kd016499- Individual Project Report

by Calum Mcgloin

Submission date: 02-May-2017 10:38AM (UTC+0100)
Submission ID: 71622662
File name: kd016499_Project_Report.pdf (2.09M)
Word count: 22084
Character count: 116594

Department of Computer Science

Wi-Fi Based Indoor Positioning System for the Polly Vacher Building

Student: Calum McGloin

Student Number: 23016499

Supervisor: Dr Giuseppe Di Fatta

Module: CS3IP16 – Individual Project

Date of Submission: 2nd May 2017

Abstract – *Indoor positioning systems aim to provide localisation capabilities for the indoor environment and deliver the many benefits that arise from this knowledge. The use of Wi-Fi signal fingerprinting is a popular method used to realise these capabilities, this project has explored this approach and the data mining technique of classification to solve the problem of accurate indoor positioning. This report will detail the research undertaken to better understand the localisation problem, it will discuss the potential solutions that could be applied and will detail why the approach of Nearest Neighbour classification was selected for this implementation. Further to that it will discuss how the chosen classifier was implemented as part of a Wi-Fi fingerprinting system as well as the supporting components necessary, including an Android application as an interface for the user, a server side program to perform the classification and a database to store the signal fingerprints of the location. This report will also demonstrate the success of this implementation in achieving a highly accurate positioning capability with an overall accuracy of 95% when tested in the Polly Vacher building. Finally possible future work will be discussed, addressing not only how the successes can be built on but also how the limitations of this project might be addressed.*

Acknowledgements

I would like to express my gratitude to Dr. Giuseppe Di Fatta for his support and guidance throughout this project.

Contents

Glossary of terms and Abbreviations

IPS – Indoor Positioning System

GPS – Global Positioning System

AP – Access point. A Wi-Fi signal transmitter.

MAC address – Media Access Control Address. Unique identifier for network interface devices.

RSSI – Received Signal Strength Indicator.

WPS- Wi-Fi Positioning System. Name of Android application created.

KDD- Knowledge discovery in database

NN – Nearest Neighbour

SVM – Support Vector Machine

PaaS – Platform-as-a-Service. Cloud computing model that allows the development and management of applications over the internet.

DaaS – Database as a Service.

JSON- JavaScript Object Notation

1. Introduction

For decades the Global Positioning System (GPS) has been providing the framework for a range of localisation services in the outdoor environment, today it is deployed in standalone systems as well as integrated as one of many features in smart devices [1]. A significant number of applications used by smartphone users on a daily basis rely on location information to deliver core functionalities of their applications. While the popularity of GPS is evident it does suffers from an important limitation, which is it does not work accurately in the indoor or sheltered environment. This limitation and the numerous benefits of localisation have given rise to a new market of Indoor Positioning Systems (IPS).

The aim of this project was to better understand IPS and implement such a system to be tested in the Polly Vacher building at the University of Reading. The project developed was to provide similar capabilities as GPS by developing an Android application as the interface for the user and by using existing Wi-Fi infrastructure in the building. The project would apply data mining techniques to develop and implement a solution as an accurate alternative to GPS indoors.

The personal motivation to pursue this project resulted from the wide range of applications and benefits that arise from an IPS, as well as the opportunity to further develop Android mobile development skills from second year studies and to explore the field of data mining. An IPS can provide many of the same benefits as GPS including as a navigation tool in large buildings such as shopping centres, warehouses and hospitals [2]. It can also be used as a life-saving tool in emergency relief, the location tracking abilities might be used to alert emergency services to the specific location of a situation, and similarly it may be used in health and social care to track vulnerable patients such as dementia sufferers. Enhancing on GPS it might be used in business as a tool to collect information to be used in knowledge discovery, for example allowing for better understanding of customers and direct marketing.

This report will discuss the processes undertaken to develop this system, firstly it will detail the research conducted to further the understanding of the approaches and techniques that were to be applied, it will discuss the specific approach decided upon based on this research as well as how it was implemented. The testing of the system implemented will be outlined and the results of which will be used to examine the successes and shortcomings of this project, reflecting also on the personal development and learning experience of the developer. Finally a brief overview will recount the outcomes and will address future work that might be pursued to enhance this project.

2. Problem Articulation and Objectives

The problem this project is addressing is the localisation of a Wi-Fi enabled device in an indoor environment. This overcomes the important limitation of GPS that the satellite based system is unable to provide accurate location prediction in large buildings, this is because to do so they require clear unobstructed line of sight to at least three satellites to then perform trilateration to pinpoint the devices location [1]. In the indoor environment the receiver would not have direct line of sight, the signals that are transmitted from the satellites will have to pass through the building structure which will attenuate them [3] making it harder for the receiver to get a fix.

This project has endeavoured to implement an accurate Wi-Fi based system to be tested in the Polly Vacher building, making use of resources that are freely available to potential users. These resources are the extensive Wi-Fi infrastructure which is fixed in place at the university and a purposely built Android application. Addressing this problem using Wi-Fi signal data to build a fingerprint of the environment gave rise to a number of problems that were addressed, this includes how the Wi-Fi data can be collected and stored, how data mining techniques can be applied to generate predictions from the data and finally how a user might interact with the system.

A number of stakeholders were identified and considered during the development of this project. The first stakeholder is the developer, myself, responsible for the evolution of this project throughout the development lifecycle. Secondly the project supervisor, Dr Giuseppe Di Fatta, who provided assistance to the developer in terms of domain knowledge and project management. The final stakeholder is the end user, this is anyone that owns an Android device and requires indoor localisation capabilities.

The potential solutions to be assessed will be required to fulfil the following technical specifications:

- The system will provide user location predictions within an Indoor environment
- The system should provide accurate location predictions based on Wi-Fi fingerprinting
- A simple interface should be created in the form of an Android application for the user to interact with the system
- The user should be able to request a location prediction
- The user should be able to update a location fingerprint
- Instructions should be provided to a user on how to update a fingerprint
- The room prediction should be displayed to the user
- Results returned should illustrate location within Polly Vacher building
- Solution coverage should be scalable (ie can add more rooms as required)
- Predictions shall be restricted to rooms, not corridors/open spaces

With this understanding of the desired system in place the following criteria will be used to establish the acceptability and validity of the solution implementation:

- A lightweight Android app will be created as the interface to the system
- The App should support the two phases of fingerprinting
    - Training
    - Positioning
- Training Data accumulated & stored for predictive algorithm use
- Resolution: Correctly identify location to resolution of typical room sizes (Polly Vacher)
- Accuracy: Correctly predict room at least 90% of the time
- Coverage: Application works in representative sample of rooms in Polly Vacher
- Execution: Application will return a result within acceptable time (<10 secs)

In summary, a number of objectives were identified for this project, primarily that an accurate Wi-Fi based positioning system should be developed. This system should employ signal strength fingerprinting of the building and should use a classification model to predict the user's location. Secondly an Android application would be created to serve as an interface for the user. The app should be simple and easy to understand and it should facilitate the core functionalities of the system, inputting data during fingerprinting and displaying the prediction results. A backend should be created for the application to handle the majority of the computation, this refers to the program that will implement the classification model and the data store that will be a collection of the readings that make up the fingerprints. This will ease the demand on the user's smart Android device. Furthermore, various classification algorithms will be evaluated and the most suitable for the exercise will be determined based on prediction accuracy.

3. Literature Review

In the early stages of this project thorough research of existing literature was conducted in order to better understand the topic and the approaches that might be implemented, this continued throughout the development process. The areas of focus will be discussed in this section.

3.1. Approaches to Wi-Fi based IPS

Wi-Fi based IPS is an area that has been explored considerably over more recent years and several approaches have been established. A paper from the Technical University of Berlin [4] highlighted three categories of approaches, trilateration, triangulation and pattern matching.

The first approach of note uses trilateration and received signal strength indicators (RSSI) to provide an estimation of the user's location; RSSI is a measure of the amount of power in a received signal from a wireless access point (AP) [5]. The paper by Robin Henniges [4] highlights that this approach requires the receiving device to be in range of at least three APs, it then uses the RSSI values to calculate the distance from the respective AP. The distance is then used as a radius to form an area that contains the position of the receiver, the point at which the three areas intersect is the predicted location. This has a similar limitation to GPS in that to get a good level of accuracy a clear line of sight between the receiver and transmitters is needed, thus for use in the test environment it is not suitable as the signal would be liable to attenuation.

Angle of Arrival is an approach that uses triangulation to predict the user's location, it is described in a guide published by Cisco [6]. This approach requires that the receiver be in range of at least two APs with each being in a known position, the angle the signal is received at for each and the line of bearing are used to determine an intersection point which is given as the predicted location. This approach is more suitable for indoors as signal attenuation would not impact the outcome, with that said it isn't suitable for this project as it requires modified equipment to determine the angles.

The final approach of fingerprinting was the focus of this project as per the project description, this is a pattern matching approach that applies data mining techniques [4]. It requires information to identify the individual APs and the respective RSSI for each. A report [7] regarding this approach identified that it comprises of two phases, the first of which is the training phase during which Wi-Fi data is collected for each location resulting in a signal map of the environment. The second phase is the positioning phase, this involves comparing a live test reading against the stored location fingerprints using a distance function which is used as part of a classification algorithm to predict the location. Fingerprinting takes advantage of existing infrastructure and is less susceptible to signal attenuation caused by the building structure and can employ techniques to combat this such as multiple fingerprints of one location taken at various times, making it the ideal choice for the Polly Vacher building.

3.2. Classification for Fingerprinting

As the fingerprinting approach was the focus of this project, it was necessary to research the data mining technique of classification to understand how the positioning phase would be implemented. Using lecture notes from the Data Mining module at the University of Reading [8], it was found that the fingerprinting task is an example of instance based classification, this involves comparing an unknown instance to a training dataset, using a distance function to perform this comparison.

Two common instance based algorithms were identified in [8] and explored in the papers [9], [10], these are nearest neighbour (NN) and k-nearest neighbour (k-NN) classification. NN uses a distance value computed against each stored fingerprint to select the one that is the closest match to the unknown instance. K-NN works in the same way, the only difference is that it finds the k lowest distances and uses a majority vote to make the prediction. Li et al. [9] analysed the performance of nearest neighbour

models for indoor localisation, through physical tests they found that as the value of *k* increased so did the number of classification errors suggesting that single NN was more suitable for this task. They also considered decision tree classification which uses the training data to form a tree structure that the test data traverses in order to be classified [11]. The root node is the first node in the tree, it is where the first condition is applied. This will then have child nodes which can either be another condition node or a leaf node that assigns a class to the record. When Li et al. compared k-NN and the decision tree, the latter had the benefit of faster processing but did not have as high levels of accuracy as the former. Support Vector Machines (SVM) were also evaluated in the paper. SVMs classify a data point by using the training data with known class labels to find the best division in a multidimensional space, between records of one class and the others [12]. Li et al. deemed these to be unsuitable for the problem due to issues of high dimensionality of the dataset.

Moghtadaiee and Dempster [10], considered the critical distance function component to the NN and k-NN algorithms. A range of distance functions were examined and considered the probability of error (POE) and mean distance of error (MDE). The functions of interest identified in the paper are variations of the Minkowski distance, namely Euclidean distance and Manhattan Distance. The paper finds that when NN is implemented the lowest POE and MDE come from using Euclidean distance, however when using implementing a version where *k* is greater than one it is Manhattan that provides the best results. These models and measures will be explored further throughout this report.

3.3. Android

By using a smart device as the receiver and interface for the user, it was necessary to understand the operating system the application would be developed for, the focus of this research was on the Wi-Fi data gathering capabilities of Android. As part of the research into IPS, a guide [13] published by Insoft Software highlighted the platform suitability of different approaches with Wi-Fi based systems declared suitable for Android devices.

Further to this, research was conducted into the Android developer documentation to understand what support is available in the Java based platform. It was found that a set of classes exist which can facilitate the collection and analysis of Wi-Fi data collected from a network the device is in range of [14]. These included the 'WifiManager' [15] class, which serves as an API for Wi-Fi based operations and the 'ScanResult' [16] class, which can be used to retrieve specific information about a scan, both of which would be essential to the application.

Other tasks that would have to be performed by the application were also researched such as the network communication with a backend [17], it was identified that a number of external libraries are supported and the use of is encouraged which would benefit the implementation process. This reinforced the advantage that the open source nature of Android and the large developer community has in providing a developer friendly platform.

3.4. Backend Technologies

To process and store the Wi-Fi data, a number of backend components would be needed. With no prior experience in this area, it was necessary research these various components that would be required to support the application.

A database would be needed to store the training dataset used to perform classification, it was identified that this application could use a NoSQL database rather than a traditional relational database. This might add a benefit as the data that will be stored will vary in size and requires greater flexibility. An article by ThoughtWorks [18] highlights the various types of NoSQL databases that exist and provides guidelines as to when each would be used. The types mentioned in [18] that were considered for this project were key-value databases which map the value attributes to an identification attribute, and document databases, that store key-value pairs within a document using a particular format.

To transfer data between the application and the database as well as applying the application logic, it was determined that this system might benefit from an additional architecture layer between them [19]. This would be a program that would interact with the Android application using HTTP methods, acting as an intermediary between the app and database by adjusting the data to the required format before storing. The classification algorithm can also be applied as part of this program, relieving the demand placed on the user's device saving battery power and reducing the amount of data transferred.

4. Solution Approach

This section will outline the solution options that were identified through the research discussed in the literature review, the proposed solution for each area will be presented with justifications for the decision. Finally the solution definition will summarise the solution chosen.

4.1. Training phase

There were a few options of how the map of fingerprints would be generated, these are concerned with what data is required and how should that data be collected. First of all was the decision of what information would be included in the database, there were a number of fields that might be considered including the MAC address which acts as a unique identifier for the AP, the RSSI from a particular AP, the room number of where the set of readings were taken, a timestamp to track the age of the record and an individual recording point identification number. The first three are considered essential for the fingerprinting approach and would be included. The timestamp field is an option used in a paper [20] by Luo et al. to select the stored readings to be used in the positioning calculations, this was ultimately decided against for this implementation as initially the age of the stored readings was not considered necessary, however as will be discussed later it might provide a way to combat difficulties of dynamic environments. The individual recording point's field was not included as it was felt to be unnecessary, this decision was based on the location prediction requirements outlined in the project objectives outlined in section 2 which requires prediction at room level only, again this will be discussed as future work later in the report.

Once the data that was to be stored had been decided, attention was turned to how the data would be stored, the key decisions involved how many readings points should be used in each room. The decision was made to select the number of reading points in proportion on the size of the room, this is to account for differences in RSSI and visible APs throughout the same room. The number of readings to be taken at each recording point was also considered, the approach selected was to take four readings each facing in different orientations. The reasoning behind this was to account for the effect of the user's body on the RSSI, it was observed during early tests that the RSSI was likely to change with a change of direction.

4.2. Positioning Phase - Classifiers

The decisions made regarding the techniques implemented for the positioning phase were crucial in determining the accuracy of the system, they involved the classification model to be implemented as well as the distance measures that might be used.

