**Spot Finder Android app for Outdoor sports such as BMX and skateboarding**

**SEAN MOYLAN & SHANE MORAN**

BSc. (Honours) in Software Development

**Final Year Project**

Friday April 24, 2020

Advised by: Martin Kenirons

Department of Computer Science and Applied Physics

Galway-Mayo Institute of Technology



**Contents**

**Table of Contents**

1. **Introduction**
   1. About
   2. Context
   3. Objectives
   4. Overview
2. **Background**
   1. Collaboration
   2. Technology
      1. Android vs Ionic
      2. Android vs React Native
      3. Android vs Flutter
      4. Why Android?
   3. Server
      1. Flask
      2. Pros
      3. Cons
      4. Python Flask vs Python Django
      5. Python Flask vs PHP
   4. Database
      1. MongoDB
      2. MongoDB vs MySQL
      3. MongoDB vs Cassandra
      4. MongoDB vs Microsoft SQL Server
3. **Methodology**
   1. Project Management
      1. Agile
      2. Planning
   2. Design
   3. Testing
   4. Evaluation and Managing
4. **System Architecture** 
   1. Types of Technologies
      1. Server
      2. Connecting Android with the Database
   2. System Integration
      1. Sublime Text
      2. Postman
      3. Retrofit
      4. Types of Testing
         1. TDD
         2. BDD
            1. Comparison
         3. ATDD
            1. Pros
            2. Cons
5. **Results**
   1. Testing
   2. Problems during Testing Process
      1. Back-End Problems
         1. Learning Flask
         2. Learning MongoDB
   3. Screenshots
6. **Conclusions**
7. **Appendices** 
   1. GitHub
   2. Video Demonstration
8. **References**

# 1.Introduction

The idea of the project is based on the problem that BMX riders, skaters and other outdoor sports do not always have available locations for recreational use at convenient times and infrequently summer.

## **About this project**

**Abstract** Skaters, for the sake of innovation, transparency and citizen participation are interested in outdoors sports such as BMX riding and skateboarding. This innovative technology allows for the decentralization of pay to play organised sports, allowing citizens to interact in an immutable, safe, and reliable environment. Throughout this study we will see the problems detected during its deployment phase, the solutions adopted and the conclusions for its use recreationally and commercially. The result will be achieved with the elaboration of a proof of concept prototype.

The purpose of this Final Year Project is to build an application using a new technology that the developers have never used before. Therefore, it allows for research to be done, learn up and coming up with technologies and discover new ways of developing software. The front-end technology being used to create the mobile application is Android Studio. Android Studio was chosen because of its new and gives the opportunity to learn a new Mobile Application framework. It will be connected to a back-end server running on a MongoDB Flask Database. This application will be useful and easy to use for users who will beneﬁt from this. This way, the developers will be able to show they have worked and have a fully functioning app that will satisfy what was required from them. The technologies used will be explained and why they were used throughout. This document will contain steps taken in creating the project and they will be documented clearly. The set research and development will all be detailed and allow others to understand why speciﬁc software was chosen. It will in detail compare similar technologies that could have been used. These sections will demonstrate the similarities between speciﬁc frameworks helping to explain what technologies were needed for the development application. This project is implemented using technologies that have not been used together too often which may lead to some issues. This means any problems faced or certain aspects found to be new will be documented along with the development.

## Context

The context of this project revolves around having fun with action sports anywhere, wishing to ﬁnd out more spots to BMX or skate. Opening the app on the user’s phone, they can view the various spots via Google Maps. This application prevents the user getting lost and gives them accurate information about each location. The information on the application will be provided by the users, to provide variety. Reading the data and retrieving images from the database needs to be very fast, as any delayed hang in performance could lead to a bad user experience.

The project will be developed as an android application which provides users with images, information, and a map of the various spots. Users of the application can find their nearest spots easier in relation to their current location at any given time. When the user decides they wish to learn more information about a speciﬁc location, they can retrieve images and text about the spot on the application. Along with the mobile application a web application will be developed for use, to update, delete or add information to the spots database.

