been no data inputted, can handle more than one by using delimiter: **“ ,”**. String
* Paragraph (Optional, will be NULL if the point is a **NodalPathPoint** or entrance/exit or there has been no data inputted, some information about the destination) String
* LocationOfImage (Optional, will be NULL if the point is a **NodalPathPoint** or entrance/exit or there has been no data inputted, image of the destination) String

InternalMapTbl (this is to allow one map to be a map which is inside another, ie a map of sentae house within a map of East Campus, just for some extra features)

* InternalMapID\_PrimaryKey (Primary Key) Autonumber
* ExternalMapID (foreign Key from MapTbl)
* InternalMapID (foreign Key from MapTbl)

PathTbl (Adjacency List, which will store all the paths on all the respective maps)

* PathID (Primary Key) Autonumber
* CurrentNode (LocationID foreign key connection)
* PreviousNode (LocationID foreign key connection)
* ~~MapOfPath (MapID foreign key connection)~~ Not Needed since it will not make the DB in 3rd Normal Form

UserTbl (users who are allowed to add/remove data)

* UserID (Primary Key) Autonumber
* Username String
* Password String