As mentioned in section 3.2 the most common options for classifiers to be used in fingerprinting are the nearest neighbour classifiers. Both NN and k-NN are simple algorithms to implement which coupled with the proven results of previous research [9], [10] make them a desirable choice for this project, as such they were the primary classifiers tested for this project. They also benefit from the choice of proximity measures that can be used, variations can be explored as in [10] to find a more accurate solution, and this will be discussed shortly. Both approaches provide a thorough classification process by comparing the test data against the large training dataset, this exhaustive approach does mean that it is not a scalable model and would not be suited to larger projects. NN does have a downside in that by only selecting the single record with the lowest distance it can be sensitive to noise caused by the dynamic environment, for instance if there was a new obstacle between the transmitter and receiver the RSSI is likely to change. K-NN also has a significant downside, as mentioned in the work done by Li et al. [9] which suggests that as k increases the accuracy of the predictions decrease meaning it is less likely to meet the primary objective of this project. Decision trees and SVMs were also considered and tested as will be discussed shortly.

With an established training dataset it was possible to explore these models and others using the data analytics platform KNIME, this is a tool that can be used to facilitate each stage of the knowledge discovery (KDD) process. The purpose of its use in this case was to compare the performance of the algorithms and distance functions highlighted in section 3.2 when applied to the training data collected for this project in the Polly Vacher building. The KNIME workflow generated for this purpose can be seen in figure 1.

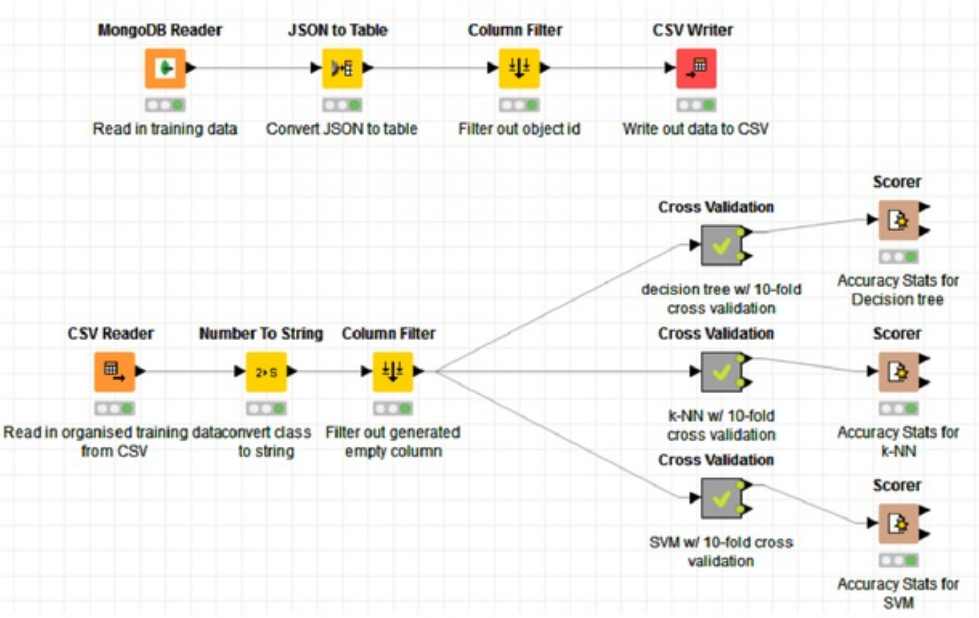

Figure 1- KNIME workflow for classifier and distance function tests

The first step of the KDD process, data gathering, can be seen in the workflow by the 'MongoDB Reader' node which is responsible for reading the training data from the data store which will be discussed later in this report. After this several nodes start to apply the data pre-processing phase in the workflow, these handle the conversion from the JSON format to a table in KNIME, the filtering out of the unneeded object id column and finally restructuring the table. The final action is not represented in the workflow by a single node, rather it is seen in part by the 'CSV writer' and 'CSV reader nodes'. The writer generates a comma separated value (CSV) file which is then passed into a separate Java program which performs the restructuring. Before being passed through the program, each reading element in the table was a string consisting of a MAC address and RSSI value. The output of the program, retrieved using the 'CSV reader node', was a table which represented each MAC address in its own column and the cells simply contained the RSSI of that AP for each training record, if the AP was not present in the reading the corresponding cell was assigned the arbitrarily small value -120. The final preparation steps are converting the class identifier to a nominal value so it could be used in the classifiers and lastly filtering out a generated empty column that was not needed.

It can be seen in the workflow that three classifier nodes were used for comparison, each was applied using 10-fold cross validation [21]. This method is used to provide accurate an estimation of errors that might occur when the model is applied to the test data, in relation to this project it calculates the generalisation of error when the unknown location reading is applied to the model. 10-fold cross

validation involves dividing the dataset into ten subsets by applying the holdout method ten times, this is when the dataset is partitioned into two smaller sets given the titles training and test data [22]. Each fold represents a different instance of the holdout method, in each fold the training subset is comprised of nine tenths of the overall dataset the test is conducted using the remaining tenth. For each of the ten tests performed a different subset is used as the test set, as a result over the ten tests the entire dataset is used for training and testing which provides a better estimation of errors.

The workflow shows that the three classifiers used were a decision tree, k-NN and support vector machine, all of which have been described in the literature review. The results of these simulations are then output using the 'scorer node'. The first simulation to consider was testing the impacts of using different values for *k* in k-NN, this was identified by setting the value used in the k-NN node using the values of one through to twelve and running the model once of each *k*. The line chart in figure 2 shows the relationship between the value of *k* and the accuracy of the predictions, it should be noted that Euclidean distance was used as the distance measure for all of these tests.

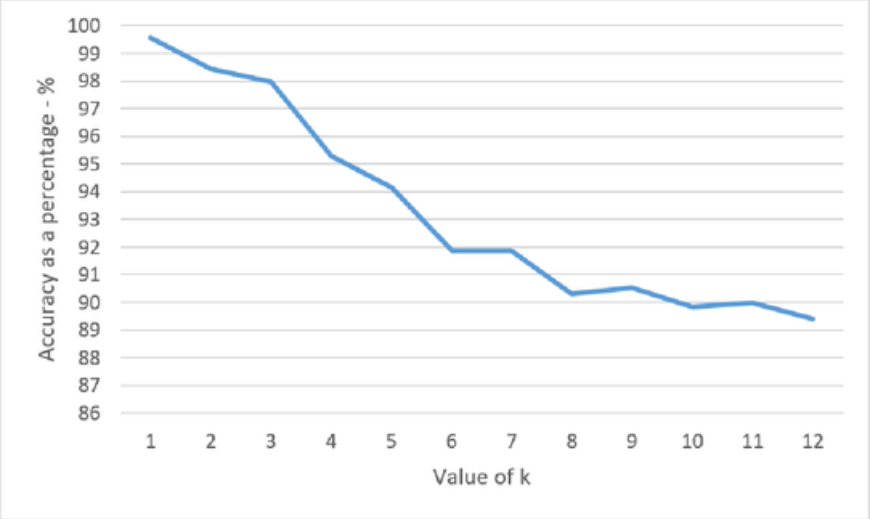

Figure 2- Line chart showing the relationship between the value of k and the accuracy of the k-NN model

The outcomes displayed in figure 2 corroborate the findings of Li et al. [9] that as the value of *k* increases the accuracy of the model is reduced, this is especially evident between the values of one and six which can be seen to steadily decrease. After which a couple of instances can be seen where the accuracy of the model using *k* is the same or even slightly better than the previous, seen between six and seven, eight and nine as well as ten and eleven. However the decrease in accuracy does continue albeit at a slower rate. These tests suggest that NN would be the better algorithm for this project than any k-NN with *k* greater than one.

Based on these findings NN was compared in KNIME against two other classification models as mentioned previously, decision trees and SVMs. The outcome of these tests can be seen in figure 3, it shows that the decision tree performed the worst of the three when considering the accuracy of the classifications. Significantly higher with little separating them, both NN and SVM performed better with NN slightly more accurate, the performance of SVM conflicts with the findings of the previous studies [9] which suggested it was not a suitable for Wi-Fi data.

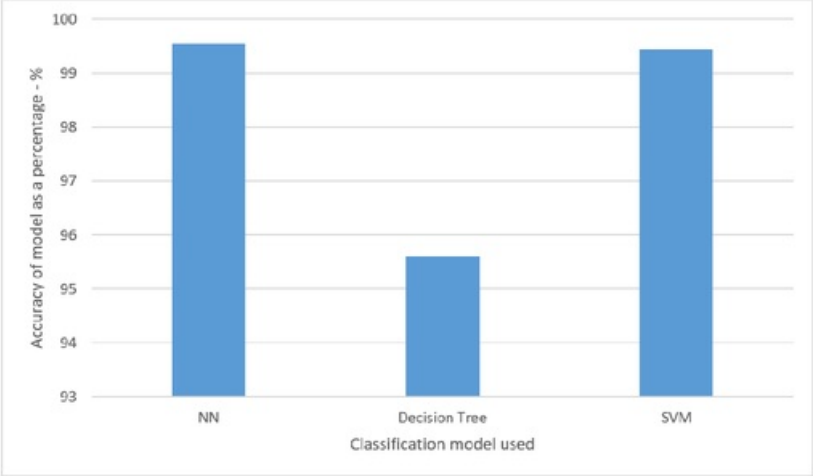

Figure 3- Bar chart showing the accuracy results of classification models

KNIME was also used to compare the effects of various proximity measures on the accuracy of the NN model, the proximity measures compared were the cosine similarity and the Minkowski variations of Euclidean and Manhattan distance. Euclidean distance calculates the square root of the squared sum of differences between the RSSI of the unknown instance vector and the current stored instance vector [23]. Manhattan distance on the other hand calculates the sum of the absolute differences as the distance value, both measures consider the lowest distance value to be the best [24]. The cosine measure, which calculates the cosine angle between two vectors, is different from the other two in that it is a measure of similarity rather than distance [25]. The closer the two readings are, the closer to one the cosine angle will be and thus for NN, a similarity of one would indicate that should be the class prediction made.

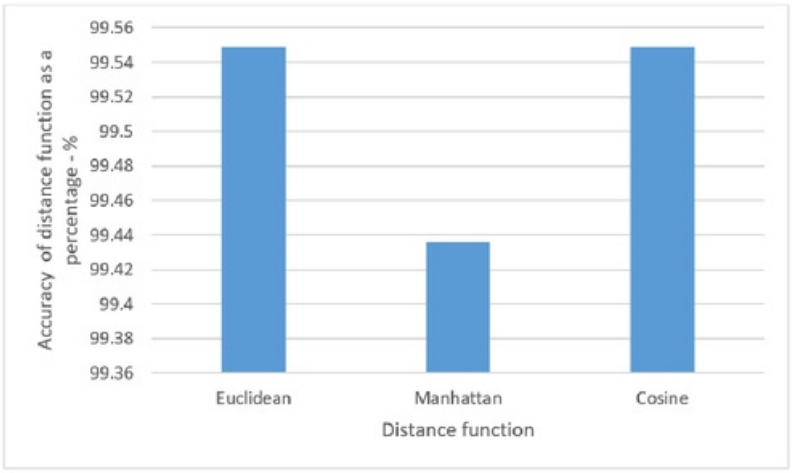

Figure 4- Bar chart showing the accuracy of the NN model when using different distance functions

The results seen in figure 4 show that Euclidean distance and Cosine similarity have the same level of accuracy when used in NN, they both perform better than Manhattan however the difference is small at just over one-tenth of a percent.

Considering the results of previous studies and the testing done with the training set of this project, NN was chosen as the classifier to be implemented for its high accuracy and greater implementation simplicity when compared to decision trees and SVMs. Following this decision, it was necessary to

choose the proximity measure that would be used. Based on the proven results of Euclidean distance with NN and the findings of the KNIME testing, Euclidean was chosen as the measure for this project. While either of the proximity measures may have been used, the accuracy and ease of implementation of Euclidean provided an advantage over the others.

4.3. Backend Technologies

In order to facilitate the chosen solutions mentioned above and meet the project objectives, a number of backend technologies must be chosen.

As discussed in section 3.4, a NoSQL database was considered a suitable solution and provided a learning opportunity of a new technology. For this MongoDB [26] was chosen, the reason for this decision was the support for the JSON data format and the Java driver which both provide greater integration with other Java based components. The Java driver would be used to perform CRUD [27] operations on the database, including submitting new fingerprints and retrieving the dataset during the positioning phase. Another benefit of the drivers is familiarity of the developer with the Java programming language. JSON support aids the integration of the database with the Android application as it also has strong support in Android through external libraries [28], this is beneficial as it was the desired data transfer format. Another deciding factor in the decision to use MongoDB is the ability to do so via the database-as-a-service (DaaS) provider mLab, the ability to access the database through a web browser makes it easier to manage. Using mLab also provided scalability for this data storage implementation, if it were to be applied to a larger environment the data store would be able to handle significantly more data as well as concurrent users accessing the data.

The other backend component that was required was a server side program to serve as an intermediary between the Android application and MongoDB and to handle the classification processing. Based on familiarity with the language Java, servlets were chosen for this implementation. The decision to implement a solely Java program for this rather than using a framework was made because the server-side program would be simple and only perform a few operations.

4.4. Solution Definition

In summary, the chosen solution outlined above will implement a fingerprint based IPS which will use the NN classification algorithm along with Euclidean distance to make a location prediction based off of a reading which consists of a set of MAC addresses and their respective RSSI value. The training data that the unclassified record will be compared against will consist of a room number as well as a set of MAC addresses and RSSI values. This dataset will be generated following a strategy of collecting readings from multiple recording points evenly distributed throughout the room, with the number of recording points determined by the size of the room. At each recording point four readings will be taken, each facing a different direction. These readings will be collected using the Android application created, they will be passed to a Java application which will forward the data to a MongoDB database on mLab. The Java application will also be used to implement the positioning algorithm, receiving the live unseen reading from the application and retrieving the training data for the database, finally outputting the room number prediction to the user's Android device.

The solution described above may be adequately verified once all of the components have been implemented and it meets the primary objective stated in section 2, which is the system can accurately predict the user's location.

5. Design and Implementation

5.1. Design

This section of the report will detail the design processes undertaken and the outputs produced from them, it will explain the design of the overall system, the Android application and finally the critical classification algorithm.

5.1.1. System Context

As this system followed the fingerprinting approach to Wi-Fi Positioning the design considered two processes, one representing each phase of fingerprinting. These were visualised through the use of a use case diagram that can be seen in figure 5.

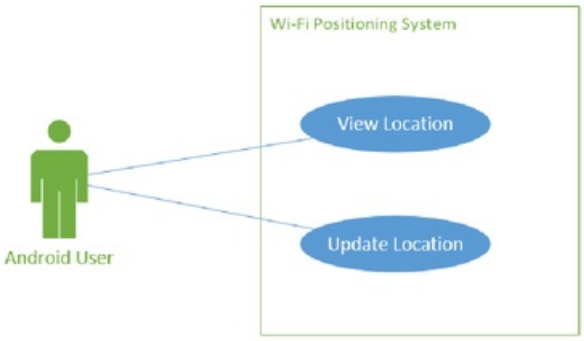

Figure 5- Use case diagram highlighting functions of IPS

This diagram helps to understand the context of the system in terms of what functionality it provides and to what users. It can be seen that the two functions this system will deliver to the user will be updating the training dataset and viewing the location, both of which will be available to all users of the system. The users are represented in the diagram by a generalised object which abstracts the possible users to be anyone in the Polly Vacher building who has an Android device, this is a fair reflection of the system as all users of the application will have the same operations available to them whether a University member or visitor.