## Objectives

The project will require a number of objectives to be accomplished in order to provide a solution that works and is suitable for use by outdoor sports enthusiasts. [<https://github.com/seanmoylan/Applied_Project/issues>]

* A Mongo database will be used to store the text which appears in the application. This database will need to be setup and hosted in such a way that it can be accessed from clients through the android application.
* Pull information from the Mongo DB using ﬂask.
* A client mobile application will be the main product / asset for the project. This application will be able to locate the user via the GPS on their mobile device. The app will then show the user the locations of each spot. This will provide the user an idea of how far they are from the chosen spot and will allow them to see when they are near their destination. Each location is speciﬁed on the application as a list on the home page. Then chosen images and information will appear for each location.
* An additional requirement is to develop a web application for the use of the company to allow them to modify the information shown to the user. The web application is a simple website that can access the database to create, update and delete information from the database.
* Using the Google Maps application programming interface (API) within the mobile application to show the user their location along with the speciﬁc spots. Each spot is indicated by a marker in the map. When the user clicks on a speciﬁc marker, the name of the location appears as a label.

## Overview

An app developed for BMX, Skateboard and any other sports that takes advantage of the outdoors and streets to have fun or help them progress in their chosen sport. The app will allow the user to locate popular locations for use in these sports that other users have uploaded. This will allow people to travel all around the world and be able to quickly find the best spots to visit to BMX, skate ... etc [<https://github.com/seanmoylan/Applied_Project/blob/master/README.md>]

# 2.Background

## Collaboration

The idea came through Sean Moylan for making a spot finder app. This application will enable the easy accessibility of information for outdoor sports people. All images and information included in this application was provided by users. We began the planning stage of the project with an initial meeting with our mentor (Martin Kenirons). The initial meeting took place in GMIT in October 2019.

During the meeting, we discussed exactly what was wanted as an outcome of this project by Martin as set out in the objectives. Meetings took place once every two weeks from this point to mid-February, but due to the unforeseen circumstances surrounding the spread of the Coronavirus; we were unable interact in person and this led to meetings having to take place online via Microsoft Teams.

that in order to iron out all the details in relation to design, requirements and any problems regarding the project. From January the development of the project, during these meetings the developers produced a working version of the application. This allowed Martin to give immediate feedback and change any of the requirements. Martin began to give the developers the information needed for the application in December and January in order to begin population of the database.

## Technology

During the initial project planning, such frameworks that were considered included Android Studio, Ionic, React native and Flutter. But we chose android studio. Android studio provides a flexible Gradle-based build system. It builds variants and multiple APK generation. It has expanded template support for Google Services and various device types. It provides a rich layout editor with support for theme editing. Line tools to catch performance, usability, version compatibility, and other problems. It has ProGuard and app-signing capabilities. Its built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine. [<https://stackshare.io/android-studio>]

The back-end services researched include Docker, MySQL, MongoDB, Cassandra, Flask, NodeJS and Django. After much research and reviewing of examples Flask and MongoDB were chosen.

### Android vs Ionic

Android Studio and Ionic are similar in what they oﬀer with regards to prebuilt components in Android Studio and a comprehensive suite of built in widgets in Ionic [<https://developer.android.com/guide/topics/ui>]. With Ionic, you create a real native app but you do this by creating a web app (with Hyper Text Mark-up Language, JavaScript and Cascading Style Sheets) which will be wrapped by a real native app that hosts a web view while Android studio you tend to write Java (or Kotlin) code which can be compiled on the target device. The main reasoning for using Android studio over Ionic is that it is reliable and resourceful. It is powered by Android and therefore is well documented and although it’s still relatively old there are loads of interactive talks online about its upcoming, current and new feature. Ionic has numerous third-party built-in tools. It has thousands of threads on Stack overflow and packages on NPM (node.js package manager) to help developers along the way.

### Android vs React Native

React Native is similar to Android development and compiles to native application by default. A basic React Native application is given a basic set of components. The developer must style most of them separately for each platform. This creates more work for the developer and increases time of development and cost of development for a company. React Native is developed by Facebook, while Android was various developers, but most notably Google. Each company is well developed and looked upon favourably by smaller businesses. React Native applications are developed using JavaScript and React libraries to build user interfaces. React previously existed to create web applications. Android is developed using various programming languages such as Java, C, C++, Kotlin and is solely used for mobile application development. Android is a very popular platform with over 980,000 results on GitHub in comparison to 145K for React Native. This is because Android is older and developed by Google, therefore it is becoming more and more easier to develop in each day.

