Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name:
Student ID:
Supervisor:

SECTION 2 Confirmation of Authorship
The acceptance of your work is subject to your signature on the following declaration:
I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature:____ Date:____________

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

Complete the sections above and attach it to the front of one of the copies of your assignment,

What constitutes plagiarism or cheating?
The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.

Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks
and credit given to the author.

When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project

Penalties for Plagiarism
If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the college's Disciplinary Committee. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend

- that a student's marks shall be reduced
- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and
- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.

National College of Ireland

BSc in Computing

2015/2016

Antoin Judge

X12116670

antoin.judge@student.ncirl.ie

Mobile Timesheet

Technical Report

Table of Contents

Executive Summary

The objective of this project is to create a mobile application for ESB Ireland, a company which frequently has employees working in off-site locations. The application will be installed on the smartphones of ESB Ireland employees, and will allow them, while working off site, to record overtime hours and other expense items, such as meals and overnight stays. The application also uses the smartphones geo location and the Google maps distance matrix API to accurately calculate the mileage of the employee. The application will allow the user to input and record his expenses incurred as they happen with an overtime and expenses form which will be used to claim for any allowances incurred.

The completed timesheets will be stored in a database on a remote server which can be accessed by the project manager in order to calculate overtime and other payroll details. This database will have a user interface which will allow the manager to search for specific employee data and also to perform CRUD (create, read, update and delete) operations on the stored data.

1 Introduction

1.1 Background

The idea for the project began as the result of a conversation with my brother, Eoghan Judge, who is employed as HV cable manager with ESB Ireland. He manages a team of ten workers. When members of his team are working off site they are compensated for any expenses incurred, such as overtime, mileage travelled and meals. These expenses are submitted by each employee at the end of each week on an expenses and overtime form. He explained that one of his tasks currently is to collect physical hard-copies of these time-sheets from every employee each week and then manually update these details to a database to calculate overtime and allowances. This is a very time consuming process and is often made more difficult when an employee is not on site when payroll is being calculated.

1.2 Aims

This gave me the idea for an application which would allow employees to submit overtime and expense details from any location by simply using their smartphone. I explained to him what my idea for my project was and he was of the opinion that it would be a very helpful application from both his perspective and from his team's perspective also. I suggested that the application could also be used to provide a more accurate calculation of mileage expenses being claimed for, and he agreed that this would be an excellent improvement on the current system in place, and that the application would definitely be something that he would be interested in implementing.

I approached Eamon Nolan in the college to discuss my project and explained to him my proposed application. He agreed that it was a good idea and also provided feedback about some possible problems which may need to be addressed. He recommended that I should consider what will happen if the user does not have any coverage or internet access on their mobile device, how the application will deal with this and what are the possible solutions. I found his feedback to be very constructive and it will be taken into consideration as the project progresses.

1.3 Technologies

1.3.1 Java in Android Studio IDE

The actual mobile application is initially being developed for use on Android devices and I have chosen the Android Studio IDE as my development tool. My main reason for choosing Android Studio is the fact that it is now the official IDE for Android development and provides an easy to understand environment to develop in. Android studio also provides an SQLite database which I use to store the users recorded expense items locally on the mobile device before submitting to the server side database. Another reason for choosing Android Studio is the fact that it is based on the IntelliJ IDE which I have some experience working in from a previous college project.

1.3.2 MySQL

MySQL is a relational database management system which uses SQL (Structured Query Language) for adding, querying and managing the content in a database. All employee and application user data is stored in a MySQL database.

1.3.3 SQLite

SQLite is a server-less SQL database engine. It is embedded in the Android operating system. It stores data in a text fie locally on the device.

1.3.4 PHP

PHP is a server side scripting language which is used for web development purposes. Communication between the mobile application and the database was implemented using server side PHP scripts.

1.3.5 Github

Git is a version control system that allows multiple developers to work on the same code concurrently while still maintaining a master version. I hosted my git repository on Github, which provided a graphical user interface for managing my project. This allowed me to safely experiment with changes in the application on different branches of my repository while maintaining a master branch.

1.3.6 Bootstrap

Bootstrap is a free front-end framework for faster and easier web development; it contains HTML and CSS-based design templates for forms, buttons, and other interface components, as well as optional JavaScript extensions. It provides a clean consistent styling for web pages. I used Bootstrap to design the web based interface which will be used by the administrator to access the database. It is an excellent tool as it saves a lot of time in development and is responsive, which means that it adjusts to various screen sizes, such as tablets and mobile phones.

1.3.7 Apache Web Server

Apache is a free open source web server. I used Apache and phpMyAdmin to host my database locally on my laptop during development.

1.3.8 Google Maps API

The Google Maps API is an application program interface which provides a set of methods and tools which can be used to manipulate various aspects of Google Maps. I implemented the Google Maps Distance Matrix API in order to calculate the mileage traveled by the user when working off site. To use the API, a key must be acquired from Google which is then embedded into the applications code. I also implemented the Google Places Autocomplete API. This API gives the application a ‘search ahead’ feature, for example, when the user begins typing in the start location text field, autocomplete begins to provide potential location matches

1.3.9 jQuery

jQuery is a javascript library which simplifies event handling and Ajax interactions. It is used in the web interface for the database to load data into the page without requiring that another page is loaded.

1.4 Structure

The purpose of the first section of this report is to give the reader a general overview of what the project is about.

Included in this section is information detailing the background of the project and the reasons that it was chosen. It describes who the target market for the application is and who the stakeholder is. This section also gives details of meetings that were conducted with Eoghan Judge of ESB Ireland to determine the requirements for the application. In this section I also describe the technologies which were chosen to develop the application, and the reasons for choosing these technologies.

The second section deals with the requirements that were elicited following the meetings with the stakeholder. It describes briefly what the main functional requirements are; these are illustrated in greater detail later on, in the Requirements Specification, which is appended to the end of this report. It also explains the non-functional requirements of the application, such as data, environmental and usability requirements.

Following from that, this section describes the architecture of the system, and then describes in detail the implementation of this architecture. Some of the more important and more interesting aspects of the code are described also. It also describes the various stages of testing that was performed on the application, and explains some of the approaches which were taken to testing and some of the technologies which were used during this process.

The GUI of the mobile application and the database interface are also discussed. Screenshots are provided to demonstrate how the application looks and how it was designed.

The third and fourth section explains the conclusions that were arrived at on completion of the project and how gives examples of how some aspects may have been implemented better. Plans for any further developments are also discussed.

2 System

2.1 Requirements

In this section I will describe the requirements for the mobileTimesheet application, as were elicited from interviews with Eoghan Judge of ESB Ireland.

The stakeholder requires an application which will allow it to accurately monitor its employee's working hours, overtime and expenses claims when they are working off site. The stakeholder is unhappy with the present system in place to record this data. Currently, it is required that each employee complete a large timesheet in paper form at the end of the working week. This sheet must then be delivered in person to their manager. The manager must then populate a database with the information from each employee manually. The stakeholder requires a system which will reduce the time taken in completing these tasks by both the site workers and their superiors, as well as reducing environmental waste.

A feasibility study was undertaken to determine what the problems with the current system are, and to ensure that the proposed application meets the objectives required by ESB Ireland. Interviews were conducted with Eoghan Judge, HV cable manager for ESB Ireland, in order to elicit the following requirements

2.1.1 Functional Requirements

This section will deal with the functional requirements of the application. These requirements are described in greater detail, including use case diagrams in the Requirements Specification document which is included in the appendix. The functional requirements, in ranked order, for Mobile Timesheet are as follows;

1. User Login.
2. Create new account.
3. Submit timesheet.
4. Add expense item
5. Add mileage.
6. View current timesheet

2.1.2 Requirement 1: User Login

The user must be able to log into the system. Upon entering his valid credentials he must be displayed with his own personal homepage which allows him to carry out the various user functions of the application.

2.1.3 Requirement 2: Create New Account

If a user is using the system for the first time, he must be able to create an account on the system which will allow him to access the database. This is done by providing his employee ID number, his email address and creating a username and password. If his employee ID is valid then his details shall be saved in the users table of the database

2.1.4 Requirement 3: Add Expense Item

A user must be able to add an expense item he has incurred to his timesheet. This data must be stored on the device locally and must be accessible to the user when requested.

2.1.5 Requirement 4: Add Mileage

A user must be able to add a mileage expense claim to his Timesheet. He must be able to calculate the distance that he has travelled using the device, and then store this data in the local database.

2.1.6 Requirement 5: View Current Timesheet

The user must be able to view his current timesheet, containing all of the expenses that he has added to date. This timesheet must be accessible to him at any time

2.1.7 Requirement 6: Submit Timesheet

The user must be able to submit his completed daily worksheets to the database when completed. The database must be accessible to him at all times.

2.1.8 Data Requirements

The application shall implement an SQLite database to store all data locally.

All data stored on the server shall reside in a MySQL database .

All of the employee's passwords which are stored in the applications database should be encrypted.

Only registered users with the necessary privileges should be permitted to access this data.

The data of an employee should be available to him whenever he requests it.

In accordance with the data protection act, the personal information of employees should not be held for any longer than it is required.

2.1.9 User Requirements

The user interface shall be icon driven, with simple buttons and textboxes for user inputs

When a user clicks on any button in on the GUI of the application, the device must respond in no longer than two seconds.

If the user does not have network access he should still be able to access the application and store his details on the internal memory of his mobile device.

In the event that the user does not have network access and cannot in implement the journey calculation feature of the application, he should be able to manually enter the distance later, using the manual distance calculator.

2.1.10 Environmental Requirements

These are the environmental requirements that are essential when developing the Mobile Timesheet application.

- Android Studio IDE must be installed on the machine being used to develop the application.
- An Android smartphone is required in order to test the application throughout the development process.
- An API key is required in order to implement the various Google Developer API's which are utilized.
- Apache web server must be installed on the machine.
- PhpMyAdmin is required to manage the database used by the application.
- Apache web server is required to test the PHP scripts used to manipulate the data in the database.