Following this it was then necessary to take a deeper look at both of the functions the system will perform, this was achieved through the formation of activity diagrams the first of which can be seen in figure 6.

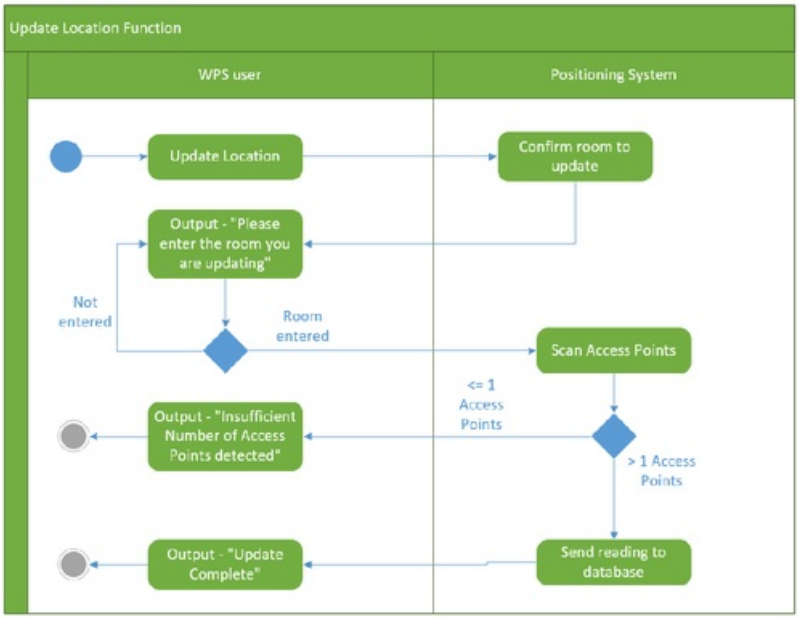

Figure 6- Activity diagram showing processes of Training phase

This demonstrates the flow of activities that occur when updating the location readings. This design focuses on creating a single reading and does not consider more specific details that have been implemented such as a button click and each of the system components, however it does still provide an accurate high-level representation of the activities involved and demonstrates the interaction of the system and the user. It can be seen that the first step is the user initiates the Update Location function, the system will then ask them to confirm the room the wish to update by outputting the question to the user, if the user does not enter a room number the system will prompt the user again. When a room number is entered the system will initiate a scan of the APs it is in range of, it will perform a check of the number identified and if it finds fewer than two it will tell the user there is an insufficient number of APs detected. If the check is successful and more than one is found the readings will be added to the training dataset and the update will be confirmed to the user.

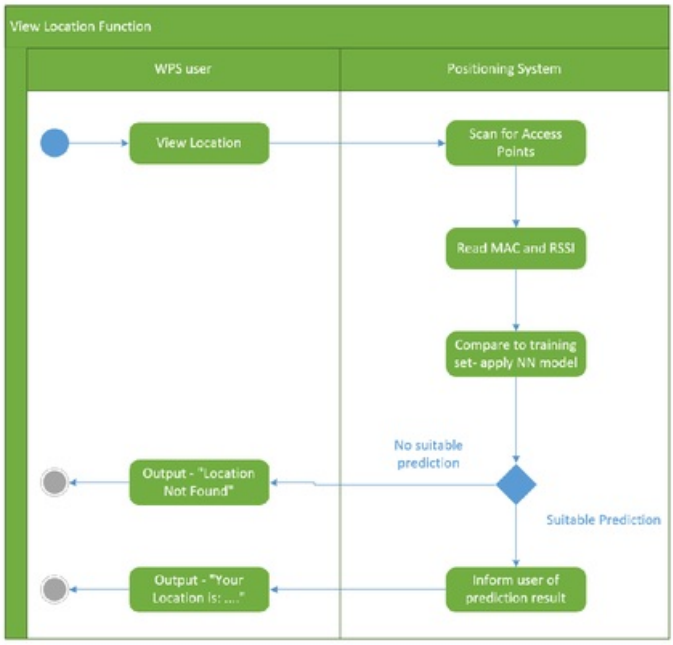

Figure 7- Activity diagram showing processes of positioning phase

The second activity diagram can be seen in figure 7, this represents the activities involved in the positioning phase. The user initiates this phase through the View Location function, similarly to the previous function this will then perform a scan of the APs in range retrieving a collection of Wi-Fi data. The MAC addresses and signal strength is read for each AP, this is the test record which is then compared to the fingerprint of the environment by applying the NN classifier. If a suitable prediction is returned the system will inform the user of the result, if not it will tell them the localisation was unsuccessful. The suitability of the prediction will be determined by lowest distance computed, if it is below a certain threshold it will be considered a sufficient match and will be used as the prediction.

5.1.2. System Architecture

With a clear understanding of each of the processes consideration was given to the various components and architecture required to facilitate them. Figure 8 shows a high level diagram demonstrating a three-tier architecture and the interaction between each tier in relation to the functions of the system, after the environment has been scanned using the Android application. The arrows in the diagram show the interaction between each subsystem and the transfer of data between them, the green arrows represent the positioning phase and the blue represent the training phase with the numbers showing the order of events.

For the positioning phase this diagram explains the stages of the 'Compare to training set' node seen the last activity model. The first step shows the transfer of the unknown location reading from the app to the View servlet, it is at this point in the system that the classification model is applied, this will be discussed later. In order to perform classification the training set is transferred from the database as seen by 2. After the completion of the classification, number 3 shows the transfer of the prediction to the Android application. The training phase indicators in this diagram visually explain the steps of the 'Send to database' node from figure 6. After the user has specified the location and the data has been collected it is sent from the app to the *Update* servlet, from here it will be forwarded to the database.

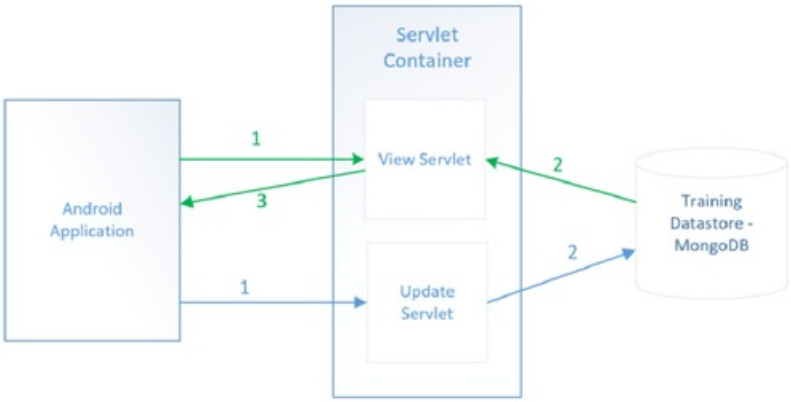

Figure 8- High Level diagram of 3-tier architecture implemented

5.1.3. Android UI Design

As the interface of the IPS, essential consideration had to be given to the applications user interface, as per the project objectives it should be easy to understand and clearly deliver the functions of the system. During the training phase it should enable the user to enter readings into the database in the simplest manner possible, and during positioning it should clearly display to the user the result of the localisation with the use of text telling the user where they are and a map to visualise there location relative to the whole building.

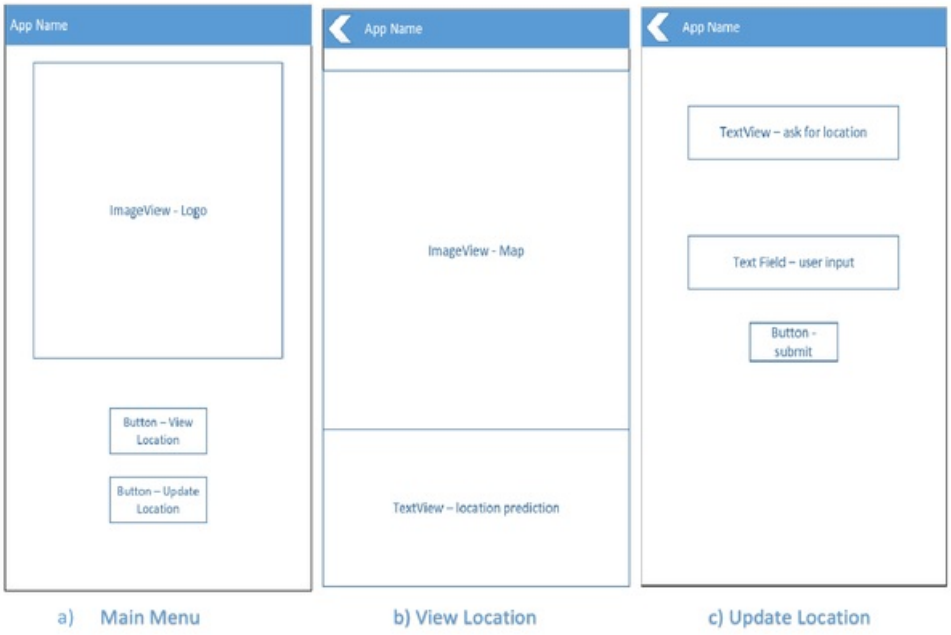

Figure 9- Wireframes produced for each screen of the application

Wireframe designs were created for the three screens this application would consist of, the main menu which should allow the user to select the function to deploy as well as the two phases highlighted above. These can be seen in figure 9. These demonstrate the visual components that would be used for each screen, considering the specific Android components that will be used and the functionality that they

deliver. The first wireframe, a, shows the design for the main menu, it consists of an ImageView which will display the logo and below there will be two buttons one to initiate the functions. The second wireframe, b, displays the design for the screen that will inform the user of their location, it consists of an ImageView to present the location map to the user and a TextView to explain what the map is showing. The final wireframe, c, shows the screen to be used during training, the TextView is used to ask for the user's location, the TextField facilitates the user input and the button will submit the reading to the database. It should also be noted in the last two, the action bar at the top contains a back button to help the user navigate back to the main menu.

The final designs implemented closely resemble these wireframes and can be seen in figure 10.

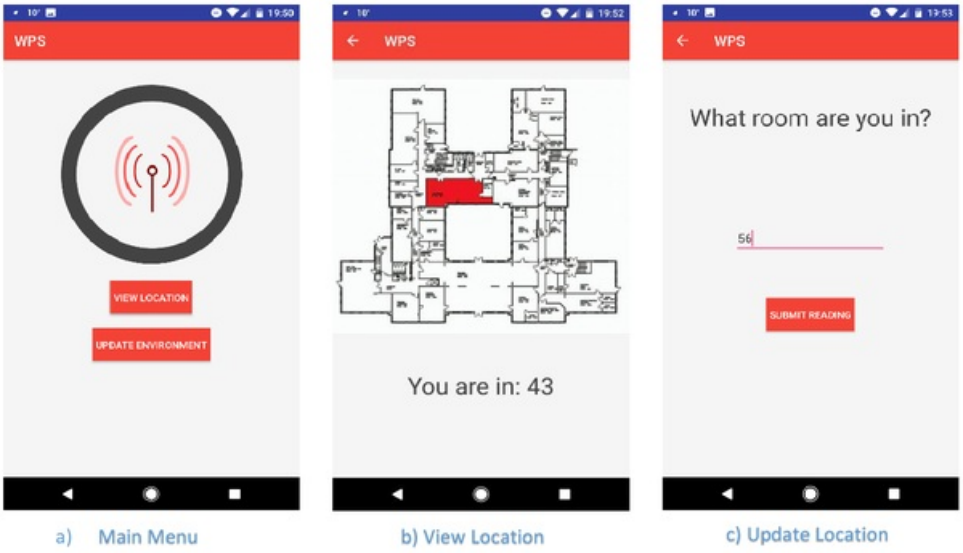

Figure 10- Screenshots of the UI implementation

5.1.4. Classification algorithm Design

At the core of this project is the classification algorithm, it is the most essential element of the adopted fingerprinting approach. As such it was necessary before implementation that a thorough design was created to establish how it would be applied to the problem given. Once an understanding of the classifier was established it was possible to apply it to this project. This was achieved through a program flow model, figure 11, which outlined each stage of the NN algorithm with Euclidean distance, the selection of which was discussed in the solution approach section of this report.

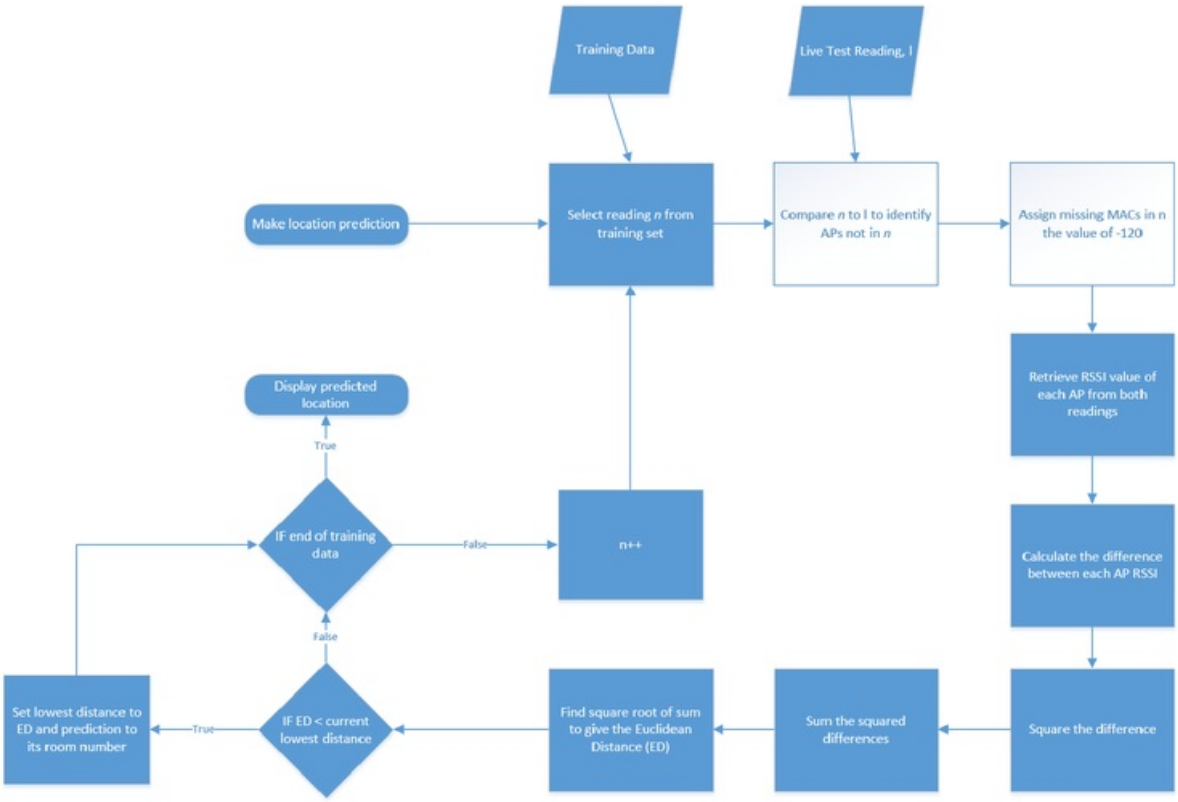

Figure 11- Flow diagram demonstrating the NN classification algorithm

It can be seen in the flow diagram that this NN model for Wi-Fi localisation takes the training data as input and will select one location reading to process at a time, it will also take the unknown live test reading as the second input. The white boxes highlight the data pre-processing steps that are undertaken to ensure the data it ready for to be apply the distance function, this involves cleaning the data by identifying APs that are present in the live reading but not in the current stored. Once the missing values have been identified they will be assigned the value of -120, the reasoning for this will be discussed shortly.

Following the completion of this cleaning, the pairs of RSSI values from each AP are retrieved and the Euclidean distance measure is applied. The first step the Euclidean distance calculation is to find the difference between the signal strength value pairs, then each of the differences are squared and the total sum of differences is calculated. The square root of the total squared differences is then computed to give the Euclidean distance between the two readings currently being compared. The final steps involve comparing the distance to the current lowest distance and assigning a new value as well as prediction if it is lower. As NN is an exhaustive algorithm this process is repeated for all of the fingerprint readings stored and once all of the distances have been computed the prediction is made.

Initially the intended data cleaning approach was to only use the APs common to both readings in the distance calculation. Early testing suggested this approach would work in a smaller environment with only a few APs however in a larger environment, such as the Polly Vacher building used to test this system, it would not. With a greater number of APs and a larger geographical space this approach had a negative effect on the distances computed, this is because it is very likely that two fingerprints either

side of the building would have very different APs. When the summation step of the Euclidean distance calculation was performed, the fingerprint of the room the user was in with a greater number of common APs would provide a greater number of inputs to summation and it was probable that it would return a greater distance than a room further away. The -120 approach seen in a 2015 article by P.Jiang et al. [29] was adopted instead only using the common APs to rectify this, the reason this was applied to this implementation is that no MAC addressed was observed to have an RSSI as low as -95 and by assigning missing APs a value lower than achievable the effect was an increase of the distance calculated.

5.2. Implementation

This section will discuss the implementation of each of the components of the system, it will focus on important or problematic sections of each and will endeavour to explain the rationale of the work done.

5.2.1 Android Application

As mentioned previously the Android application, WPS, would be the system interface for the users, as a result it is a critical component of this implementation. An advantage of using Android operating system to implement this interface, beyond what was discussed in section 4, is that it is one of the most popular mobile operating systems in the world meaning it would be available to a greater number of potential users. WPS is created using a minimum SDK version 10 so that it will run on 100% of Android devices in use.

The WPS application consists of three activities or screens, the first of which is the *Main Activity* which is responsible for loading the main menu (seen in figure 10) for the user to select the function they would like to use. The possible functions are represented by two buttons located underneath the app logo, these buttons are the *Update Environment* button and the *View Location* button, and they initiate the core fingerprinting processes of the training phase and positioning phase respectively.

Firstly the important aspects of *Update Environment* function will be discussed. When the user clicks this button the *UpdateLocation* activity will be started, as part of the training phase it is used as a tool to gather and store Wi-Fi data. When the screen opens a dialog box (figure 12) will appear explaining to the user how best to take readings, following this the user is asked to input the number of the room they are in and they can take readings by clicking the *Take Reading* button.

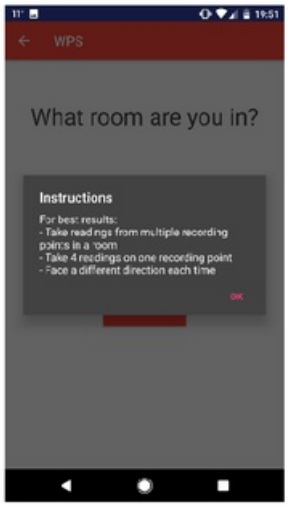

Figure 12- Screenshot showing the instruction dialog box

When the button is pressed the collection process is initiated within the application, it does so by calling an asynchronous class to perform a background scan of the surrounding APs. The background operation

starts by calling a method which uses an instance of the Android intent class and the *startActivityForResult* method to get the Wi-Fi data to be stored, this will launch the *ViewLocation* activity but does not launch its UI. The processes of this activity will be discussed shortly, for now the focus will remain on the updating functionality. The results from the scan performed in the other activity are returned in the form of an arraylist of strings, with each string containing a MAC address and it's RSSI. Once this data has been received an instance of the *Location* class is created, this comprises of two attributes which represent the room number and the list of readings retrieved.

The data format that was to be used for this program was an important decision to make, as mentioned in section 4.3 it was decided that JSON would be used as it could be easily applied throughout the system components. It was used in the *UpdateLocation* activity send the newly generated *Location* instance to the server to then be stored, this was achieved through the use of the GSON library which is described as "…a Java library that can be used to convert Java Objects into their JSON representation." [28]. The main advantages of employing GSON is that is saves time when converting objects to the data interchange format, this coupled with the simplicity of use in an Android application made it a valuable asset for this project. Figure 13 demonstrates how it was used to convert the *Location* objects to then be sent to the server.

```
private void saveLoc(){

    //create new location object
    location = new Location(locationID, result);

    //create instance of  gson
    Gson gson = new Gson();

    //uses gson to convert object to string and stores it in String json var
    json = gson.toJson(location);

    //sends initiates data transfer
    new sendToServer(json).execute("");
}
```

Figure 13- Code taken from WPS application showing use of Gson

Also seen in the figure on the last line, is a call to another asynchronous task which implements the transfer of the JSON string to the server and beyond. To communicate with the server application another external library is used, this is the OkHTTP library [30] which implements data transfer using HTTP messages. The reasoning for using this library is that it provides an efficient way to do so in terms of both development time and the actual transfer process. The API was used in a class called *PostServer*, a method was called from the background task and OkHTTP was used to create a HTTP POST request with the Wi-Fi data contained in the body of the request. This can be seen in figure 14. The request is sent to the *Update Servlet* which will be discussed later, WPS will then receive a response message and the WPS portion of the training is completed.

```
private String postFramework(String url, String json) throws IOException{
    //create body of request
    RequestBody body = RequestBody.create(JSON, json);
    //create request
    Request request = new Request.Builder()
            .url(url)
            .post(body)
            .build();
    //initialise response
    Response response = null;
    try {
        //send request messsage and assign returned value to response
        response = client.newCall(request).execute();
        //return the response as a string
        return response.body().string();
    } finally {
        if (response != null) {
            response.close();
        }
    }
}
```

Figure 14- Code showing how OkHTTP was implemented in WPS

Turning focus to the other activity of the WPS application, the *ViewLocation* activity which is initiated by the user pressing the *View Location* button in the main menu. This facilitates the first and last step of the positioning phase, firstly to collect the Wi-Fi data from the location that will be the subject of the prediction and lastly to output the prediction to the user.

In order to carry out the data collection task, used by both this and the *Update Environment* function, the WifiManager API provided in Android was used. On top of this a broadcast receiver was also needed to scan for the APs in range, it was created as a subclass and registered in the *onResume* method that is called every time the activity runs. It is responsible for listening for information from the system, in this case used to listen for the Wi-Fi data that the system was receiving, to do so does not require the device be connected to the network only to have Wi-Fi enabled. This was checked programmatically and if it wasn't it was switched on using *setWifiEnabled* method provided by the WifiManager API, after checking for permission with the user.

```
void scanWPS() {

    int numResults = 0;

    wifiManager = (WifiManager) getSystemService(Context.WIFI_SERVICE);

    //check wifi is enabled
    Boolean enabled = wifiManager.isWifiEnabled();
    if (!enabled) {
        wifiManager.setWifiEnabled(true);
    }

    //start scanning
    wifiManager.startScan();

    //getScanResults and store in arrayList
    List<ScanResult> scanResults = wifiManager.getScanResults();
```

Figure 15-Code snippet demonstrating the Wi-Fi scanning implementation in Android

Once this was confirmed a scan is started by using the *startScan* method also provided by the WifiManager, the results of which are retrieved using the *getScanResults* method which will store the data in a list of *ScanResult* objects using the class provided by Android. Each of these objects contain

attributes which describe information about the individual AP including the network it belongs to, the MAC address of the access point, the authentication and encryption used, frequency and signal level to name a few. As discussed previously the important attributes for this project include the MAC address and signal level, RSSI, from this list. If the scan returned multiple APs these two attributes would be saved from each of the APs identified, for each they would be combined into a string and stored in an arraylist. As mentioned previously this activity can be run for the purpose of collecting data during training, at this point the data would be passed back to the *UpdateLocation* activity.

At this point the *ViewLocation* activity performs much of the same processes as the *UpdateLocation*. Each string in the list is converted to JSON format using GSON and the data list is sent to a Java servlet using a HTTP POST method created using the OkHTTP API. There are a couple of distinctions between the two however, firstly the data sent from the *ViewLocation* does not include a room number as it is unknown, and it is sent to the *View* servlet this time which will use the data as part of the classification to find the unknown value. In the same fashion as before the servlet will send a HTTP response, however this time it will contain the predicted location. The system is to return a prediction to the users device within 10 seconds after which the server connection will timeout and the user will be informed that the localisation was unsuccessful, it should be noted that the system was always able to respond within 10 seconds during testing.

The prediction along with a map image is displayed to the user, the map to be displayed is selected based on the value returned. This is achieved through the use of another third-party library, this time the library Glide which is an image loading framework that fetches and displays remote images [31]. The decision to use this was based on the simplicity of use and to reap the performance benefits. Figure 16 demonstrates how it was implemented to load the prediction map to the user display.

```
//loads map using glide library
private void loadRoomMap(String s){

    String url;

    //calls loadMap class which is used to select the appropriate map
    //based on prediction
    LoadMap loadMap = new LoadMap();

    //assigns url of map to be loaded
    url = loadMap.loader(s);

    if(!isDestroyed()) {
        //Glide used to load map into the imageview
        Glide.with(this)
                .load(url)
                .into(imageView);
    }
}
```

Figure 16- Code showing the use of Glide to load maps of the Polly Vacher building with prediction

5.2.2 Java Servlets

The server component of this system comprises of two Java servlets contained in an Apache Tomcat container on the platform-as-a-service (PaaS), Amazon Elastic Beanstalk. Although this very powerful cloud computing service was not necessary for a project of this size, it was still pursued as a learning opportunity extending knowledge gained through the Advanced Computing module by developing a better understanding of the practical elements. Its use did have a number of benefits on this project beyond education. The immediate benefit of using Elastic Beanstalk is that it simplifies the process of deploying and managing the web server portion of the system, initially an environment was created through the management console accessed using a web browser then subsequent versions were simply

deployed through a plugin for the Eclipse development environment. It also allowed for simplified debugging during development by accessing server logs again using the online management console. Furthermore it provides support for future extension of this system, for instance should it be implemented as part of a university wide system with a large number of users Elastic Beanstalk would be able to handle scaling and load-balancing to ensure reliability.

The servlets developed were to provide two core functions as alluded to previously, these were to act as an interface between the Android application and the database and to implement the classification algorithm. As a result two servlets were created, the *Update* servlet and the *View servlet.*

The *Update* servlet handles the connection of WPS to the MongoDB database, it does so through the *doPost* method. As mentioned during the Android implementation section WPS transfers the Wi-Fi data using a HTTP POST method, the *doPost* processes the request when it is received. It does so by reading the JSON string from the body of the POST request and passing it as a parameter to the *MongoConnection* class, which is responsible for communicating with the database and storing the data, this will be discussed in the next subsection.

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

    //get live data from HTTP request
    readLocationData rld = new readLocationData();
    liveReading = rld.read(request);

    //send data to prediction class to be applied to classifier
    prediction p = new prediction();
    p.getLiveData(liveReading);

    //perform classification. Returns room number prediction
    int i = p.predict();
    String location = Integer.toString(i);

    //return prediction as HTTP response message to app
    response.setContentType("text/plain");
    response.getWriter().println(location);
}
```

Figure 17- Code snippet showing the doPost method from the View servlet

The *View* servlet performs similar operations as *Update* in order to fulfil its responsibility of initiating the classification algorithm. Once again it receives the Wi-Fi data through a POST request and uses *doPost* to process the message, reading the JSON data from the request body. However this time it passes it to another class called *prediction* and the method *predict* is called initiating the classification process which will be discussed in a later subsection. Once the prediction has been made it is returned to the servlet which then passes it to the application as the response message.

5.2.3 MongoDB

As mentioned during the solution approach, the data store that was implemented for this project is a MongoDB database through the DaaS mLab. Once again while a cloud computing solution was not necessary it was chosen in order to gain some practical experience with different technologies that might be of use to the developer in the future. The particular deployment for this application is simple with a comparatively small dataset with only a few attribute types per document. The database was created to store a range of Wi-Fi readings that make up the training set, currently it consists of a single collection called ‘sse’, this number could be increased if the system was to be expanded to include a range of buildings in the university. The collection contains a set of documents which represent the various pairs of locations and Wi-Fi data, an example can be seen in the figure 18.

```
{
  "_id": {
     "$oid": "58cbb5de4ae8bb1a763b8554"
  },
  "locationID": 43,
  "readings": [
     "24:de:c6:26:b1:d3 -49", "24:de:c6:26:b1:d0 -49",
     "24:de:c6:26:b1:d1 -49", "24:de:c6:26:b1:d2 -49",
     "24:de:c6:27:02:90 -67", "24:de:c6:27:02:93 -67",
     "24:de:c6:27:02:92 -67", "24:de:c6:27:02:91 -68",
     "24:de:c6:26:e8:11 -68"
  ]
}
```


Figure 18- Example of Document taken from MongoDB database (Modified to show fewer readings)

Each document is assigned a unique identifier as can be seen by the '$oid' field, the data that is stored and used by the positioning system can be seen under the locationID and the readings fields. The locationID is simply the room number assigned by the user during training and the readings field is a MongoDB binary JSON (BSON) array which contain a MAC address and RSSI, the length of this array does vary according to the number of APs found but they are all treated the same way by the database.

The communication between the database and the other components is achieved using the Java drivers provided by MongoDB. The part of the server program that implemented the connection was the *MongoConnection* class, as discussed methods from this class were called by the *Update* servlet to prepare the data and then store it, this was the *sendToMongo* method. The particular database that was to be used was selected using a *MongoConnectionURI* object which was then passed to an instance of the *MongoClient* class which was responsible for establishing the connection with the MongoDB deployment. A connection to the specific database being used was achieved through the *getDatabase* method for a *MongoDatabase* object and similarly for the 'sse' collection using *getCollection* for a *MongoCollection*. The string of Wi-Fi data that had been transferred from WPS gets converted to BSON and then inserted into the collection.