2.1.11 Usability Requirements

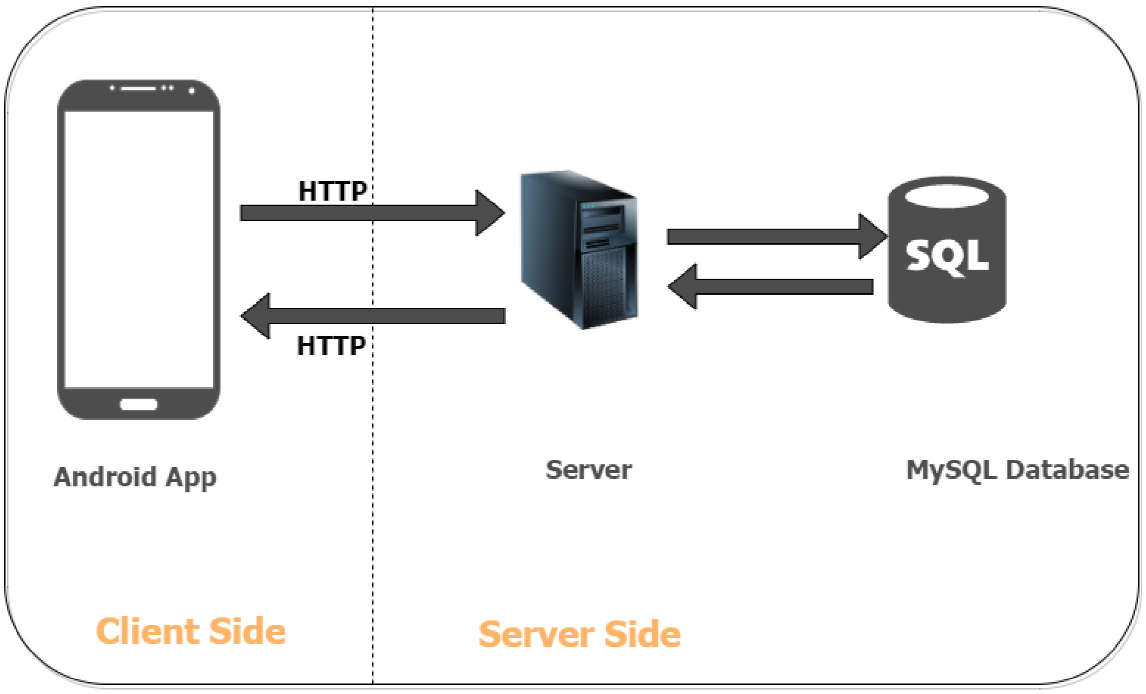
These are the requirements used as a basis for the design of the applications GUI, and were elicited during meetings with the stakeholder. These requirements should be measureable.

- An inexperienced user should be able to successfully navigate the application after no more than 2minutes.
- The application should be capable of operating on all Android devices.

2.2 Design and Architecture

Throughout the development process the database and PHP scripts have been hosted on an Apache web server using the EasyPHP package which allows me to run and manage these locally on my own machine. It is not possible to communicate directly with localhost from an android device without reconfiguring the wireless router in the network, so for the purpose of this project I am also hosting the database and files on *www.hostinger.co.uk*, in order to demonstrate how the system would look when integrated into an organisation's existing system.

- The application is implemented as client server architecture.
- The client side of the application is the users' smartphone, which displays the GUI.
- On the server side of the application are PHP web services which are used to communicate with a MySQL database.
- When the client wishes to read or write to the database, it sends a HTTP request to the server in JSON format using the GET or POST methods. If the client is writing to the database, it takes the parameters in String format and parses them to JSON format before sending to server. These JSON objects are then sent to the database by implementing the PHP webs services resting on the server.



Overview of Architecture

2.3 Implementation

For the application to function correctly it is vital that communication between the user's smart device and the server is successful. To achieve this communication I used the Android Volley HTTP library, which allows network communication to be performed with a relatively small amount of coding.

To post registration credentials to the server I use a Volley *StringRequest.* This passes the users' employee ID, password, username and email address as a String to the server address, and in return receives a String response. In this instance the address targeted, ***REGISTER_URL,*** is a PHP file residing on the server. Below is a sample of code demonstrating how this is method is implemented.

```
private void registerUser(){
//Getting users values from edit texts
final String username =
editTextUsername.getText().toString().trim();
final String password =
editTextPassword.getText().toString().trim();
final String email = editTextEmail.getText().toString().trim();
final String empid = editTextEmpid.getText().toString().trim();
//Creating a string request
StringRequeststringRequest = new
StringRequest(Request.Method.POST, REGISTER_URL,
new Response.Listener<String>() {
@Override
//Receives a response from the server indicating whether
registration is successful or not
public void onResponse(String response) {
//Displays the response to the user
Toast.makeText(MainActivity.this,response,Toast.LENGTH_LONG).show
();
                }
            },
new Response.ErrorListener() {
@Override
//Displays any error messages to the user
public void onErrorResponse(VolleyError error) {

Toast.makeText(MainActivity.this,error.toString(),Toast.LENGTH_LO
NG).show();
                }
```

Before the user can register an account, a script must first query the employee database to ensure that he has a valid employee number and that he is on the system, the code snippet below demonstrates how this is executed by the PHP script. The first part of the code gets the username, email, password and employee id from the user. It then query's the *employee* table of the database to check that the employee id exists. If the employee exists in the database, it adds the user's details to the *users* table of the database; otherwise it alerts the user that he has not been successfully registered. The purpose of this registration process is to grant the user access to input their data into the database.

```
<?php
if($_SERVER['REQUEST_METHOD']=='POST'){
 $username = $_POST['username'];
 $email = $_POST['email'];
 $password = $_POST['password'];
 $empid= $_POST['empid'];
require_once('dbConnect.php');


 $sql = "SELECT * FROM employees WHERE empid=$empid";
 $rs = mysqli_query($con,$sql);
 $data = mysqli_fetch_array($rs, MYSQLI_NUM);
if($data[0] > 1) {
   $newSql = "INSERT INTO users (username,password,email,empid)
VALUES ('$username','$password','$email','$empid')";
            if(mysqli_query($con,$newSql))
 {
echo "Successfully Registered";
 }
else
 {
            echo "Employee ID Does not exist in Database, contact your
administrator";
 }
 }
```

I needed to store data locally on the device. SQLite is an open source SQL database which is included in the Android operating system, so I utilised an SQLite database to store the data as it is added to the timesheet on the device. It was relatively easy to create the database on the device, requiring only a few lines of code;

```
protected void createDatabase(){
db=openOrCreateDatabase("ThisDailyTS", Context.MODE_PRIVATE,
null);
db.execSQL("CREATE TABLE IF NOT EXISTS times(empid INTEGER ,
basic INTEGER  NOT NULL DEFAULT 0,overtime INTEGER NOT NULL
DEFAULT 0, meals INTEGER NOT NULL DEFAULT 0, mileage INTEGER NOT
NULL DEFAULT 0, date TEXT PRIMARY KEY  NOT NULL, sent INTEGER NOT
NULL DEFAULT 0);");

}
```

The syntax for queries to the database is almost identical to MySQL so getting used to working with SQLite proved to be a relatively easy task.

Implementing API calls to the Google Distance Matrix service was the aspect of the applications development that I found to be most difficult and time consuming. Unlike the registration and login calls which were made using *StringRequests*, to make calls to the API I needed to use *JSONObjectRequests.* Before doing this I needed to acquire an API key from the Google Developers Platform which enables the holder to use the service. To get the distance of a journey I pass the names of the start and end locations to the API's URL, along with my key, as a String. This returns a JSON object response which needs to be parsed to find the information which I require for my application. For example, to calculate the distance from my home to the college, the String passed to the API would return the following JSON response;

```
{
   "destination_addresses" : [ "Custom House Quay, Dublin,
Ireland" ],
   "origin_addresses" : [ "34 Fountain Hill, Ashfield, Drogheda,
Co. Louth, Ireland" ],
   "rows" : [
      {
         "elements" : [
            {
               "distance" : {
                  "text" : "53.5 km",
                  "value" : 53503
               },
               "duration" : {
                  "text" : "42 mins",
                  "value" : 2505
               },
                "status" : "OK"
            }
        ]
    }
  ],
  "status" : "OK"
}
```

For the purpose of my application, I need to calculate the driving distance between the two locations, therefore the only element that I am interested in is the distance element. In order to retrieve this value from the JSON, I parse it until I have an integer value, *distKm.* This is achieved using the following code;

```
JSONArray array = response.getJSONArray("rows");

for(inti=0; i <array.length(); i++) {
JSONObjectmyJ = array.getJSONObject(i);
    String myD = myJ.getString("elements");

JSONArrayjsonObject = new JSONArray(myD);
for (intj = 0; j <jsonObject.length(); j++) {
        String theObj =
jsonObject.getJSONObject(j).getString("distance");

JSONObjectjObjt = new JSONObject(theObj);
        String finalDist = jObjt.getString("value");
        Integer myDist = Integer.valueOf(finalDist);
intdistKm = (myDist / 1000);
```

2.4 Testing

2.4.1 Unit Testing

Unit, or component, testing was carried out throughout the development lifecycle. The application was developed iteratively, meaning that small functionalities were implemented as the lifecycle progressed. After each iteration, unit testing was carried out to ensure that added components were functioning as required. For example, when the login component was added, it was tested by entering an incorrect username, but correct password to check if the correct outcome was achieved, which of course is failure to log in. The same test was done with an incorrect password but correct username, and again with both username and password being incorrect. Finally I enter legitimate values and test that login is successful. This type of testing was carried out for each individual component of the application, when faults were found they were rectified before the next iteration of development was carried out.

I found *Postman* to be a very useful tool to test my PHPfiles as well as the API calls that I was making. *Postman* is a Google Chrome application which launches on your browser for interacting with HTTP APIs. It provides a GUI which allows the user to construct requests such as *GET* and *POST* and read the responses from these requests. This was particularly useful when testing my own scripts as it did not require me to launch the application on an android device each time I needed to test the response from a request, it could be done very quickly on my laptop. It also provided a detailed description of raw JSON objects which were being sent from these API's, which is more difficult to do within the Android Studio IDE.