```
protected void sendToMongo(String loc){

    //establish connection to database
    MongoClientURI uri = new MongoClientURI("mongodb://kd016499:passwerc@cs145019.mlab.com:45019/wifi_readings");
    MongoClient client = new MongoClient(uri);
    MongoDatabase db = client.getDatabase(dbName);

    //select the collection to update
    MongoCollection<Document> readings = db.getCollection(collection);

    //create a Document from the location string
    Document doc = Document.parse(loc);
    //insert document to the collection
    readings.insertOne(doc);
    //close connection
    client.close();
}
```

Figure 19- Code snippet showing the primary method of the Update servlet

The *View* servlet calls a different method, *getFromMongo*, which uses the Java drivers to retrieve already stored readings rather than generating a new one. This method connects to the same database using the same instances and methods as above. This time however it iterates through each of the

documents using an iterator with a *MongoCursor* instance, the document is then converted back to standard JSON and the required fields are extracted. These values are then used to create a *location* object, which similarly to the location object in Android contains attributes for the room number and a list of APs and RSSI. An object is created for each document and is then added to an arraylist which is used as part of the input data for the NN classification implementation.

5.2.4 Classification Algorithm

The final part of the server program to discuss is the part that implements the NN model to perform the location prediction. Once the *prediction* class is instantiated in the *View* servlet, two methods are called to facilitate the function these are the *getLiveData* and predict methods. The first is simply used to retrieve the unknown record that has been transferred from the Android device, the data is converted from JSON array and stored in an arraylist to be input to the model. The predict method is the primary method responsible for performing the classification, it is here that the training data is also retrieved to be input in the dataset, achieved through the *getFromMongo* method described in the last subsection and which returns a list of objects to be used for comparison.

The *prediction* class is subdivided into methods which each implement a stage of the NN algorithm. Firstly some data pre-processing is applied to handle the likely possibility that not each stored reading will contain all of the same MAC addresses as the unknown record. The first step can be seen in figure 20, it involved comparing each stored record against the unknown record, if the stored record does not contain a MAC address that is seen in the unknown it is temporarily added and the value -120 is assigned. The reason this value was chosen is that no AP has been observed to have an RSSI value that low, setting it to -120 thus has the effect of increasing the distance that is computed.

```
//for each of the live readings
for(String s : liveDataList){
    //assign current MAC to use for comparison
    currentMac = s.substring(0, 17);

    //if MAC isn't in stored readings list
    if(!tempStored.contains(currentMac)){

        //add it and assign the value -120
        currentMac = currentMac + " -120";
        storedReadingsList.add(currentMac);
    }
}
```

Figure 20- Code snippet demonstrating the data pre-processing

Once the data has been prepared the comparison between the live data and each of the stored instances can begin. Each MAC address that is present in the live data is entered as a row in a two dimensional array, the signal strength from the live reading is entered as the first element of that row and the signal strength from the stored reading is entered as the second element. This table is created for one pair at a time and the distance is calculated before then moving on to the next pair. Following the creation of this table it is possible to apply the Euclidean distance calculation to the data.

As described during the solution approach section, Euclidean distance is the square root of the sum of squared differences [23]. Each stage of this calculation is split into different methods, the first being the diffSquared method which is responsible for calculating the squared differences between RSSI values of the two instance of the MAC addresses present and can be seen in figure 21. This is accomplished by iterating through the pairs of signal levels stored in the two dimensional *signalPairs* array created above, both elements from the current row are assigned to integer attributes and the difference between them is calculated an assigned to the integer attribute *diff*. This was then squared by multiplying it by itself and the product was added to an arraylist of squared differences, this was done for each row in the *signalPairs* array.

```
private void diffSquared(){

    sqResults.clear();

    //iterate through each pair in the array
    for (int n = 0; n < signalPairs.length; n++) {
        //retrieve pair of signals
        int q = signalPairs[n][1];
        int p = signalPairs[n][0];

        //find difference between pairs
        int diff = q - p;

        //square the difference
        int sQ = diff*diff;

        //add squared difference to list to be summed
        sqResults.add(sQ);
    }

    signalPairs = null;
}
```

Figure 21- Code snippet from View servlet showing the diffSquared method

Once the list of squared differences had been accumulated a method called *euclideanDistance*, seen in figure 22, was used to apply the final part of the distance function. The squared differences in the list were then summed using an enhanced for loop, the *sqrt* method from the *Java.lang.Math* class was applied to the total sum value to give the Euclidean distance between the two instances. The value was then returned to the primary method predict which then compares the distance to the current lowest distance that has been identified. The *lowestDistance* attribute is initially set to an arbitrarily large value, if the *euclideanDistance* is lower it will replace the *lowestDistance* value and a new predicted location will be set. NN performs an exhaustive search through the training set, thus the program waits for all of records to have their distance processed before returning the final prediction to the *View* servlet which then sends the HTTP response with the prediction.

```
private double calculateEuclidean(){

    int sum = 0;
    double euclideanDistance = 0.0;

    //add all of the squared differences
    for (int i : sqResults) {
        sum += i;
    }

    //find square root of sum to give euclidean distance
    euclideanDistance = Math.sqrt(sum);

    //return value to predict method
    return euclideanDistance;
}
```

Figure 22- code snippet showing the calculateEuclidean method

5.2.5 Location Visualisation

To enhance the user experience, when the prediction is made a map of the building is displayed to the user with the estimated room highlighted for the user to identify. To accomplish this, for each room in the building a map has been created and the room has been highlighted. The decision was made to produce individual images rather than using a single image and using the canvas class in Android to draw on the location, since it provided a clearer visualisation based on the quality of the provided maps.

As a result of this decision a large number of images would need to be available, this was implemented by privately storing the maps using the online image hosting service Photobucket. This service assigns a URL to each image so that they can be shared, each of these are stored in the Android application and the URL of the required map is passed as a parameter into the Glide load method as seen in Figure 16.

5.2.6 Summary

To summarise the implementation outlined in detail, this section will briefly describe the overall system implementation at a higher level in relation to how the user would implement each phase of the fingerprinting approach.

When the user wants to add to the training dataset they will go into the target room and will take readings at multiple points using the WPS application. From the main menu they will press the *Update Environment* button which will take them to the *UpdateLocation* screen where they will be prompted for the number of the room they are updating, for instance if they were in the G56 computer lab they would input 56. Instructions on how best to take the readings will also be displayed on the screen through a dialog box before asking the user for the location. Each time they press the submit button a reading is sent from the application to the web server application, specifically the *Update* servlet. The data will then be formatted as a MongoDB document and submitted to the database, a notification of completion is then presented to the user.

When a user wishes to find their current location, they will navigate to the *ViewLocation* screen using the appropriate button which will also initiate a scan of the APs they are in range of. At this screen they will initially be shown an empty map and a text field informing them that the system is 'Locating…', during this time the results of the scan are packaged appropriately and sent from the application to the server using HTTP. When the server receives this data via the *View* servlet it will fetch the training data from the Mongo database. Once all the data has been gathered the server application will apply the NN model to the data, resulting in a prediction which is then returned to the user's Android device and displayed on the screen along with the relevant map from the Photobucket store.

6. Testing: Verification and Validation

Testing is a critical phase of any software project, it identifies faults and ensures that the implementation has delivered a solution which meets the requirements of the project. This section will discuss the testing that has been conducted for this project and focusing on four levels, unit, module, subsystem and system testing. The hierarchy followed for this testing process can be seen in table 1 which represents the overall IPS system, the first column represents the sub-systems identified, the next level down is represented by the module column and the lowest level of unit testing is represented by the final column.

Subsystem	Module	Unit
App	Reading Wi-Fi Data	setWifiEnabled
		startScan
		getScanResults
	Package Wi-Fi in JSON for data transfer	toJSON (both al in view and object in update)
	Send to server	sendToServer (in both)
		post
		postFramework
	Read data sent from servlet	Response.body.string
	Retrieve relevant map	loader
		Glide.with.load (within loadMap)
	Display information to user	onPostExecute
		setText
		loadMap
	Take user location Input	editText.getText.toString
Servlet 1 (update)	Read data sent from app	read
	Prepare data to send	Document.parse
	Send data to db	insertOne
Servlet 2 (view)	Read data sent from app	read
	Read training data	find
		iterator
	Format training data for prediction	toJSON
		getINT
		toString
		location
	Perform prediction	setDifferentMacs
		getSignalPairsToCompare
		diffSquared
		calculateEuclidean
	Send prediction to app	getWriter.println
Database	Handles requests	New data exists

Table 1- A table representing the hierarchical breakdown of the system that was followed in testing

6.1. Unit Testing

The lowest level of testing applied was unit testing, the purpose of which is to assess whether particular sections of source code are functioning properly and deliver the correct results [32]. For this project, manual unit testing was conducted throughout the development process through the use of log statements in Android and the *println* method in Java. This was done to confirm that the particular function was performing as intended before integrating it with the other functions in that module. The most significant unit tests conducted are outlined below as an example of the work done.

One particular example of unit testing was done early on in development when implementing the Wi-Fi ability of the application sub-system, the *getScanResults* method was tested by printing out the list of results to the Android monitor using the Log class provided, the purpose of this was to verify that the results could be retrieved and used later on. Another example from later on in the development phase is the *calculateEuclidean* method that is part of the 'perform prediction' module in the *View* servlet sub-system. To test this values a set of integer values were passed to the method, the results from the method were compared against manual calculations to verify that it was performing correctly.

6.2. Module Testing

The second level of testing conducted was module testing, a module is made up of a composition of units and once it was confirmed that these were performing as intended it was possible to combine them and test them as a collective [33]. The purpose of this was to verify that the components of a subsystem work before combining them to implement the intended functionality, as such this was also performed during the development phase. An example of this is testing the Wi-Fi scan component to verify that it will assist the training data collection as part of the Android application.

The table below demonstrates the module testing that was conducted during the development of this IPS. Each row corresponds to a test case carried out, it consists of a unique ID, the test scenario explaining which module was being tested, a description of how the test case was applied, the desired result from the test and finally the actual result for comparison. Test cases 1-7 are testing modules in the app sub-system, 8-10 are modules in the *Update* servlet, cases 8-15 are module tests for the *View* servlet.

Test Case ID	Test Scenario	Description	Desired Result	Actual Result
1	Reading Wi-Fi Data	Scan was conducted and readings were printed to screen	Set of Wi-Fi readings visible on screen	Wi-Fi readings on screen
2	Convert Wi-Fi in JSON for data transfer	Wi-Fi data before and after JSON conversion printed to Android logcat	Two sets of identical data with exception of format	Two sets – data preserved
3	Send to server	Server returns reading as response. Both HTTP request and response body printed	Matching request and response	Matching sets
4	Read data from server	Response value set in server. Request sent	Response received should	Response value matches server

			match value set in server	
5	Retrieve relevant map	URL of known map passed	Known map should be displayed	Correct map was displayed
6	Display information to user	Prediction value is input	Map should be displayed as should text explanation	Map and text appear
7	Take user location input	Input value to field, print value stored to logcat	Logcat output and input value should be same	Values match
8	Read data sent from app	Print out data on client side and then again on server side	Both values should be the same	Values match
9	Prepare data to send	Check if generated document is empty	Should not be empty	Is not empty
10	Send data to database	Data is sent. Visual verification through database interface	Last entry into database should be data sent	Correct data is last entry
11	Read data sent from app	Print out data on client side and then again on server side	Both values should be the same	Values match
12	Read training data from database	Print out training records to server logs	All values from database should be present in logs	All values from database are present
13	Format training data for prediction	Print out length of list of location objects	List size should be match number of entries in database	List size equal to database size
14	Perform prediction	Reading is input to prediction algorithm	Integer value should be returned	Integer is returned
15	Send prediction to app	Print out prediction on server side and again on app	Predictions on both should be same	Prediction values matched

Table 2- Table showing the module tests performed

6.3. Sub-system

The third level of testing executed was sub-system testing, this involved combing the modules which comprise the sub-system to then test it as a whole [33]. As seen in table 3 the sub-systems identified and tested for this project were the Android application, the two servlets and the database. Before this level of testing can be applied knowledge that the individual modules are accurate is required, thus the module testing must happen before. This level was conducted after the implementation phase was completed. The testing done will be presented in the following subcategories:

Sub-system 1- Android Application:

Test Scenario	Test Description	Desired Result	Actual Result
Buttons perform as expected when pressed by user	ViewLocation button	Taken to new screen and prediction made	New screen loads and prediction is output
	UpdateEnvironment Button	Taken to update environment screen	New screen loads with prompt to input location
	Submit button in Update	Confirmation a reading has been sent to the database	Temporary confirmation appears
	Back button from View	Taken to main menu	Main menu is loaded to screen
	Back button from Update	Taken to main menu	Main menu is loaded to screen
Invalid User input is handled	INPUT: empty and submit button pressed	Dialog informing user of required input format should appear	Dialog appeared
	INPUT(tablet): non-numeric character entered and submit button pressed	Dialog informing user of required input format should appear	Dialog appeared
Application handles different changes in state	Screen: Main Menu State change: Paused	Application pauses successfully. Removed from the screen to reveal home screen.	Home screen is loaded
	Screen: Main Menu State change: Resume from pause	Application should resume successfully after being paused. Same screen as left should be loaded.	Main Menu is loaded again
	Screen: Main Menu State change: Close	Application should close successfully. Home screen should be revealed. Start of app should be loaded when opened again.	Home screen appears on close. When started again the main menu is loaded
	Screen: View Location State change: Paused	Application pauses successfully. Removed from the screen to reveal home screen.	Home screen is loaded
	Screen: View Location State change: Resume from pause	Application should resume successfully after being paused. Same screen as left should be loaded.	View location screen is loaded again
	Screen: View Location State change: Close	Application should close successfully. Home screen should be revealed. Start of app should be loaded when opened again.	Home screen appears on close. When started again the main menu is loaded
	Screen: Update Location State change: Paused	Application pauses successfully. Removed	Home screen is loaded

		from the screen to reveal home screen.	
	Screen: Update Location State change: Resume from pause	Application should resume successfully after being paused. Same screen as left should be loaded.	Update screen is loaded again
	Screen: Update Location State change: Close	Application should close successfully. Home screen should be revealed. Start of app should be loaded when opened again.	Home screen appears on close. When started again the main menu is loaded

Table 3- A table demonstrating the testing performed on Android sub-system

Sub-system 2- Java Servlet 1 (update):

The *Update* servlet provides limited functionality and thus is a simple sub-system to test. The testing was performed to verify that it is able to take readings as input and send them to the database, in order to test this the *Update environment* function was run on the Android device and the reading was sent to this servlet. The desired result was that the new reading, identified by the room id value entered, should be present in the database. Visual inspection of the database through mLab in a web browser was performed to verify this, the result of this test was that the new reading was present.

Sub-system 3- Java Servlet 2 (view):

With confirmation that the individual modules of the *View* servlet were performing as desired the overall sub-system was relatively simple. The testing was performed to prove the servlet was able to take two inputs, one reading from the Android app and training data from the database, and use them to perform classification to return a prediction to the app. In order to test this the *View Location* function was run on the Android device which sent a reading the servlet initiating the process. The desired result was that a prediction was returned to the Android app, this was verified through outputting the prediction to the devices screen. The observed result of this test was that a prediction was made and output, confirming that the servlet performed as desired.

Sub-system 4- Database:

The database is provided by a third-party and it does not perform any functionality beyond storing data that is sent to it, this has been verified through the previous tests.

6.4. System Testing

The previous three levels have been concerned with verifying the system delivers the functionality required, this final level is used to validate the solution approach that has been implemented by testing the whole system in the context of the project scope [34]. For this project, system testing involved physically testing the positioning capabilities of the solution, this was conducted in the primary test environment which was the Polly Vacher building. In order to do so the accuracy of the NN using Euclidean distance was determined by the tester going into rooms throughout the test locations and recording the predicted value against the known room number. For comparison further accuracy tests were performed using NN with Manhattan distance.

Floor	Correct	Errors
Upstairs	23	2
Downstairs	15	0
Total	**38**	**2**

Table 4- Results of physical tests of NN using Euclidean distance

Table 4 shows the results of the physical testing of the solution chosen when tested in the Polly Vacher building. The upstairs testing was conducted in rooms 139, 141, 165, 167 and 177, the testing downstairs was in rooms 43, 45, 56. Each set of tests involved going into the room and taking five readings at various positions and different orientations. From the table it can be seen that NN with Euclidean had an overall accuracy of 95% in the tests presented.

Floor	Correct	Errors
Upstairs	19	6
Downstairs	15	0
Total	**34**	**6**

Table 5- Results of physical tests of NN using Manhattan distance

Table 5 shows the results of the physical testing of NN with Manhattan distance for comparison. The testing was conducted in the same rooms as the tests represented in table 4 and in the same manner. The results of these comparison tests indicate an overall accuracy of 85%. The full test results can be seen in appendix C and D.

6.5. Testing Limitations

A number of limitations of the testing conducted have been identified upon reflection. Firstly the collection of the training data and the subsequent testing performed were done using a single Android device, this approach has not explored the effect different hardware has on the effectiveness of the system. The device used was a newer Android smartphone which would benefit from greater Wi-Fi signal sensitivity than older devices, this is likely to have an effect on the number of APs visible to the receiving device and thus the predictions made.

Another limitation is that the testing was conducted using training data that was collected over a short period of time, as a result this is not reflective of the evolving Wi-Fi environment. These tests represent how the system would perform under the ideal conditions of up to date fingerprints rather than a more realistic scenario, this limitation will be presented in greater depth during the discussion section. Similarly due to accessibility, the testing was only conducted across a selection of rooms that were chosen to represent different positioning scenarios for instance between a collection of neighbouring rooms, between floors and isolated rooms. To enhance this, a greater number of rooms could have been used for the formal testing.

7. Discussion: Contribution and Reflection

The previous section presented the results of the bottom-up hierarchical testing performed during this project, this section will endeavour to discuss the findings of each level and will highlight the successes and limitations of this project implementation. Finally the learning experience of the developer will be reflected on.

The testing section of this report highlighted the various levels testing was conducted at, the first of which was unit testing. It was described as on ongoing process during the development of this system, focusing on testing individual units of code ensuring they work as expected before moving on in an incremental manner. As such it proved to be an extensive process, that does not require formal detailing of the individual tests performed rather a couple example were presented to explain the process. This highlighted the advantages of this level when applied, primarily it aided the implementation phase in that it meant less time was spent on debugging when something didn't work properly. With early testing, it was easier to pinpoint the cause of the problem as a better understanding of the individual units had been achieved.

Module testing was also performed throughout the development and provides the same early testing benefits that unit testing did. For this project its focus was at a higher level, considering the small functionality that the combination of units should provide. The core modules tested are seen in the table 2. The first case tested the ability of the Android application to read Wi-Fi data from the APs in the environment, the results verified that this module was functioning as expected and facilitating the basis of the system. The second case tested the application of the GSON external library for this project, these confirmed that the format conversion was successful and could be used to transfer the data. Test cases 3 and 4 were still focused on the Android application, however it required the use of a servlet to confirm the integration ability of the app with regards to transferring data over the network. Similar tests were done on the servlets ability to take input and send outputs over the network, these are seen in the test cases 8, 10, 11, 12 and finally 15. Each of these tests confirmed that the separate components would be able to facilitate the transfers. Another important module presented in the table is seen in test case 14, which is testing the methods that combine to make a location prediction, these tests identified that provided with the right data this module would return a result.

The sub-system focused at a further higher level, testing the ability of each sub-subsystem to deliver the primary behaviours they were created for. For the Android app, this involved manual testing from a user's perspective such as testing the buttons delivered the response that was expected. This can be seen in the table which for each button confirms that they delivered the desired result without faults, also through the tests of the user input which confirmed that invalid input was queried and finally that each activity in the application gracefully handled a change in state without crashing. As the interface for the user to interact, the testing of this sub-system focused on the usability of the app by verifying that a user would easily be able to use the app to take advantage of the systems functionality without any issues. Testing of the servlets was also conducted at this stage, as a backend component the focus of this testing was purely based on the ability to deliver the behaviour required. For the *Update* servlet with limited functionality the testing confirmed that given a Wi-Fi reading through a HTTP message it would be able to store the data in the database. For the *View* servlet it confirmed the ability to take a set of inputs from the database as well as the Android app to then perform classification and return a value to the app.

The overall system testing validated the implementation, considering the primary object of the project to provide an accurate localisation using Wi-Fi fingerprinting. The results of the physical testing conducted suggested a successful implementation of an IPS in the Polly Vacher building, demonstrating high levels of accuracy at 95%. Alternative approaches to classification were also tested, the results of which indicate an accuracy of 85% using Manhattan as the distance measure. These tests show that for

this project test environment, NN with Euclidean distance performed significantly better than NN using Manhattan distance echoing the findings of the paper by Li et al. [9] discussed during the literature review as well as the findings of the KNIME simulations. The results show that downstairs in the Polly Vacher building the implementation performed very well with 100% accuracy in the tests conducted, this success might be due to larger rooms with a greater physical distance to other rooms which would result in a clear division between the distances computed for readings in the room and readings from other rooms. The tests of NN with Euclidean upstairs also performed well albeit slightly less accurate than downstairs with an accuracy of 92% in the tests conducted. The reason for this slight reduction is likely a result of upstairs being a more densely populated environment with a greater number of rooms of smaller sizes and less distance between them. These results demonstrate a clear ability of this implementation to select the correct location of the user.