2.4.2 System Testing

The iterative approach described previously meant that after each unit was tested individually, I could then integrate them and test various components together. For example, when I had tested the login functionality and found that it was working effectively. I could then integrate this with the component of the system that allows the user to send their daily timesheet to the database. This function requires that the user's employee ID is stored on the device in *SharedPreferences,* which is a part of the login component, both of these units needed to function together.

I used a program called *Monkey* to stress test my application. *Monkey* is command line tool which runs on either the emulator, or your own device. You specify which package you want to test, and if you want, you can restrict testing to a single activity. You then specify the number of events to attempt to trigger. When it is run, *Monkey* then generates a random stream of events on the application, if the application crashes or receives an unhandled exception, monkey will stop and report the error. This is a very useful tool to see how the application will perform under pressure.

2.4.3 End User Testing

In order to get an outside opinion I asked a colleague in my current workplace to install the application on his smart phone and use it for a number of days. He input his working hours for each day and any extra hours worked. He owns a smart phone, but he confessed that he is not particularly enthusiastic when it comes to this type of technology. He reported no problems in using the application, and found the interface to be easily understood and manageable. He was particularly impressed with the distance calculator component of the application and the autocomplete function when typing in a location address.

I also installed the application on a friend’s device and asked him to use it for a week to record his own expenses and working hours. He is more astute when it comes to using applications like this and would use a lot of apps regularly on his smartphone. He also reported no problems in using the application, his only suggestion was that it could be more aesthetically pleasing, and perhaps needed some extra features. This was taken into consideration, however, the application has been developed to meet a set of requirements set out by the stakeholder and it is doing just that.

Apart from these testing scenarios, perhaps the most obvious thing that needed to be tested was the ability of the application to run on multiple devices with varying screen sizes. I installed the application on my own device, a Sony Xperia, and on a Samsung Galaxy S5, and also on a Samsung tablet. I concluded that the application performed equally well on all of these devices.

2.5 Graphical User Interface (GUI) Layout

The user interface is displayed on the smartphone of the user. Currently it is has been designed to function only on the Android operating system; however, should the application be successful, further development will allow it to be compatible with Apple and Windows devices also. I have included a number of screenshots of the GUI. The purpose of the Mobile Timesheet application is to provide a means for users to maintain their working hours and expenses; it is not for entertainment purposes. For this reason the GUI was designed with straightforward, uncomplicated features, using large logos as icons, which are easy to read and understand.

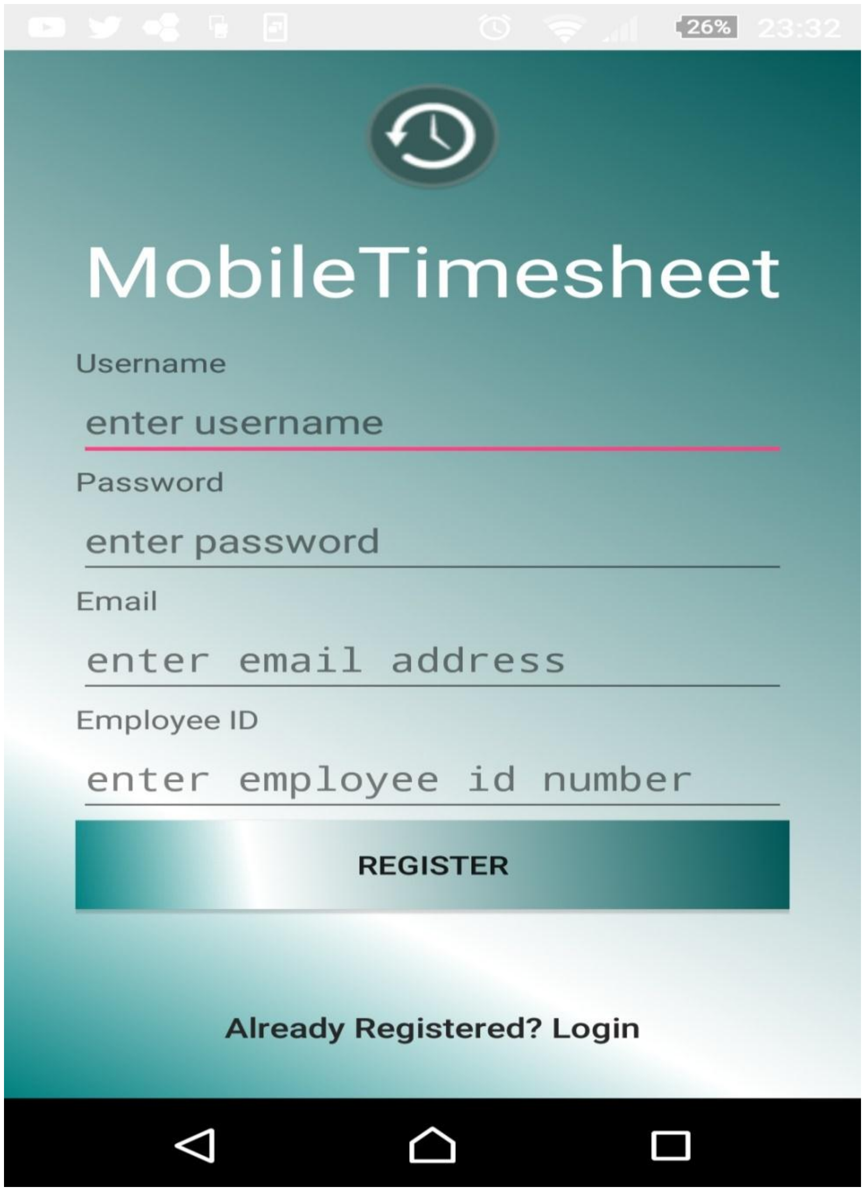


Figure 1

Figure 1 shows the screen that will be displayed to the user upon launching the application. It provides the user with a simple form to register using his employee ID number, email, username and password. Alternatively, if the user is already registered, he can click on the *login* text which will launch the login screen.

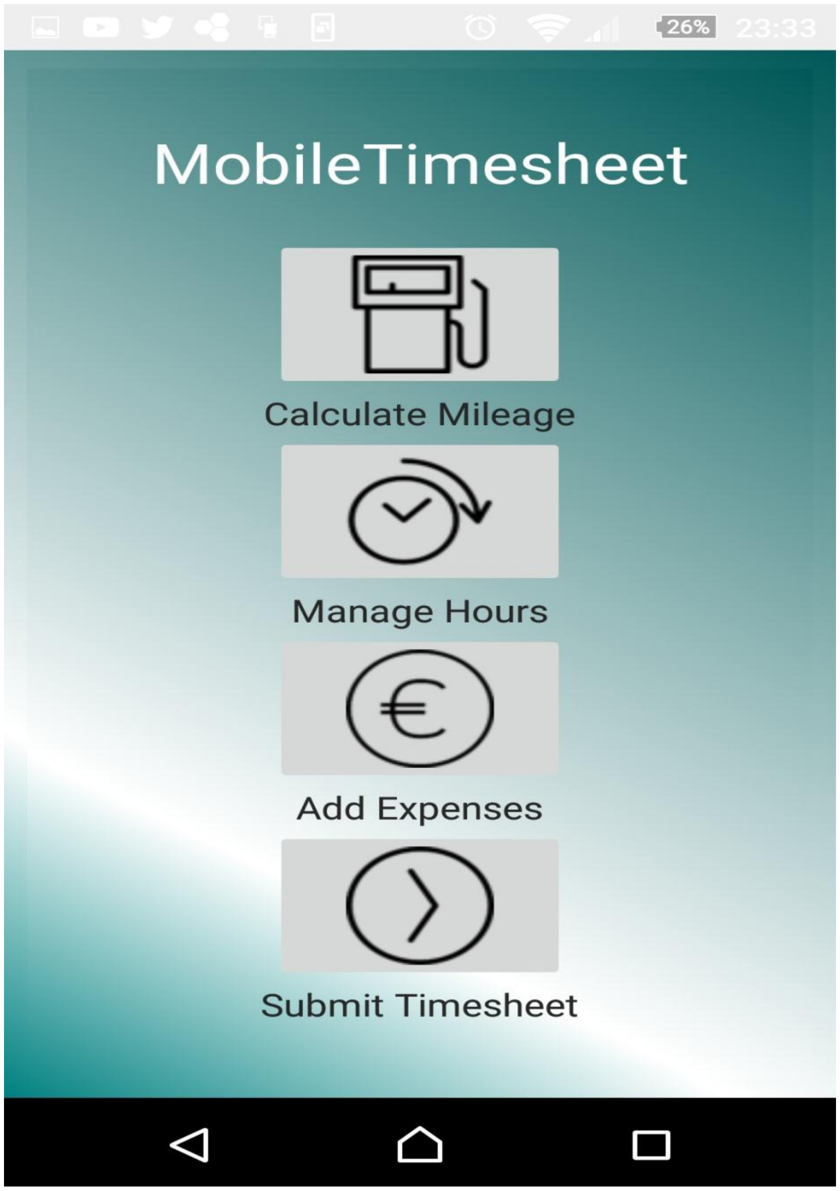


Figure 2

Figure 2 shows screen that is displayed to the user when he has successfully logged into the system. He is provided with four icons which launch activities that allow him to perform various functions on the device. He can calculate a journey, enter his hours, add an expense item, or submit his daily expenses to the server side database.