Through the formation of this report, a number of successes and limitations of this solution have been identified. The first point to make is of the successfulness of the system created to provide accurate indoor positioning capabilities, the system testing has identified that the use of NN with Euclidean distance has proved to be a successful implementation with an accuracy of 95% in the Polly Vacher testing environment. Further to that the system implemented provides a framework which with little modification could be applied to other buildings on campus or indeed anywhere else with sufficient Wi-Fi infrastructure, and can serve as the basis for a number of potential applications. This system may be considered a proof of concept for future work that will be discussed in the following section. The Android app that was created as part of this system might also be considered a success as it is able to facilitate the functionality that the system aims to deliver, it crucially provides the Wi-Fi data for the training and positioning phases, and it does this by providing a clean, simple interface for the user to interact with the system. The architecture of this solution provides a good level of portability for the system, the three tiers that currently exist co-operate together however could work independent of this implementation using other technologies with only little modification. For instance the java servlet programs might be replaced with another program using a specific server-side scripting language such as node.js, or the Android application might be replaced with another with a different range of functionalities that include an indoor positioning ability. The use of Amazon web services and MongoDB through mLab whilst not necessary for a project of this scope does have an advantage beyond serving as a learning opportunity, it provides scalability should this system look to be expand to a campus wide system with many concurrent users. Should greater capacity be needed from the system, these two cloud computing solutions could deliver this with minimal effort through the vast resources at their disposal.

A few limitations of this solution were also identified, most crucially it does not automatically handle changes in the Wi-Fi environment. During physical testing an observation was made that recording training data at different dates resulted in the latest readings having a greater probability of matching, this is because the Wi-Fi infrastructure at the university is managed to maximise coverage throughout the entire campus. As a result if the status of an AP was to change, for example if it was deactivated, the other APs will alter their signal strength to ensure the coverage is optimised which will change the outcome of predictions with more recent training records likely to dominate when applied to a smaller space. For this implementation the current solution to this problem would be to ensure the entire training data is kept up to date, more thorough options will be considered in the following section for future work.

Another limitation of this project that has been identified and mentioned previously is the training data used for this project was collected using the same device, the device is a relatively new Android phone and as such it contains the latest hardware. This could be considered a limitation of this project because the Wi-Fi receiver on the more recent hardware is likely to be more sensitive and will have a greater range of APs, the limitation is that this project has not been able to test the impact of using devices of differing ages and hardware.

A further limitation of this project is that due to restricted access to rooms, full coverage of the building in the training data was not attained, the coverage of the Wi-Fi fingerprints can be seen in appendix E. The aim of the fingerprinting conducted was to have sufficient coverage through both levels of the building to give an accurate representation of its localisation ability. The aim was to get coverage of each section of the building with multiple rooms in each section, this was to test the ability of the system to distinguish between floors and then between rooms situated closely on each floor. Having full coverage of the building would have provided the best representation of the system as possible.

One final limitation is again of the training and testing of this implementation, it is that currently only a room level positioning ability has been covered. This might have been furthered to include corridors as part of the system and could have partitioned rooms to test a finer granularity, these would've allowed for the application of a location tracking feature which would show the user exactly where they are in the building as they moved around.

This project has had a number of challenges that had to be overcome, in doing so it has served as a great learning experience. The most significant challenge was that the field of data mining was completely new to me at the start of the year, initially this required self-study through online tutorials and text books to start to build an understanding. However it was not until the spring term when I attended the Data Mining module that I feel a true understanding of the project and the techniques necessary was developed, the lectures and coursework greatly aided my understanding of the KDD process and classification and how they could be applied to this project. With the benefit of the teaching Data Mining teaching module in autumn term progress in the early phases of this project would have been greatly enhanced.

New technologies were also explored throughout this project, as mentioned the cloud computing solutions of Elastic beanstalk and mLab were used with the sole intention of furthering my understanding and experience of them in a theoretical manner through the Advanced Computing module. KNIME was also explored for the first time, the simplicity of its interface allows the application of what otherwise would be significantly more complicated and time-consuming tasks. The incredible power of this tool was realised again through the coursework in the Data mining module and once applied to this project it provided great assistance in understanding the abilities of the various classifiers that were considered, again if I had more experience with this tool from earlier on in the project and with better understanding of data mining I feel this project would've seen greater levels of success.

Through this project I revisited technologies that I had experience with from previous years, namely the Java and Android. I was able to further develop skills learnt in the second year of university and apply these technologies to a much greater project than those of previous, in doing so I covered a number of areas I had little or no experience with such as network programming in Java and the use of external libraries. This was the first time I had deployed third party libraries in any of my university projects, the use of OkHTTP, GSON and Glide proved very useful and primary benefit that is the time saved on development through using already tried and tested solutions.

The final point of reflection is that this was the first time I have worked on a major project with a life cycle of more than a couple of months as I opted not to do a year in industry. While it was a challenge adapting to a project of this size from the typical coursework exercises I was used to, it provided a great opportunity to understand the work that goes into larger projects and the various skills that are required, both technical and non-technical. It helped me to identify weaknesses in my understanding of topics I have learned at university such as software engineering methodology, it has helped me begin rectification of these and has prepared me for my professional life post-university.

8. Social, Legal, Health & Safety and Ethical Issues

The initial PID highlighted a couple social, legal and ethical issues, however upon reflection these do not apply to this project implementation rather they might arise from applications of this technology which extend this current implementation.

The first issue outlined in the PID was the social implications of acquiring information about a user, this is not an issue for this project as no information is collected about the user or their device simply the Wi-Fi APs they are in range of. Similarly an ethical concern about tracking and storing a user's location was addressed, again this is not considered an issue for this project because the localisation capabilities are only implemented once the user initiates them and no data about the user is stored.

Another issue was with regards to the ownership of the data, currently this is not an issue for reasons stated previously. The final issue to address is the misuse of location data by an individual, as the system does not identify specific users or store location information this is not considered an issue.

No health and safety issues were raised in the PID and this remains the case.

9. Conclusion and Future Improvements

The principle objective of this project was to deliver an accurate indoor positioning system that followed the approach of Wi-Fi fingerprinting, it was to facilitate two core functions that represent the training and positioning phases of this approach. In order to perform location prediction the technique of classification would be applied to the Wi-Fi data that made up the training and test data. An Android application was needed to serve as an interface for the user, allowing them to view their current location as well as contribute to the system by storing readings that would make up training data. Further to the Android application a combination of backend technologies were needed to apply the classification that would generate a prediction as well as storing the training data that is collected through the app. Finally a number of classification models were to be analysed and compared to ensure that the optimum solution was found for this indoor localisation problem.

Through the work described in this report these objectives have been realised. The solution implemented achieves the primary objective of this project, demonstrating high levels of accuracy in the physical testing of the system. It does this by applying nearest neighbour classification to the positioning phase of the Wi-Fi fingerprinting approach, identifying the closest match from the stored fingerprints using the Euclidean distance measure. To achieve this a number of classification models were tested using the KNIME platform and the best solution of NN using Euclidean distance was selected to be implemented. The decision was supported by previous studies as well as the simulation testing, NN had the advantage of delivering high levels of accuracy and simple implementation. The Android application was successfully implemented and provided a simple and easy to use interface which allowed the user to initiate the accurate positioning ability. In support of the application a combination of backend technologies were deployed, this reduced the demand on the user's device and added a greater level of portability to the implementation.

The system produced during this project, whilst not a finished product, can be considered a proof of concept demonstrating the feasibility of such a system should it be applied in the future. As such there are a number of areas that would be considered for refinement and future improvement, the first of which addresses the limitation of the system's ability to handle a change in state of the Wi-Fi infrastructure. In the future this might be refined through the implementation of an adaptive classification algorithm, this could be a simple extension of adding a validation phase to the current system. This would involve submitting a known reading to the training set and identifying the change that has occurred, this could then be used to adjust the readings in the training set. This would be a fairly simple task as the portability of the system would mean only the server side program would have to be changed.

An obvious extension of the current system would be to implement a finer granularity, this would involve attempting to predict the user's exact position in the room rather than just identifying the room they are in. The current system could easily be altered to accommodate this, it could store the specific recording point that a fingerprint reading was taken at as well as the room. Similarly the current system could be slightly adjusted to work on a larger scale, a building identifier field could be added to the training records allowing for the expansion across multiple buildings on campus. Finally this proof of concept might be extended to provide location based services on campus, these might include a navigation tool to find lecture rooms, study space capacity monitoring or a lecture attendance tracking tool which would identify if students are in the lecture hall at the time specified.

10. References

[1] D. Lee, "Garmin | What is GPS?", *Www8.garmin.com*. [Online]. Available: http://www8.garmin.com/aboutGPS/. [Accessed: 09- Apr- 2017].

[2] "Indoor Positioning for Retail - Senion | Indoor Positioning System", *Senion | Indoor Positioning System*. [Online]. Available: https://senion.com/indoor-positioning-for-retail/. [Accessed: 09- Apr- 2017].

[3] Kleusbeg A, Langley R.B. 1990. "The Limitations of GPS". [Online]. Available at: http://gauss.gge.unb.ca/gpsworld/EarlyInnovationColumns/Innov.1990.03-04.pdf (Accessed: 21/03/2017)

[4] R. Henniges, "Current Approaches of WiFi Positioning", Berlin, 2012, Available at: https://www.snet.tu-berlin.de/fileadmin/fg220/courses/WS1112/snet-project/wifi-positioning_henniges.pdf.

[5] M. Sauter, *From GSM to LTE-advanced*, 1st ed. 2010, p. 160.

[6] Wi-Fi Location-Based Services 4.1 Design Guide, Cisco Systems Inc, San Jose, CA, 2014, Available at: http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Mobility/WiFiLBS-DG/wifich2.html

[7] E. Navarro, et al., "Wi-Fi Localization Using RSSI Fingerprinting", California Polytechnic State University, Available at: http://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1007&context=cpesp.

[8] G. Di Fatta, "Regression and Classification", From CS3DM16 – Data Mining, University of Reading, Available on Blackboard

[9] D.Li, et al., "Indoor Positioning System Using WiFi Fingerprint", Standford University, Available at: http://cs229.stanford.edu/proj2014/Dan%20Li,%20Le%20Wang,%20Shiqi%20Wu,%20Indoor%20Positioning%20System%20Using%20Wifi%20Fingerprint.pdf

[10] V. Moghtadaiee and A. Dempster, "Vector Distance Measure Comparison in Indoor Location Fingerprinting", in *IGNSS Symposium*, Qld, Austrailia, 2015, Available at: http://www.ignss.org/LinkClick.aspx?fileticket=yG6nT9XYQPk%3D&tabid=147&mid=558

[11] S. Sayad, "Decision Tree", *Saedsayad.com*, 2010. [Online]. Available: http://www.saedsayad.com/decision_tree.htm. [Accessed: 30- Apr- 2017].

[12] S. Ray, "Understanding Support Vector Machine algorithm from examples (along with code)", *Analytics Vidhya*, 2015. [Online]. Available: https://www.analyticsvidhya.com/blog/2015/10/understaing-support-vector-machine-example-code/. [Accessed: 30- Apr- 2017].

[13] Indoor Positioning and Navigation – A Guide on Technologies and Use Cases, insoft GmbH, Dusseldorf, Germany, 2016, Available at: https://www.infsoft.com/portals/0/images/solutions/basics/whitepaper/en-indoor-navigation-indoor-positioning-infsoft-ebook.pdf

[14] "android.net.wifi | Android Developers", *Developer.android.com*. [Online]. Available: https://developer.android.com/reference/android/net/wifi/package-summary.html. [Accessed: 09- Apr- 2017].

[15] "WifiManager | Android Developers", *Developer.android.com*. [Online]. Available: https://developer.android.com/reference/android/net/wifi/WifiManager.html. [Accessed: 09- Apr- 2017].

[16]"ScanResult | Android Developers", *Developer.android.com*. [Online]. Available: https://developer.android.com/reference/android/net/wifi/ScanResult.html. [Accessed: 09- Apr- 2017].

[17]"Connecting to the Network | Android Developers", *Developer.android.com*. [Online]. Available: https://developer.android.com/training/basics/network-ops/connecting.html. [Accessed: 30- Apr- 2017].

[18] P. Sadalage, "NoSQL Databases: An Overview", *ThoughtWorks*, 2014. [Online]. Available: https://www.thoughtworks.com/insights/blog/nosql-databases-overview.

[19] T. Marston, "What is the 3-Tier Architecture?", *Tonymarston.co.uk*. [Online]. Available: http://www.tonymarston.co.uk/php-mysql/3-tier-architecture.html. [Accessed: 14- Apr- 2017].

[20] Y. Luo, et al., "Enhancing Wi-Fi fingerprinting for indoor positioning using human-centric collaborative feedback", 2013, Available at: https://hcis-journal.springeropen.com/articles/10.1186/2192-1962-3-2

[21]C. Thornton, "Machine Learning - Lecture 5: Cross-validation", *Users.sussex.ac.uk*. [Online]. Available: http://users.sussex.ac.uk/~christ/crs/ml/lec03a.html. [Accessed: 30- Apr- 2017].

[22] J. Schneider, "Cross Validation", *cs.cmu.edu*, 1997. [Online]. Available: https://www.cs.cmu.edu/~schneide/tut5/node42.html. [Accessed: 01- May- 2017].

[23]"Euclidean and Euclidean Squared Distance Metrics", *Improvedoutcomes.com*. [Online]. Available: http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Euclidean_and_Euclidean_Squared_Distance_Metrics.htm. [Accessed: 01- May- 2017].

[24]"Manhattan Distance Metric", *Improvedoutcomes.com*. [Online]. Available: http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Manhattan_Distance_Metric.htm. [Accessed: 01- May- 2017].

[25] C. Perone, "Machine Learning :: Cosine Similarity for Vector Space Models (Part III)", *Terra Incognita*, 2013. [Online]. Available: http://blog.christianperone.com/2013/09/machine-learning-cosine-similarity-for-vector-space-models-part-iii/.[Accessed:30-April-2017].

[26] "What Is MongoDB?", *MongoDB*. [Online]. Available: https://www.mongodb.com/what-is-mongodb. [Accessed: 09- Apr- 2017].

[27]"Java MongoDB Driver", *Docs.mongodb.com*. [Online]. Available: https://docs.mongodb.com/ecosystem/drivers/java/. [Accessed: 09- Apr- 2017].

[28] "google/gson", *GitHub*. [Online]. Available: https://github.com/google/gson. [Accessed: 10- Apr- 2017].

[29] P. Jiang, et al., "Indoor Mobile Localization Based on Wi-Fi Fingerprint's Important Access Point", *International Journal of Distributed Sensor Networks*, vol. 11, no. 4, 2015. Available at: http://journals.sagepub.com/doi/full/10.1155/2015/429104

[30]"OkHttp", *Square.github.io*. [Online]. Available: http://square.github.io/okhttp/. [Accessed: 10- Apr- 2017].

[31]"bumptech/glide", *GitHub*. [Online]. Available: https://github.com/bumptech/glide. [Accessed: 10- Apr- 2017].

[32]"What is Unit testing?", *Istqbexamcertification.com*. [Online]. Available: http://istqbexamcertification.com/what-is-unit-testing/. [Accessed: 01- May- 2017].

[33] I. Sommerville, *Software engineering*, 2nd ed. Addison-Wesley, 1985.

[34]"What is System testing?", *Istqbexamcertification.com*. [Online]. Available: http://istqbexamcertification.com/what-is-system-testing/. [Accessed: 01- May- 2017].

11. Appendices

11.1. Appendix A – Project Initiation Document

Individual Project (CS3IP16)

Department of Computer Science
University of Reading

Project Initiation Document

PID Sign-Off

Student No.	**23016499**
Student Name	**Calum McGloin**
Email	**c.mcgloin@student.reading.ac.uk**
Degree programme (BSc CS/BSc IT)	**BSc CS**
Supervisor Name	**Dr Giuseppe Di Fatta**
Supervisor Signature	**Giuseppe Di Fatta**
Date	**7/10/2016**

SECTION 1 – General Information

Project Identification

1.1	**Project ID** (as in handbook)
	160- Data Mining
1.2	**Project Title**
	Wi-Fi positioning system for the SSE Building at the University of Reading
1.3	**Briefly describe the main purpose of the project in no more than 25 words**
	The main purpose of the project is to develop a Wi-Fi Positioning System for the engineering building at the university.

Student Identification

1.4	**Student Name(s), Course, Email address(s)** e.g. Anne Other, BSc CS, a.other@student.reading.ac.uk
	Calum McGloin, BSc CS, c.mcgloin@student.reading.ac.uk

Supervisor Identification

1.5	**Primary Supervisor Name, Email address** e.g. Prof Anne Other, a.other@reading.ac.uk
	Dr Giuseppe Di Fatta, G.DiFatta@reading.ac.uk
1.6	**Secondary Supervisor Name, Email address** Only fill in this section if a secondary supervisor has been assigned to your project

Company Partner (only complete if there is a company involved)

1.7	**Company Name**
1.8	**Company Address**
1.9	**Name, email and phone number of Company Supervisor or Primary Contact**

SECTION 2 – Project Description

2.1	**Summarise the background research for the project in about 400 words. You must include references in this section but don't count them in the word count.**
	The initial stage of background research on the Indoor Positioning System (IPS) project was to research Global Positioning System (GPS), this was to develop an understanding of how GPS is used to locate the position of the receiver and the limitations that current GPS technologies have. The key limitation identified was the need for a clear unobstructed line of sight to at least three satellites. By learning about the limitations it was possible to appreciate how useful an IPS would be and how a similar approach to GPS may be applied to an enclosed environment using Wi-Fi transmitters rather than satellites. The next stage of background research to better understand the project was to start exploring the various approaches that may be used to implement an IPS. These approaches include a Wi-Fi positioning system (WPS), Bluetooth system, Visible light communication system, RFID choke point system as well as many others. For the purpose of this project the focus is on Wi-Fi systems, this has a number of advantages such as an existing Wi-Fi infrastructure can be used, this is particularly useful for a WPS at the university as there is already a developed infrastructure with no shortage of transmitters which will increase the accuracy of the predictions. A disadvantage highlighted is WPS are less accurate than a Bluetooth based system, however with the resources available for this project the WPS is the most appropriate solution. The accuracy may also be enhanced by the use of inertial sensors available in most smartphones and devices. In order to better understand the requirements of this type of IPS, research was conducted into the various applications and the possible features that could be applied to this project. One application would be a navigation system for an airport or shopping centre, this would involve mapping the location of the shops, restaurants, terminals and gates, identifying the user's current location and calculating the best route to the destination. This application could also be used on a university campus, for instance on an open day or to help students find different lecture halls and seminar rooms. Another similar application would be in grocery stores to help the user find the most efficient route around the stores, this could be applied to the university library helping users find the location of particular books or resources. [387] References: https://en.wikipedia.org/wiki/Wi-Fi_positioning_system https://www.infsoft.com/blog-en/articleid/40/indoor-navigation-using-wifi-as-a-positioning-technology https://senion.com/indoor-positioning-system/ http://www8.garmin.com/aboutGPS/