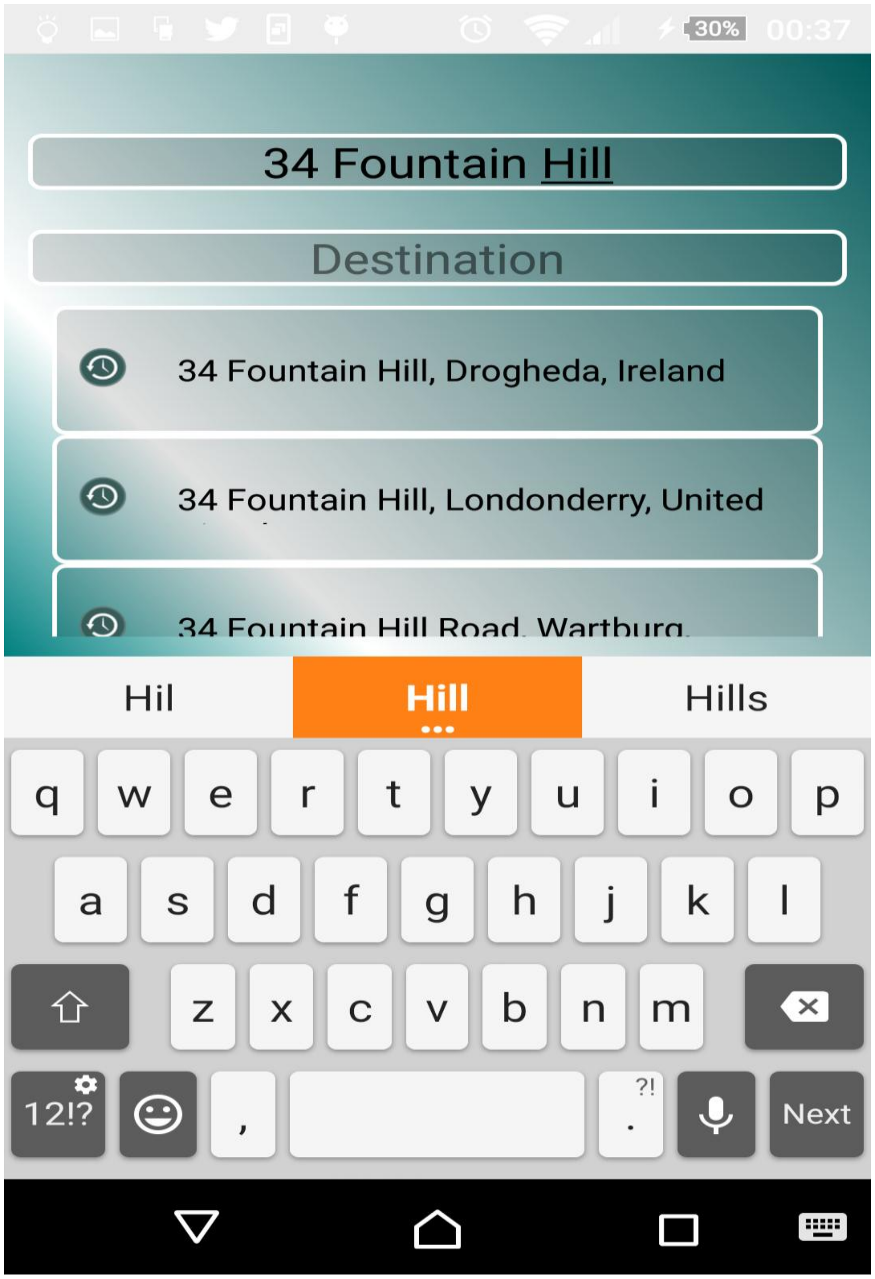
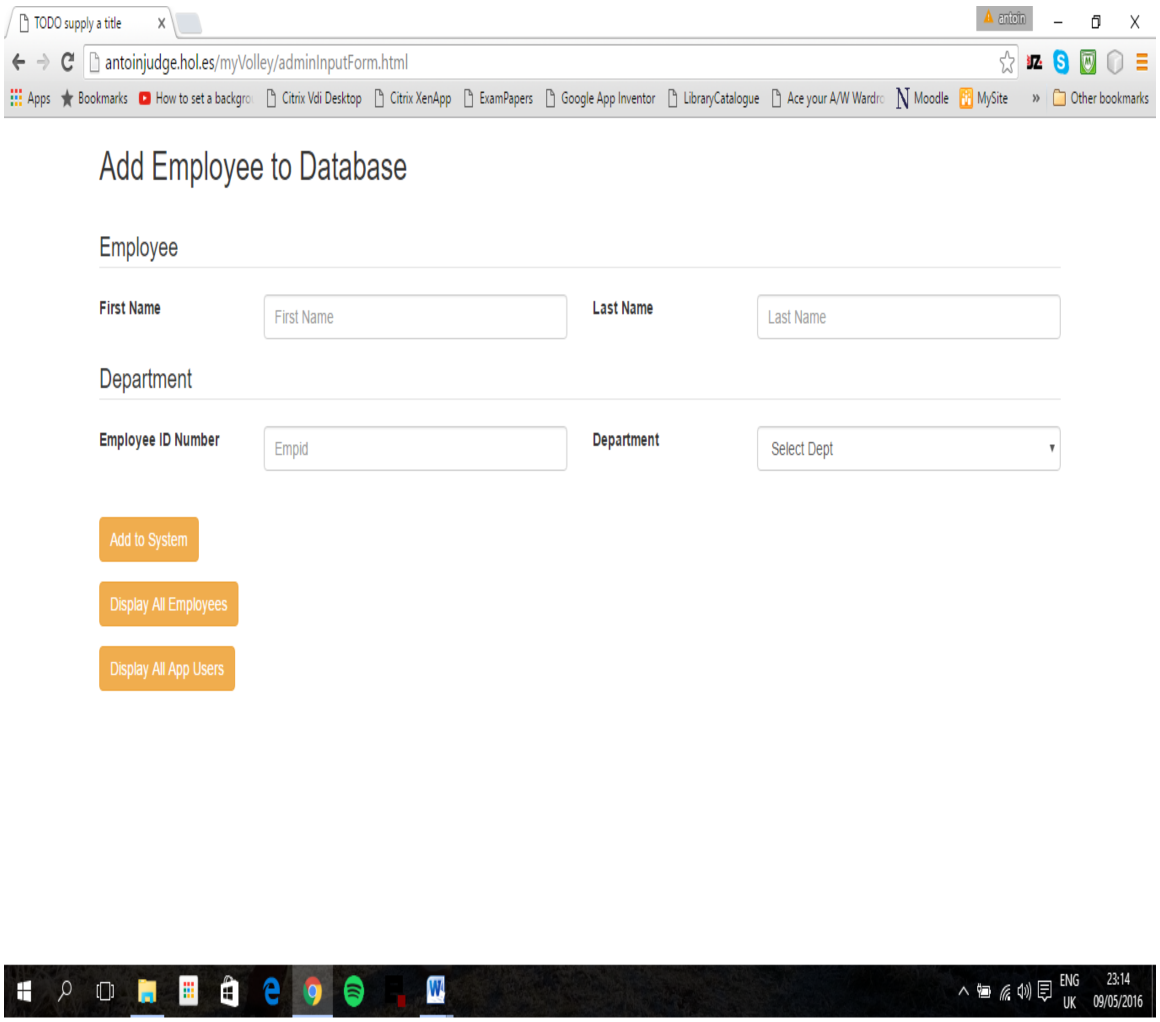

30%
00:37
34 Fountain Hill
Destination
34 Fountain Hill, Drogheda, Ireland
34 Fountain Hill, Londonderry, United
34 Fountain Hill Road, Wartburg,
Hil
Hill
Hills
Next

Figure 3

Figure 3 demonstrates the autocomplete feature of the application. As the user is typing an address into the text field, the application automatically displays suggestions of possible matching locations.



Add Employee to Database

Employee

First Name	First Name	Last Name	Last Name

Department

Employee ID Number	Empid	Department	Select Dept

Figure 4

Figure four is a screenshot of the server side database web interface. This screenshot demonstrates how an administrator can input an employee's data into the Employee Database. It is a simple design; *Bootstrap* was used to assist in the design of the web interface.

2.6 Customer Testing

After each iteration of the development process, any components which had been added were tested and then presented to Eoghan to gain his feedback and approval before moving on to the next stage. This was very valuable as it meant that when the final application was presented to him he did not require me to make changes to parts of the system which would affect other components. This is also an example of the value of an iterative approach to software development.

The final stage of the software testing process was user acceptance testing (UAE) which was carried out by Eoghan and a number of his work colleagues. These end users are ultimately the intended target of the application and it is their needs which the system has been developed to meet. All other testing to date has been done by myself and a number of friends, but this now gave me the opportunity to see how the system would work in a real-world working environment. Eoghan concluded that all feedback he received from his team was positive and there were not any problems which needed to be addressed. He was pleased with how the application had evolved and described how much more convenient it was for him to be able to access all of his teams timesheets from his laptop whenever he needed. The fact that timesheets were posted to the database each day rather than weekly meant that he could have a better understanding of the daily operations of each of his team at any time.

3 Conclusions

Overall I feel that the application has been a success. Developing a project like this from start to finish has been a fantastic experience, and while it has been difficult at times I feel that my development skills have increased tremendously as a result. I have particularly enjoyed watching how an idea, which was borne from a simple conversation with my brother, has developed into an actual application which is now on my smartphone and which, I feel, has the potential to be developed further commercially. I have been surprised by the reaction of a lot of people when I explain what my application does. People I've spoken to, who work in the type of organisation that has employees working off site, have been extremely enthusiastic about the app, and believe that it would be extremely beneficial to their own situations.

Looking back at the start of the project, when I created my project plan, I must admit that my forecasted milestones were extremely inaccurate. The actual development process was not anywhere close to how I had predicted in my plan originally. In hindsight I would have liked to have paid greater attention to this plan and aimed to meet the milestones I had set in the correct timeframe. It is only now, having completed this project that I can fully appreciate the necessity of a well devised project plan.

I can also appreciate now the importance of the early stages of the software development lifecycle. When we studied these stages of the development process I did not fully grasp just how crucial they are to ensure that a project progresses successfully, and as smoothly as possible. I think that these last two points are the greatest lessons that I take away from this project.

4 Further development or research

So far the response from the stakeholder has been positive, and if Mobile Timesheet is deemed to be a success, then a further version will be developed for use on iPhones and Windows operating systems also.

Following my last meeting with the stakeholder, I have decided that I will continue to develop this application, and that further features will be implemented. Eoghan explained to me that his role in ESB Ireland is HV cables and lines manager. This role involves him installing and commissioning underground cables of varying lengths, which include a number of underground joints. When each of these joints is completed, certain data needs to be recorded by the engineer before the underground joints are cemented over. These details include the GPS coordinates of the joint. Eoghan enquired would it be possible to implement a feature in the application which would allow the engineer to record the data, and get the exact GPS coordinates at the point of the joint and send these to a database. I had predicted that I would get to implement this feature in my application as an extra, however time was not sufficient and I have had to leave it out of the final project. I am confident that I will be able to add this feature in the near future without too much difficulty.

5 References

Android, Introduction. "Introduction To Android | Android Developers". *Developer.android.com*. N.p., 2016. Web. 9 May 2016.

"Google API Console". *Console.developers.google.com*. N.p., 2016. Web. 9 May 2016.

"Google Places API For Android | Google Developers". *Google Developers*. N.p., 2016. Web. 9 May 2016.

Tutorials accessed:

Segato, Gianluca. "An Introduction To Volley". *Code Envato Tuts+*. N.p., 2015. Web. 9 May 2016.

"Sending A Simple Request | Android Developers". *Developer.android.com*. N.p., 2016. Web. 7 May 2016.

Tamada, Ravi. "How To Connect Android With PHP, Mysql". *androidhive*. N.p., 2012. Web. 10 May 2016.

Appendix 1 – Project Proposal

Project Proposal

Mobile Timesheet

Antoin Judge

X12116670

antoin21ok@gmail.com

BSc (Hons) in Computing

Specialisation: Networking and Mobile Technologies

Date : 28/09/2015

Objectives

The objective of this project is to create a mobile application for a company which frequently has employees working in locations off site. The application will be installed on the smartphones of employees, and will allow them, while working off site, to track and calculate their mileage and various other expense items as well as overtime hours. The application will use the user's geo location and the Google maps API to accurately calculate the mileage of the user. The application will provide the user with an overtime and expenses form which will be used to claim for any allowances incurred.

When the user has entered his details as well as his login credentials, the application will allow this data from the completed form to be sent to a server side database. This database will then be used by the project manager or supervisor to calculate overtime and other payroll details.

The applications user interface should be attractive and user friendly.

Background

The idea for the project came from a conversation with my brother who is employed as a project manager with ESB Ireland. He manages a team of ten workers. When members of his team are working off site they are compensated for any expenses incurred, such as overtime, mileage travelled and meals. These expenses are submitted by each employee at the end of each week on an expenses and overtime form. He explained that one of his tasks currently is to collect physical hard-copies of these time-sheets from every employee each week and then manually update these details to a database to calculate overtime and allowances. This is a very time consuming process and is often made more difficult when an employee is not on site when payroll is being calculated.

This gave me the idea for an application which would allow employees to submit overtime and expense details from any location by simply using their smartphone. I explained to him what my idea for my project was and he was of the opinion that it would be a very helpful application from both his perspective and from his team’s perspective also. I suggested that the application could also be used to provide a more accurate calculation of mileage expenses being claimed for, and he agreed that this would be an excellent improvement on the current system in place, and that the application would definitely be something that he would be interested in implementing.

I approached Eamon Nolan in the college to discuss my project and explained to him my proposed application. He agreed that it was a good idea and also provided feedback about some possible problems which may need to be addressed. I found Eamons feedback to be very constructive and will take it into consideration as I progress with my project.

Technical Approach

The mobile application will be developed in the Android Studio IDE as I have used this when working on a previous project and found it to be a powerful tool and very easy to use. I was initially undecided about what language will be used when developing the server side. I have considering either PHP or Ruby. I am inexperienced in both of these languages but having researched them I have found a large range of online tutorials and lessons which will be of benefit to me. Having taken into consideration the learning curve that each of these languages will prove to be, I have decided that I will develop my server side using PHP.

I will be implementing a MySQL database to store my data.

When the user begins his journey his GPS coordinates will be recorded and when he reaches his destination the coordinates will again be recorded. I will use these locations and the Google Maps Distance Matrix API to calculate the total mileage of the user.

Special resources required

I will need to improve my understanding of the PHP programming language. To enhance my skills I will be availing of the numerous online resources and tutorials available.

Pluralsite has some excellent online courses in PHP which I am currently researching.

I am also using the Code Academy and Ruby Monk websites to help with my understanding of Ruby.

Technical Details

Java, XML, PHP and JavaScript will be the main languages which will be implemented**.**

Android Studio IDE will be used to develop the mobile application.

SQL queries to manipulate a MySQL database.

I am currently undecided on where my database will be hosted but am researching what options are available to me.

Evaluation

This application is being developed as a possible solution to problems encountered by my brother in his current position in ESB Ireland. For this reason he has agreed to test my application as progress is made and to provide feedback.

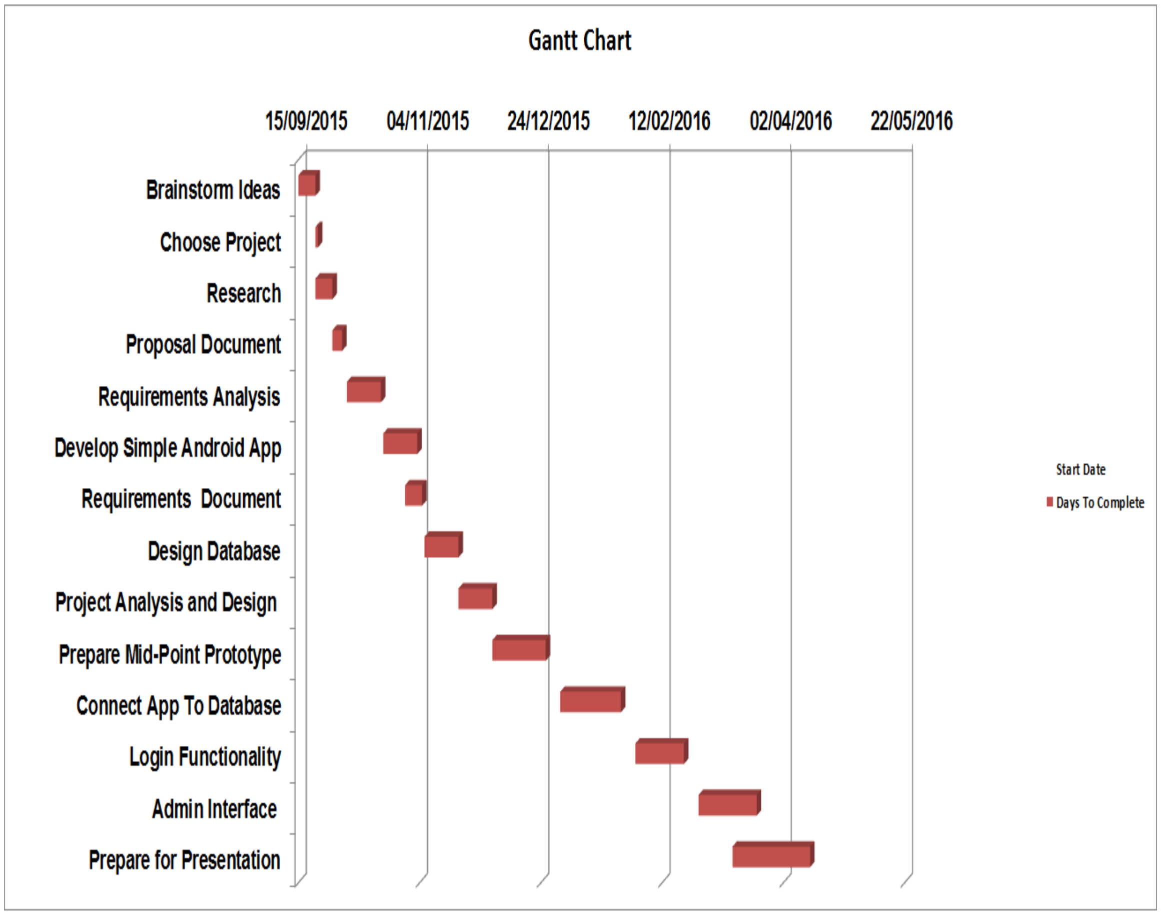
Antoin Judge 28/09/15

Signature of student and date

Appendix 2 - Project Plan

Project Plan

Task	Start Date	Days To Complete
Brainstorm Ideas	15/09/2015	7
Choose Project	22/09/2015	1
Research	22/09/2015	7
Proposal Document	29/09/2015	4
Requirements Analysis	05/10/2015	14
Develop Simple Android App	20/10/2015	14
Requirements Document	29/10/2015	7
Design Database	06/11/2015	14
Project Analysis and Design	20/11/2015	14
Prepare Mid-Point Prototype	04/12/2015	22
Connect App To Database	01/01/2016	25
Login Functionality	01/02/2016	20
Admin Interface	27/02/2016	24
Prepare for Presentation	12/03/2016	32


Gantt Chart
15/09/2015
04/11/2015
24/12/2015
12/02/2016
02/04/2016
22/05/2016
Brainstorm Ideas
Choose Project
Research
Proposal Document
Requirements Analysis
Develop Simple Android App
Requirements Document
Design Database
Project Analysis and Design
Prepare Mid-Point Prototype
Connect App To Database
Login Functionality
Admin Interface
Prepare for Presentation
Start Date
🟥 Days To Complete

Requirements Specification (RS)

Document Control

Revision History

Date	Version	Scope of Activity	Prepared	Reviewed	Approved
01/11/2015	1				
07/11/2015	2				

Distribution List

Name	Title	Version
Antoin Judge	Student	1

Related Documents

Title	Comments
Title of Use Case Model	
Title of Use Case Description	

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to set out the requirements for the development of the MobileTimesheet smartphone application. It will explain the purpose of the application and also all of its features.

The intended customer is ESB Ireland.