2.2	**Summarise the project objectives and outputs in about 400 words.** These objectives and outputs should appear as tasks, milestones and deliverables in your project plan. In general, an objective is something you can do and an output is something you produce – one leads to the other.

	The initial objective of this project is to develop a thorough understanding of existing IPS technologies and approaches, expanding on the research done so far. The focus is on WPS and technologies that can be used in conjunction with Wi-Fi to increase the accuracy of predictions. This might involve producing a document of research which can be analysed and used to capture the requirements for this project. Once an in-depth understanding has been achieved, the next objective will be to start applying this knowledge to produce a design document for a WPS for the SSE building with the potential to other buildings. The initial phase of this design will be to produce a Use Case model, this will be used to capture the functionalities and requirements of the system, providing context of the whole system. The following stage in the design process will be to take a deeper look at each function the system will perform and produce an activity model, demonstrating the flow of activities in the system. The next step will be to produce a class model diagram to represent the design of the data structures for the IPS. The subsequent stage will be to focus on the use cases to produce a functional workflow model for each of them, this will demonstrate the flow of activities involved with each function and how each of the objects in the class diagram interact. The final stage is to create a component model diagram which demonstrates the components required to implement the functions outlined in the previous models. Each of the models created in the design phase will be used in the next objective, the implementation of the WPS, to ensure all of the requirements are met. This involves the development of the android application which will run the data mining algorithm to predict the location of the receiver device and the orientation of the user, this will use data gather from Wi-Fi access points and sensors within the phone. The final objective will be to conduct thorough testing of the system and components to ensure it satisfies the specified requirements. Testing will be conducted throughout the project lifecycle, this will include such things as checking the use case model against the specification, reviewing the design with the intent of improving as well as testing the system once it has been completed. [395]
2.3	**Initial project specification - list key features and functions of your finished project.** Remember that a specification should not usually propose the solution. For example, your project may require open source datasets so add that to the specification but don't state how that data-link will be achieved – that comes later.
	– The system will provide user location predictions within an Indoor environment – The system should provide accurate location predictions based on Wi-Fi fingerprinting – A simple interface should be created in the form of an Android application for the user to interact with the system – The user should be able to request a location prediction – The user should be able to update a location fingerprint – Instructions should be provided to a user on how to update a fingerprint – The room prediction should be displayed to the user – Results returned should illustrate location within the building – Solution coverage should be scalable (ie can add more rooms as required) – Predictions shall be restricted to rooms, not corridors/open spaces
2.4	**Describe the social, legal and ethical issues that apply to your project. Does your project require ethical approval?**

	The most prominent social, legal and ethical issue that applies to this project is concerned with the privacy of the user. In terms of a social issue, many people are concerned about individuals or organisations acquiring information about them through their online activities, consideration will have to be given to reassure potential users that their data is private and secure. As an ethical issue, attention has to be given to whether it is right to track and store the users current location especially knowing that there are growing numbers of people who actively avoid publishing too much information. Another ethical issue of an IPS is who owns the data, is it the organisation that owns the infrastructure, the app developer that gathers the data or the user who generates the data. Consideration must be given to how each of the potential owners may use the data to their own advantage, for instance the use of targeted advertising may be an issue for some. A legal issue is that the app maybe misused by individuals to take advantage of knowing the users location, for instance they may see they are away from their home and could decide to burgle them or mug the user, this technology may facilitate individuals to commit crimes.
2.5	**Identify and lists the items you expect to need to purchase for your project. Specify the cost (include VAT and shipping if known) of each item as well as the supplier.** e.g. item 1 name, supplier, cost
	Android Smartphone or Tablet
2.6	**State whether you need access to specific resources within the department or the University e.g. special devices and workshop**

SECTION 3 – Project Plan

3.1	**Project Plan** Split your project work into sections/categories/phases and add tasks for each of these sections. It is likely that the high-level objectives you identified in section 2.2 become sections here. The outputs from section 2.2 should appear in the Outputs column here. Remember to include tasks for your project presentation, project demos, producing your poster, and writing up your report.

Task No.	Task description	Effort (days/weeks)	Outputs
1	**Background Research**	**1 week**	
1.1	Data Mining techniques	3 days	
1.2	WPS and phone sensor research	3 days	Research Document
1.3	WPS applications	1 day	Requirements List
2	**Analysis and design**	**5 weeks**	
2.1	Context Analysis	1/2 week	Case model diagram
2.2	Behaviour Analysis	½ week	Activity Diagram
2.3	Data model design	1 week	Class diagram
2.4	Functional Workflow	½ week	Sequence Diagram
2.5	Component Design	½ week	Component Diagram
2.6	Database Design	1 week	Relational Db Design
2.7	Interface Design	1 week	Interface Design
3	**Develop prototype**	**6 ½ weeks**	
3.1	Client side Android app framework	1 week	Basic android application
3.2	Database (Server)	1 week	Database to store Signal Values & Location, connected to app
3.3	Mapping of the location	3 weeks	Training data set
3.4	Data mining algorithm- classification	3 ½ weeks	Data mining model which checks values against data sets
3.5	Extension Tasks	n/a	Any extensions decided on after creation of the initial prototype
4	**Testing, evaluation/validation**	**7 weeks**	
4.1	unit testing	1 week	Analysis of each portion
4.2	System Testing	1 week	Analysis of system as whole
4.3	Functional Testing	1 week	Verify the system meets requirements
4.4	Usability Testing	1 week	Verify the usability
5	**Assessments**	**3 ½ weeks**	
5.1	write-up project report	2 weeks	Project Report
5.2	produce poster	4 days	Poster
5.3	Prepare Presentation and Demonstration	1 week	Presentation
TOTAL	**Sum of total effort in weeks**	**20 weeks**	

SECTION 4 - Time Plan for the proposed Project work

For each task identified in 3.1, please *shade* the weeks when you'll be working on that task. You should also mark target milestones, outputs and key decision points. To shade a cell in MS Word, move the mouse to the top left of cell until the curser becomes an arrow pointing up, left click to select the cell and then right click and select 'borders and shading'. Under the shading tab pick an appropriate grey colour and click ok.

START DATE: 10/10/2016

Project Weeks

Project stage	0-3	3-6	6-9	9-12	12-15	15-18	18-21	21-24	24-27	27-30	30-33	33-36	36-39
1 Background Research													
2 Analysis/Design													
3 Develop prototype.													
3b Potential prototype extension period													
4 Testing, evaluation/validation													
5 Assessments													

Week 12- Completion of first prototype (end of December)

Week 19- Demo to Supervisor

End of Project & Submission of Project Poster – 24th March

Demo & presentation – 17th-21st April Report submission -24th April

AREA HEALTH AND SAFETY RISK ASSESSMENT FORM (RA1)

Assessment Reference No.	**n/a**	**Area or activity assessed:**	
Assessment date			
Persons who may be affected by the activity (i.e. are at risk)			

SECTION 1 : Identify Hazards - *Consider the activity or work area and identify if any of the hazards listed below are significant (tick the boxes that apply).*

1.	Fall of person (from work at height)		**6.**	Lighting levels		**11.**	Use of portable tools / equipment		**16.**	Vehicles / driving at work		**21.**	Hazardous fumes, chemicals, dust		**26.**	Occupational stress	
2.	Fall of objects		**7.**	Heating & ventilation		**12.**	Fixed machinery or lifting equipment		**17.**	Outdoor work / extreme weather		**22.**	Hazardous biological agent		**27.**	Violence to staff / verbal assault	
3.	Slips, Trips & Housekeeping		**8.**	Layout , storage, space, obstructions		**13.**	Pressure vessels		**18.**	Fieldtrips / field work		**23.**	Confined space / asphyxiation risk		**28.**	Work with animals	
4.	Manual handling operations		**9.**	Welfare facilities		**14.**	Noise or Vibration		**19.**	Radiation sources		**24.**	Condition of Buildings & glazing		**29.**	Lone working / work out of hours	
5.	Display screen equipment		**10.**	Electrical Equipment		**15.**	Fire hazards & flammable material		**20.**	Work with lasers		**25.**	Food preparation		**30.**	Other(s) - specify	

SECTION 2: Risk Controls - *For each hazard identified in Section 1, complete Section 2. For more complex activities or projects you are advised to use Form RA2.*

Hazard No.	Hazard Description	Existing controls to reduce risk	**Risk Level** (tick one)			Further action needed to reduce risks *(provide timescales and initials of person responsible)*
			High	Med	Low	
Name of Assessor(s)			**SIGNED**			
Review date						

Health and Safety Risk Assessments – continuation sheet

Assessment Reference No	
Continuation sheet number:	

SECTION 2 continued: Risk Controls

Hazard No.	Hazard Description	Existing controls to reduce risk	Risk Level (tick one)			Further action needed to reduce risks *(provide timescales and initials of person responsible for action)*
			High	Med	Low	
Name of Assessor(s)			**SIGNED**			
Review date						

11.2. Appendix B - Logbook

FYP: Wi-Fi based positioning system - Logbook

Date:	Notes:	References:	Actions:
28/09/16	– Research into GPS ○ How it works ○ Limitations – Research into IPS approaches ○ BLE ○ Wi-Fi ○ VLC – Research of IPS Applications	http://www.physics.org/article-questions.asp?id=55 http://www8.garmin.com/aboutGPS/ https://www.infsoft.com/portals/0/images/solutions/basics/whitepaper/en-indoor-navigation-indoor-positioning-infsoft-ebook.pdf https://www.snet.tu-berlin.de/fileadmin/fg220/courses/WS1112/snet-project/wifi-positioning_henniges.pdf	– Create project initiation document (PID)
05/10/16	– First PID completed ○ Need to check scope of project		– Get clarification of project details – Get approval from supervisor
07/10/16	– First Meeting with supervisor ○ Clear approach laid out ▪ Using RSS fingerprints ○ Scope of project established ▪ SSE building ▪ Localisation only ○ New time plan – PID updated ○ New time plan		– Research into Wi-Fi fingerprinting approach

	▪ Prototype by the end of December ▪ SSE focused – Updated PID submitted to Lily Sun		
11/10/16	– Research into Wi-Fi fingerprinting approach	http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5648247&tag=1 https://hcis-journal.springeropen.com/articles/10.1186/2192-1962-3-2 http://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1007&context=cpesp http://cs229.stanford.edu/proj2014/Dan%20Li,%20Le%20Wang,%20Shiqi%20Wu,%20Indoor%20Positioning%20System%20Using%20Wifi%20Fingerprint.pdf	– Research android APIs
12/10/16	– Research of android APIs and data storage ○ Wi-Fi manager ○ Accessing databases	https://www.tutorialspoint.com/android/android_wi_fi.htm https://developer.android.com/reference/android/net/wifi/WifiManager.html https://developer.android.com/guide/topics/data/data-storage.html	– Practice with android
18/10/16	– Started forming requirements		
20/10/16	– Defined Specification – Created an initial use case diagram ○ Need to decide whether necessary to have so many actors		– Highlight relationships – Adjust actors

	○ Establish relationships between use cases		
24/10/16	– Android exercises ○ Focusing on layouts and data storage		
26/10/16	– Research of classification Algorithm	http://www.saedsayad.com/k_nearest_neighbors.htm http://www.ignss.org/LinkClick.aspx?fileticket=yG6nT9XYQPk%3D&tabid=147&mid=558	
30/10/16	– Research into increasing accuracy of location prediction ○ User feedback	https://hcis-journal.springeropen.com/articles/10.1186/2192-1962-3-2	
05/11/16	– Updated PID to include activities previously missed ○ Interface design		– Behaviour analysis
15/11/16	– Activity Diagram for 'view location' function – Activity model for 'input feedback' ○ May be able to remove steps – Activity model for 'update location'		– Class diagram

17/11/16	– Completed activity models		
20/11/16	– Completed sequence diagram for view location ○ Consistent with class but may need to swap functions later ▪ setLocKNN() ▪ KNN variable ▪ May go to locationFingerprint – Created v2		– Sequence diagrams
21/11/16	– Completed sequence diagram for input feedback function ○ Consistent with other diagrams ○ Unsure on the loops ▪ Possibly redundant methods – Completed sequence diagram for update location ○ Consistent ○ Check confirmRoom()		–
26/11/16	– ERD for storing RSS data ○ Flat table		– Get replacement android device – Relational model
28/11/16	– Completed relational model		– Meet with supervisor

01/12/16	– Met with supervisor ○ Crack on with dev ○ User server to store readings rather than relational database ▪ Too little data to warrant it ○ Will be a visual map of building instead of text output		– Targeting some prototype by 19th Dec
05/12/16 - 07/12/16	– Working on reading the Wi-Fi data from routers		
14/12/16	– Decided to work through an online android course to revise android		
28/12/16	– App able to read the Wi-Fi data and display it to the user		- Figure out how to setup server and store the data
10/01/17	– Met with supervisor ○ Discussed taking a more bottom-up approach ○ Focus on creating a client only version ○ Look at expanding to server later on		- Build client only wps (target 13th) - Then explore storage options
11/01/17	– Started dev on filtering Wi-Fi data received and storing it in list of readings		- Figure out how to store in phone

12/01/17	– Learning how to store readings in phone – Best approach to use shared preferences		- Implement
14/01/17	– Able to scan for readings, filter and store in list. Then store into preferences from same activity (scan) – Trouble when getting results from scan and storing in prefs from update activity		- Figure out how to send data across activities - Store data in phone
18/01/17	– Now able to scan in one activity and pass results to another – Room Id taken as input from user – Location object created and stored in prefs using hashmap		- Need to retrieve and perform distance comparison
20/01/17	– Retrieving stored readings from prefs and displaying dara		- Distance calc using data
23/01/17	– Performs scan to get live data which is then stored in al of strings – Stored data is retrieved, each readings added to list		- Perform comparison
24/01/17	– Comparing data to find common mac addresses between live and instance of stored		- Use common to calc distance

27/01/17	– Start on Euclidean implementation – Getting sigs from each common mac and storing in array		-
28/01/17	– Performing Euclidean distance calc on Wi-Fi data in reading house – Printing distance to logs		- Select lowest and print to screen - Research android backends
29/01/17	– Selecting the lowest value distance – Can't print to screen, won't update		
31/01/17	– May need to use no-sql db to store data – Google app engine maybe able to host java programs		- Establish server
02/02/17	– GAE account created and test servlet launched on – Can access using a url		- Connect server to app and pass data
04/02/17	– Current readings being passed from app to GAE and printing out to server logs ○ Using OkHttp library – Mongo db account created using mlab		- Connect server to db - Store data between

05/02/17	– Issue connecting serv to db ○ Some issue with multithreading – Research gae and mlab, doesn't seem to be possible		- Find another approach for a backend
07/02/17	– Possible to use AWS elasticbeanstalk – Transferred servlets to there ○ Can't find logs		- Find log statements and test connection - Connect to mlab
08/02/17	– Connected to mongo db through mlab – Readings stored with loc id ○ Using json		- Perform prediction on server
09/02/17	– Model transferred from app to server in prediction class – Stored data being retrieved from mlab – Performing calc and returning roomID to app		- Test calculations to see if correct
12/02/17	– Calculations give correct results – Establishing fingerprint of house in Wiltshire ○ Single scan in each room		- Test predictions
13/02/17	– Mixed results across four rooms of house – Able to distinguish what part of house in – Floor is inconsistent		

17/02/17	– Tested with multiple reading points in one room ○ Much of the same		
18/02/17	– Took multiple readings from one position ○ Noticed difference in signals in one position ○ Might be effect of body in the way	See distance model paper in FYP folder	– Take multiple readings from one spot and average them
19/02/17	– Code to average multiple readings added to program		– test
20/02/17	– Tested using averaged readings in Wiltshire – performed considerably better across four rooms ○ not as well across six		– need to repeat in SSE ○ bigger building may have better results
23/02/17	– Established fingerprint in small section of SSE ○ G56 and surrounding area		– Test accuracy in SSE
27/02/17	– Tested readings in SSE – Prediction accuracy around between 40-50% ○ Appears to have element of randomness		- Research optimisations - Consider k-nn rather than where k>1

03/03/17	– Researched optimisations ○ 2 most important access points ○ Using all aps not just common ○ Setting a threshold	http://journals.sagepub.com/doi/full/10.1155/2015/429104	- Investigate if system would benefit
08/03/17	– Met with supervisor ○ Discussed methods to improve the accuracy ○ Discussed current progress and functionalities to be implemented ▪ Additional buildings not necessary ○ Focus must be on increasing the accuracy ○ Establishing UI with map ○ Need larger dataset ○ Will need to compare approaches in report		- Get more readings with greater coverage of the building - Explore accuracy
18/03/17	– Project poster completed		- Submit before 24th march
27/03/17	– Collected readings downstairs in SSE – Tested accuracy based on server outputs ○ Poor results		- Collect more readings - Investigate results
28/03/17	– Collected further readings – Confirmed accuracy issues are a result of only using common mac addresses in distance calc		- Collect more readings - Test whether using all would improve accuracy. - How would it be done

	○ Was possible and likely to have lower distance with fewer matching MACs		
29/03/17	– Implemented new approach which uses all mac addresses, if mac in live but not test set value to -120. ○ Results in greater distance		- Test new implementation
30/03/17	– Test new implementation which involves setting uncommon MAC to -120 ○ Tested across three rooms with 29/30 correct		- Need more readings - Decide whether I should reinstate averages - Test different classifiers - Add map to show location - Polish app (UI and handle crashes)
02/04/17	– Re-structured data so could be tested in KNIME – Tested in KNIME ○ 1-NN shows best classification results, compared in k-NN		- Work on UI
03/04/17	– Tidied up UI – Removed bugs causing app to crash – Added Logo		- Add location using map
04/04/17	– Map added to prediction activity ○ Image using glide lib		- Start report writing

08/04/17	– Test data from 07/04 analysed. – Tests in 177 failed because a new ap was introduced and aps signal strengths have been adjusted ○ 24:de:c6:26:d6:80 ○ Not present in 177 stored (part of older set) ○ Is present in 176 stored (part of new set) ○ Present in test from same day as 176		- Consider approaches to combat this.
11/04/17	– Met with supervisor ○ Happy with state of system ○ Happy with chosen report structure ○ Discussed points to include in report ▪ Adaptive alg to combat periodical changes in wifi data ○ Discussed presentation and demo briefly		- Finish writing current draft and upload to dropbox - Prepare presentation

11.3. Appendix C -Euclidean Distance Full Test Results

Actual	Predicted
43	43
43	43
43	43
43	43
43	43
56	56
56	56
56	56
56	56
56	56
45	45
45	45
45	45
45	45
45	45
167	167
167	167
167	167
167	167
167	167
165	165
165	165
165	165
165	165
165	165
141	139
141	141
141	141
141	141
141	141
139	141
139	139
139	139
139	139
139	139
177	177
177	177
177	177
177	177
177	177

11.4. Appendix D - Manhattan testing results

Actual	Predicted
56	56
56	56
56	56
56	56
56	56
43	43
43	43
43	43
43	43
43	43
45	45
45	45
45	45
45	45
45	45
167	167
167	167
167	167
167	167
167	167
165	164
165	164
165	164
165	165
165	165
177	176
177	175
177	177
177	177
177	177
139	140
139	139
139	139
139	139
139	139
141	141
141	141
141	141
141	141
141	141

11.5. Appendix E - Coverage Maps of both floors in the Polly Vacher building

kd016499- Individual Project Report

GRADEMARK REPORT

FINAL GRADE

/100

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

PAGE 56

PAGE 57

PAGE 58

PAGE 59

PAGE 60

PAGE 61

PAGE 62

PAGE 63

PAGE 64

PAGE 65

PAGE 66

PAGE 67

PAGE 68

PAGE 69