`1.2 Project Scope

The scope of the project is to develop a mobile application which will allow employees of ESB Ireland to record their working hours, overtime, mileage and other expenses on their smartphone and submit these to a centralised database at the end of the working week. It is intended that the application will replace the current system in place. The application shall have a GUI on the smartphone of the user. The application shall have a server side database to store the employee working hours and expenses data. The application shall have a web based interface which allows the system administrator to perform create, read, update and delete actions on the database. The application shall have login functionality. The application will be developed for Android devices; therefore the programming language used shall be Java.

A feasibility study was undertaken to determine what the problems with the current system are, and to ensure that the proposed application meets the objectives required by ESB Ireland. Antoin Judge conducted a series of interviews with Eoghan Judge, HC cable manager for ESB Ireland, in order to elicit the following requirements.

1.3 Definitions, Acronyms, and Abbreviations

ESB Ireland: ESB (Electricity Supply Board) is a semi state owned electricity company in Ireland.

GUI: Graphical User Interface. An interface which allows the user to interact with the system through the use of graphical icons and indicators

Site Worker: This term is used to describe users of the application who are employees of ESB working on sites away from the main organisation buildings. This term is used instead of simply “user”, to differentiate between an administrator using the system and a worker using the application to submit his timesheet.

Timesheet: This describes a document which is used to record all working hours and expense items of employees of ESB Ireland in a given week.

CRUD: This refers to operations performed on a database. CRUD is an acronym for, Create, Read, Update and Delete.

API: Application program interface. A set of routines, protocols and tools for developing software applications.

Google Maps API: An API which allows the user to display Google maps on their application.

Google Location Services API: An API which provides the longitude and latitude coordinates of the user's device when requested.

MySQL: An open source relational database management system.

Timesheet: A document which is completed by employees of ESB Ireland, allowing them to enter the number of hours they have worked as well as overtime and expenses incurred for each week.

2. User Requirements Definition

The stakeholder requires an application which will allow it to accurately monitor its employee's working hours, overtime and expenses claims when they are working off site. The stakeholder is unhappy with the present system in place to record this data. Currently, it is required that each employee complete a large timesheet in paper form at the end of the working week. This sheet must then be delivered in person to their manager. The manager must then populate a database with the information from each employee manually. The stakeholder requires a system which will reduce the time taken in completing these tasks by both the site workers and their superiors, as well as reducing environmental waste.

3. Requirements Specification

Here I will describe the requirements for the mobileTimesheet application, as were elicited from interviews with Eoghan Judge of ESB Ireland.

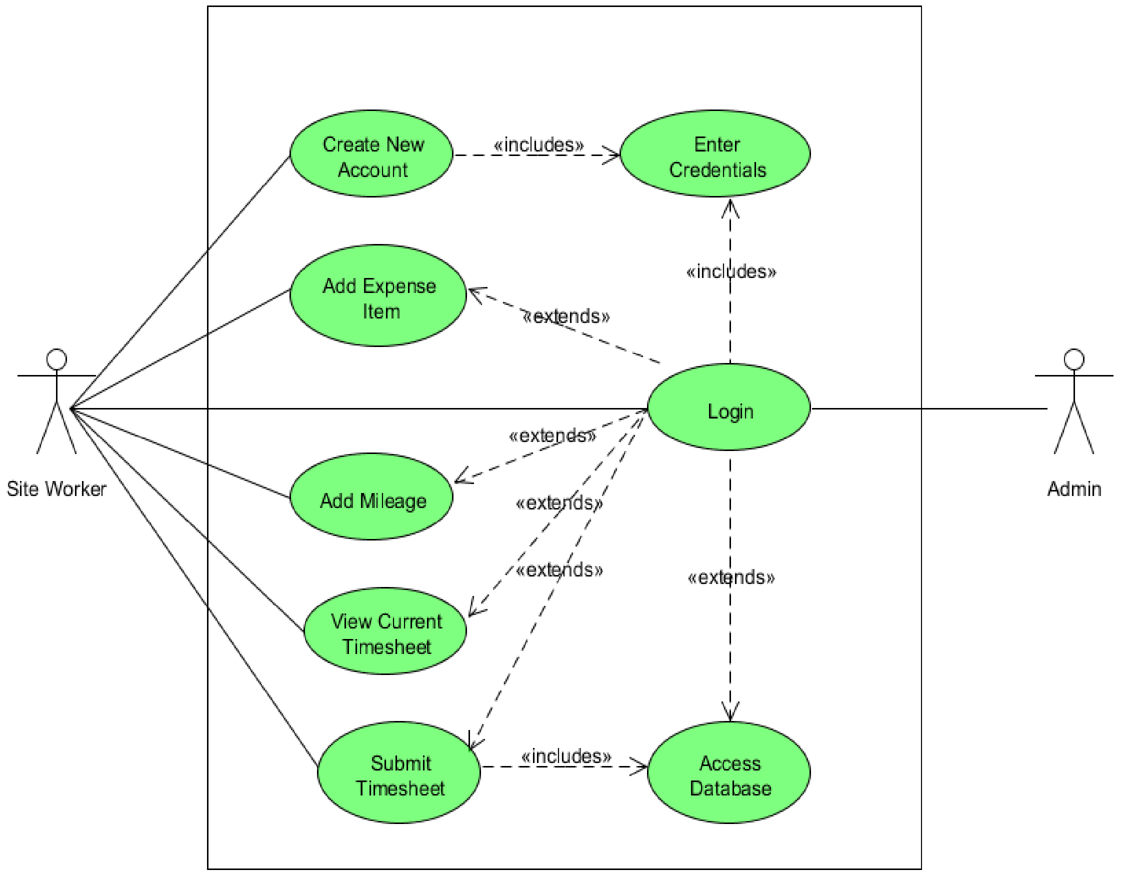
3.1 Functional requirements

The functional requirements, in ranked order, for Mobile Timesheet are as follows;

7. **Login.**
8. **Create new account.**
9. **Submit timesheet.**

10. **Add expense item**
11. **Add mileage.**
12. **View current timesheet.**

3.1.1 Use Case Diagram



3.1.2 Requirement 1: User Login

- **Description & Priority**

This use case describes how a user logs into the system. It is a high priority as the user must be logged in before the system can be used.

- **Use Case**

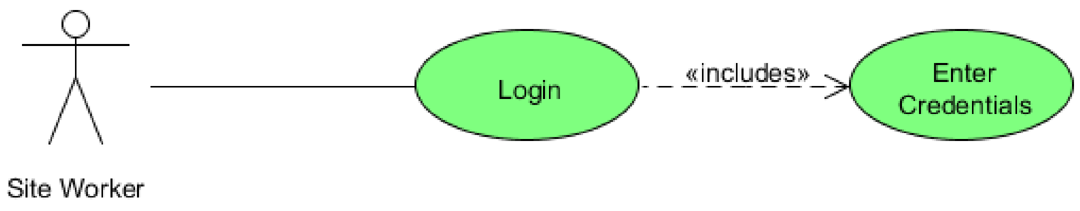
Scope

The scope of this use case is that the user is able to login to the application.

Description

This use case describes the user logging into the application by entering his username and password

Use Case Diagram



Flow Description

Precondition

The user must already be registered.

Activation

This use case starts when a user opens the home page of the application

Main flow

1. The user opens the application.
2. The user is directed to the login page.
3. The user enters his username.
4. The user enters his password.
5. The user's personal page is displayed.

Alternate flow

A1: Invalid credentials entered.

1. The system alerts the user that the username or password entered is invalid and prompts for the user to re-enter his credentials.
2. The use case continues at position 3 of the main flow

Exceptional flow

None

Termination

The application displays the user's homepage.

Post condition

The system goes into a wait state

3.1.3 Requirement 2: Create New Account

- **Description & Priority**

This use case describes how a user registers an account on the system for the first time and creates his profile. It is high priority as the user cannot use the application fully until he has created an account.

- **Use Case**

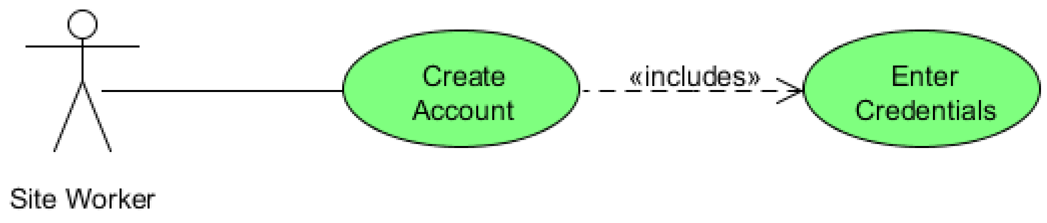
Scope

The scope of this use case is that a user can create an account on the system.

Description

This use case describes a user creating an account by entering his employee ID number and choosing a password.

Use Case Diagram



Flow Description

Precondition

The user's device must have an internet connection. The user must have a valid employee ID number.

Activation

This use case starts when a user clicks on the "Create account" button.

Main flow

1. The user clicks the "create account" button.
2. The application prompts the user to enter his employee identification number.
3. The application prompts the user to choose a password.
4. The ID and password are sent to the server side database for verification.
5. The new account is created.

Alternate flow

A1: Invalid Employee ID

1. The employee ID does not match any employees in the database
2. The user is alerted that the ID entered is invalid.
3. The user is prompted to enter another ID number.
4. The use case continues from position 2 of the main flow.

Exceptional flow

None

Termination

The application displays the user’s homepage.

Post condition

The system goes into a wait state.

3.1.4 Requirement 3: Add Expense Item

- **Description & Priority**

This use case describes how a user adds an expense item to his current timesheet. It is high priority as it is essential that the user is able to record his daily expenses.

- **Use Case**

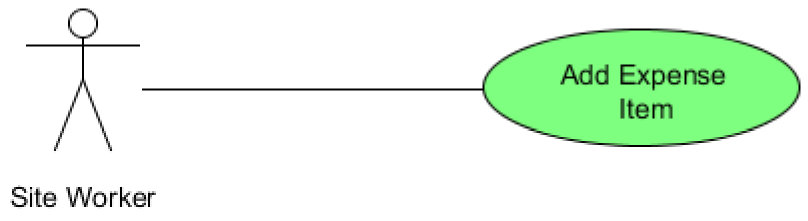
Scope

The scope of this use case is that a user can add an item to the expenses field of his weekly timesheet.

Description

This use case describes how a user can record an expense item that he has incurred.

Use Case Diagram



Flow Description

Precondition

The user's device must have an internet connection. The user must be logged in.

Activation

This use case starts when a user clicks on the "Add Expense Item" button.

Main flow

1. The user clicks the "Add Expense Item" button.
2. The application displays a drop down menu which allows the user to choose what expense he is claiming for.
3. The expense item is stored on the user's timesheet.

Alternate flow

None

Exceptional flow

None

Termination

The application displays the user's homepage.

Post condition

The system goes into a wait state.

3.1.5 Requirement 4: Add Mileage

- **Description & Priority**

This use case describes how a user adds a mileage claim to his current timesheet. It is high priority as it is essential that the user is able to record his daily mileage expenses.

- **Use Case**

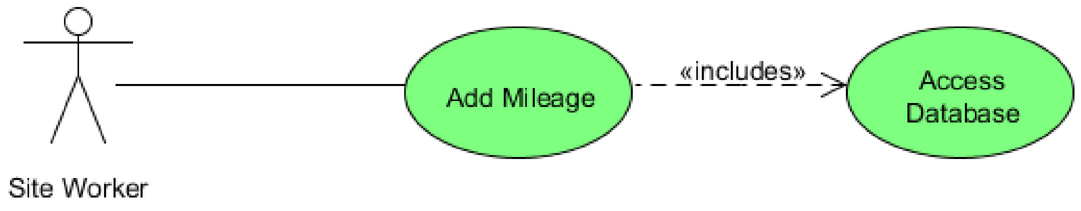
Scope

The scope of this use case is that a user can add a mileage amount to the mileage field of his timesheet.

Description

This use case describes how a user can record an expense item that he has incurred.

Use Case Diagram



Flow Description

Precondition

The user’s device must have an internet connection. The user must be logged in.

Activation

This use case starts when a user clicks on the “Add Mileage” button.

Main flow

1. The user clicks the “Add Mileage Item” button.
2. The application displays a form which prompts the user to enter the distance he has travelled today.
3. The user enters a value for his travelled mileage.
4. The mileage value is added to the user’s timesheet.

Alternate flow

A1: Invalid data type

1. The data type entered for mileage does not match the required data type.
2. The user is alerted that the data type entered is invalid.
3. The user is prompted to enter another value for mileage, a double.
4. The use case continues from position 2 of the main flow.

Exceptional flow

None

Termination

The application displays the user’s homepage.

Post condition

The system goes into a wait state.

3.1.6 Requirement 5: View Current Timesheet

- **Description & Priority**

This use case describes how a user views his current timesheet on his device, showing all of the expense details he has already recorded for the current week. It is high priority as it is important that the user is able to keep a track of his timesheet throughout the week and modify it if necessary.

- **Use Case**

Scope

The scope of this use case is that a user can view his current timesheet.

Description

This use case describes how a user can view all of his expense items recorded for the current week.

Use Case Diagram

View Current Timesheet
Site Worker

Flow Description

Precondition

The user's device must have an internet connection. The user must be logged in.

Activation

This use case starts when a user clicks on the "View Current Timesheet" button.

Main flow

1. The user clicks the "View Current Timesheet" button.
2. The application displays the current timesheet populated with the data already entered for the current week.

Alternate flow

None

Exceptional flow

None

Termination

The use case terminates when the user closes the current timesheet and the user's homepage is displayed.

Post condition

The system goes into a wait state.

3.1.7 Requirement 6: Submit Timesheet

- **Description & Priority**

This use case describes how a user updates the database with his working hours, mileage and expense items for the week. It is high priority as the main purpose of the application is to allow users to submit their timesheet on a smartphone.

- **Use Case**

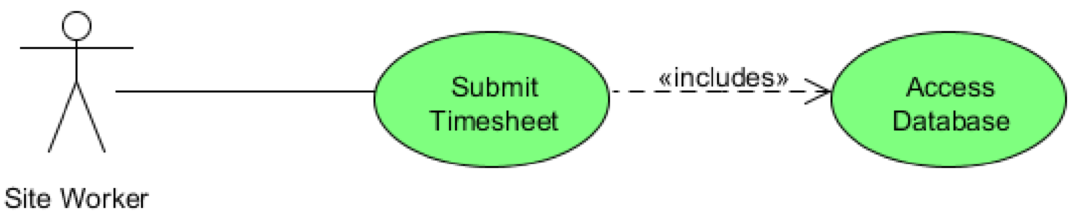
Scope

The scope of this use case is that a user can submit his completed timesheet to the database.

Description

This use case describes how a user can view a timesheet, with all of his already recorded expense items for the current week and then submit this timesheet.

Use Case Diagram



Flow Description

Precondition

The user's device must have an internet connection. The user must be logged in. The user's timesheet must be complete.

Activation

This use case starts when a user clicks on the "Submit Timesheet" button.

Main flow

1. The user clicks the “Submit Timesheet” button.
2. The application displays the current timesheet populated with the data already entered for the current week.
3. The user is prompted to review the timesheet before submitting
4. The user clicks the “submit” button.
5. The database is populated with the user’s timesheet data.

Alternative flow

A1: Incomplete Timesheet

1. The timesheet that the user has tried to submit is incomplete.
2. The user is alerted that the timesheet is incomplete.
3. The user is prompted to finalize the timesheet and re-submit it.
4. The use case continues from position 3 of the main flow.

Termination

The use case terminates when the timesheet has been submitted and the database has successfully been updated with the user’s data.

Post condition

The system goes into a wait state.

3.2 Non-Functional Requirements

3.2.1 Performance/Response time requirement

- When the user requests to view his current timesheet, it should take no longer than 3 seconds for the data to be displayed on the device.
- When the user submits his weekly timesheet, it should take no longer than 10 seconds for the database to be updated.

3.2.2 Availability requirement

- The server should be running at all times allowing the user to access the database whenever required.
- As long as the user has an internet connection he should be able to access the system once logged in.
- If the user does not have an internet connection he should still be able to add an expense item to his timesheet. This should be stored locally on the device until a connection is available.

3.2.3 Recovery requirement

- The database should be backed up daily. In event of a failure, the database will be restored to its last recorded consistent state.

- If the users device fails while updating the database, the database should roll back any changes that have not been committed. This will ensure database integrity.

3.2.4 Robustness requirement

- In the event of the application failing to connect to the database, the system should not crash; the user should be alerted with an error message.

3.2.5 Security requirement

- Only users who have verified login credentials should be able to access the system.
- Users should only be able to access data related to them from the database. It should not be possible for them to view other employee’s records.
- Only users with administrator rights should be able to gain full access to the database.
- Users with administrator rights should be able to perform CRUD operations on all tables of the database.

3.2.6 Reliability requirement

- The application should not fail more than twice in any given month.

3.2.7 Maintainability requirement

- Should the server database fail, it should take no longer than 1 hour to have it functioning correctly again.
- In the event of the server failing, it should require only one engineer to repair and return to full functionality.

3.2.8 Extendibility requirement

- Should the application prove to be successful, it should require no more than 5 weeks and 2 software developers to expand the system and database to include all departments of ESB Ireland.

4. Interface requirements

GUI

The user interface will be displayed on the smartphone of the user. Currently it is being designed to work only on Android devices; however, should the application be successful further development should allow it to be used on Apple and Windows devices also.

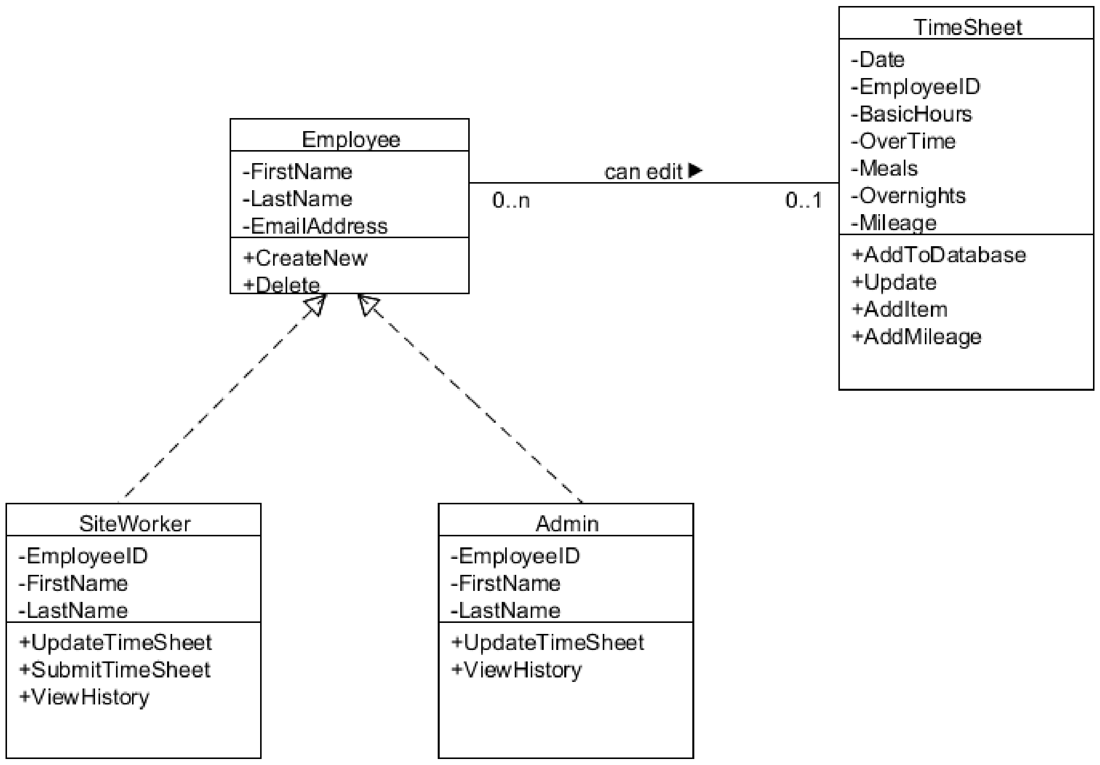
It should be simple in appearance with the ESB logo displayed in the header of all pages. The user’s homepage will have buttons which allow the user to add expense items, view their current timesheet and submit their timesheet.

Application Programming Interfaces (API)

The application will implement the Google Maps API as a web service to allow the user to calculate the mileage between his source and his destination.

In the event that the user is working in a remote area, their GPS coordinates will be will be revealed using the Google Location Services API.

5. System Architecture



This a simple overview of the class structure of the application. The completed timesheets will be stored in a MySQL Database.

6. System Evolution

Currently the application is being developed for android devices. If the application is deemed to be successful by the stakeholder, then a version for iPhone and a version for Windows devices will also be developed.

The application is being designed with ESB Ireland as the stakeholder; however it is foreseeable that if successful, Mobile Timesheet could be tailored to meet the requirements of many other organizations.

Appendix 3 - Monthly Journals

Reflective Journal

Student name: Antoin Judge

Programme: BSc in Computing

Month: October 2015

My Achievements

Having submitted my project proposal I spent the next few weeks trying to figure out how I was going to develop my application. My previous projects in the college have not been as large as this and the fact that I am flying solo with this one is quite daunting. I have been attempting tutorials covering Ruby on Rails and PHP to try to determine what language I am going to use for the back end of the application. I am still none the wiser as to what technology I will use to design my server although I know that I will be using Android studio for the actual app, having used this on a previous project.

Eamon spoke in class about how some students are having difficulty mastering new technologies. He told us that he doesn't think that we should actually be writing our code yet and that we should concentrate on our next deliverable, the requirements document. I took Eamon's advice and began to write my requirements specification document. I went back through all of my Software Engineering and Object Oriented Software Engineering slides from year two and tried to remember how to draw class diagrams and use case diagrams.

I had my first meeting with Padraig on Wednesday evening in the college. I showed him what I had done on my document and he was very helpful in advising me in areas where I might be going wrong. We agreed to meet in the college once every fortnight, I found it to be a helpful meeting overall and think that his guidance will be very useful throughout the next few months.

My Reflection

I was relieved to have my first meeting with Padraig as I was worried that when I explained to him what my project consists of he would tell me that I needed to rethink the whole thing, gladly he didn't and seemed to think that I had quite a good idea.

However, I am still not decided on what technologies I will be using for my back end. I think that I need to figure this out very soon.

Intended Changes

Next month, I will continue with the online tutorials I have been doing. I will begin developing the actual Android application; at least this will get me started on the coding side of things.

I realise that I need to improve my coding skills greatly if I am to make a decent attempt at this project, but I am confident that I can do it.

Supervisor Meetings

Date of Meeting: 4/11/15

Items discussed: Security for the application, Requirements specification document, use case diagrams.

Action Items: Decide how I will deal with the security aspect of the application. How will I develop my database?

Reflective Journal

Student name: Antoin Judge

Programme: BSc in Computing

Month: November 2015

My Achievements

At the beginning of this month I finished and submitted my requirements document. I was not overly happy with my submission as I felt that I had rushed it slightly towards the end just to get it submitted on time. I had another meeting with Padraig and we discussed my requirements document. Padraig gave me some tips on how I could improve my submission and I was very happy to learn that I would have the opportunity to alter my document and re-submit it if I wished to do so.

Eamon gave a seminar in which he demonstrated how to connect to a database using the EasyPHP package. I found this very interesting and began to attempt to develop a very simple Android application to connect to a simple MySQL database. I am beginning to think that I will concentrate on Android and PHP and not bother trying to learn Ruby, as I think that I will be able to achieve my goals without it.

Having submitted the requirements document I now had to begin my analysis and design specification document. I found this difficult to be honest as I was not really sure what needed to go into the document. I am not satisfied with my submission and hopefully will be able to speak with Padraig about it on Wednesday.

My Reflection

Getting the requirements document submitted was a relief, as was finding out that it’s possible to alter the document between now and the final submission date. I found Eamon’s class about database connection very helpful and it will definitely be of use in my project.

My original proposal was that my application would implement the Google maps API to calculate the user’s mileage based on their geo-location. I am worried now as I am not sure how this can be implemented into the application so that it is practical. I am thinking that perhaps I will have a feature on the application which will allow the user to calculate mileage if he wishes to, otherwise he should be allowed to simply input the mileage he has calculated himself.

Intended Changes

Next month, I will try to develop a simple database in phpMyAdmin.

I realised that I need to get cracking on the coding aspect of the project, most of the documents have been done at this stage so I need to focus on the practical side of the application.

Supervisor Meetings

Date of Meeting: 02/12/2015

Items discussed: PHP and database connections

Action Items: Develop database

Reflective Journal

Student name: Antoin Judge

Programme: BSc in Computing

Month: December 2015

My Achievements

I began this month by creating a simple database using phpMyAdmin, which I hosted on my own laptop using apache server. The database was very basic, consisting of one table with three fields, username, password and email. I then created PHP scripts to add records to the database. I also created a basic web page which called these scripts to add records to the database and to display them. This was very simple to implement on my laptop, so I then set out to do the same on an android application. I created a basic android app using Android Studio and attempted to call the PHP scripts on my server from the app. I had great difficulty doing this and spent at least a week trying to alter my code to connect properly. I had another meeting with Padraig. I didn’t really have much to show him with regard to my progress but we had a discussion about how the project was coming along and he gave me some advice regarding my database connection problems.

I continued to research methods to connect an Android device to a localhost server and found that it was not as simple as I had envisioned. I decided to try to host my application on a web hosting provider and found one called Hostinger, which also allowed me to use phpMyAdmin to create my database. I created a domain and placed all of my PHP and HTML files on it and created a new database. I was thrilled when I was able to update my database from my own android device for the first time.

It’s now Christmas and I have an exam coming up in January. I will continue to try to develop my project further but I think that most of my free time will be spent trying to get some study done for my exam.

My Reflection

When I finally managed to connect my Android application to my server side database it was a massive relief. I think that I can now build on this and attempt to have a basic prototype ready in the January for the midpoint presentation which is coming up in February.

Intended Changes

Next month, I will try to further develop my database to store all of the data required and make sure that it is relational. I need to implement the Google maps API in my app to calculate distance between two locations.

I realise that I am not as far along as I had intended at this stage and know that I need to really put my head down after Christmas and concentrate on my project.

Supervisor Meetings

Date of Meeting: 16/12/2015

Items discussed: Connection between android application and server side database. Possible web hosting providers.

Action Items: Start with a basic app and database and make sure that they can communicate with each other and build from there.

Reflective Journal

Student name: Antoin Judge

Programme: BSc in Computing

Month: January 2016

My Achievements

I continued to develop my application on Hostinger, I implemented web user interface for the administrator on the server using Bootstrap and jQuery. This interface allows the administrator to add employees to the database and also to view a table of all employees, and a table of all app users. I further developed the application so that when a user wants to register an account, he must provide a valid Employee number. This allows him to log in to the app successfully and adds him to the user table of the database.

The rest of the month was spent preparing for the mid-point presentation which we have on Saturday.

My Reflection

I feel I am progressing with the project, however a little slower than I anticipated.

I am concerned about the security aspect of my application. I managed to encrypt the user’s passwords on the database on my own machine, but have been unsuccessful implementing this on the one on hostinger.

Intended Changes

Normalise all of the data which I need to store in the database and redesign it as such.

I realised that I need to seriously consider the security aspect of hosting my database on a site like hostinger and look at the alternatives.

Supervisor Meetings

Date of Meeting: 04/02/2016

Items discussed: Mid-point presentation, database normalisation

Action Items: Normalise data to be stored.

Reflective Journal

Student name: Antoin Judge

Programme: BSc in Computing

Month: February 2016

My Achievements

The mid-point presentation went pretty well, I was not looking forward to it at all, as this was the first presentation that I had done on my own. I prepared well for it and when it came to the day I wasn't half as nervous as I had anticipated. I knew exactly what I was going to say and didn't resort to simply reading from the slides too much. Padraig and Eugene gave fairly positive feedback and also pointed out things that I needed to focus on if I am to complete the project successfully.

My Reflection

Since the mid-point presentation I'm afraid that I've neglected the project slightly. I have a number of CA's coming up and two other projects to think about also. I am beginning to realise that I really need to put my head down and work extremely hard for the next couple of months, which is difficult as I'm very limited with the free time available to me.

Intended Changes

I still have not got around to my database normalisation which is the main thing I had intended to have completed before the next reflective journal entry. I will be concentrating on this for the next week or so.

So far my mobile application is extremely basic, so I will try to develop it more and add features to it, as well as make it more attractive looking.

Supervisor Meetings

I did not have a meeting with Padraig since the presentation as it was not possible due to holidays and not being able to make it up to the college on particular evenings. I will arrange a meeting as soon as possible.

Reflective Journal

Student name: Antoin Judge

Programme: BSc in Computing

Month: March 2016

My Achievements

I have added functionality to the app which allows the user to get his current longitude and latitude coordinates. The app can now calculate the distance travelled by the user when requested. This is done by making a call with the users start and finish coordinates to the Google Distance Matrix API, which in turn returns a JSON Object with the time and distance between the two locations.

I had three CA’s which needed to be submitted this month as well as another project in Distributed Systems. This meant that I was under a lot of pressure between work and college assignments. I have all of my CA’s or the other modules completed now, and I am focusing on next week’s final exams.

My Reflection

This semester is definitely proving to be the most intensive one since I began the course. I have been under a lot of pressure trying to complete other projects and CA’s, and as a result of this, my time spent working on the main project has been limited.

Intended Changes

This is the final hurdle; I need to get the project finished as soon as possible!

Supervisor Meetings

I haven’t been able to arrange a meeting with Padraig this month as I simply did not have any free time to do so. I have been in touch with him via email and he has been understanding of this, time is a precious commodity at this stage of the semester and I need to make the most of what little time I get to spend on my project.