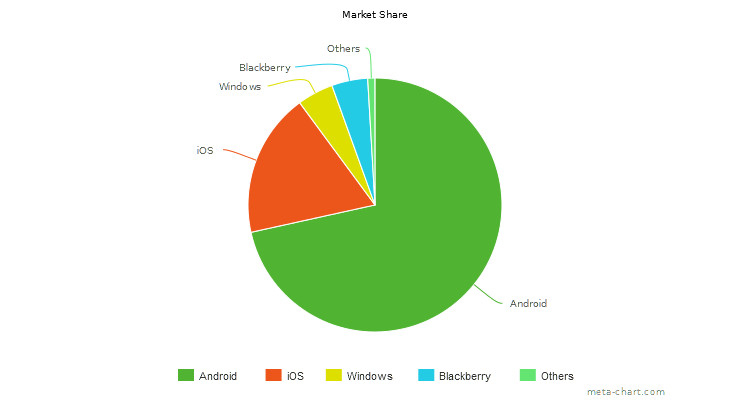
### Android vs Flutter

The original reason for using Android was the fact that Flutter is so new and the idea and challenge of learning a new cross platform mobile application platform did not sound appealing. Also, learning a programming language (Dart) with less resources to that of Android, would not be ideal. Android provides you with the API libraries and developer tools necessary to build, test, and debug apps for Android. It also provides a rich application framework that allows you to build innovative apps and games for mobile devices in a Java language environment. Flutter is Cross-platform mobile framework from Google as well and is a mobile app SDK to help developers and designers build modern mobile apps for iOS and Android. Android SDK belongs to "Frameworks (Full Stack)" category of the tech stack, while Flutter can be primarily classified under "Cross-Platform Mobile Development". "Android development" is the top reason why over 280 developers like Android SDK, while over 13 developers mention "Hot Reload" as the leading cause for choosing Flutter. Flutter is an open source tool with 69.5K GitHub stars and 8.11K GitHub forks. Here's a link to Flutter's open source repository on GitHub. [<https://stackshare.io/stackups/android-vs-flutter>]

### Why Android?

We opted to develop an Android app for many reasons, other than the fact that it has the most resources. After a good deal of researching other options of Mobile application development, Android came out on top by far. Firstly, the main reason is the use of java as a programming language makes it easy to port the app to multiple operating systems like Symbian and Ubuntu. Thus, businesses can target multiple platforms with Android app development. Android apps have a rapid development cycle lasting a few hours. It offers a competitive edge to companies who wish to have a quicker go-to-market for their new idea. Reduced Time to Market (TTM) is, thus, one of the best benefits of Android development.

Another one is its availability of the Android SDK. The development teams can use the material design from these SDKs to build interactive apps. However, developers are required to pay a one-time registration fee for application distribution. After that, they can leverage any computer device to build and test the product for their smartphones, ensuring low investment and increased user engagement. In turn, the end users, are benefited by an interactive app, and the enterprise gains a higher return on investment. Android P introduced several additional and in-built security features. It will help with the protection against malware and viruses. Thus, safety and reliability are exceptional benefits of android application development. To conclude, with over 75% Android device users today, developing an app on this platform is a value proposition for organizations globally. It helps them address a higher range of audience and gain control over the market. [<https://www.rishabhsoft.com/blog/5-advantages-of-android-app-development-for-your-business>]



## **Server**

### **Flask is a micro web framework written in Python. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website. Micro-framework is normally framework with little to no dependencies to external libraries. This has pros and cons.**

### **Pros would be that the framework is light, there are little dependency to update and watch for security bugs,**

### **Cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins.** **[**<https://pymbook.readthedocs.io/en/latest/flask.html>]

**It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks. Flask offers suggestions but does not enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.** **[**<https://palletsprojects.com/p/flask/>]

Python Flask vs Node.js

Choosing a language back end is an important step to the development planning phase of any project. While ﬂask is written is Python, node.js is JavaScript which allows JavaScript run on the server side and not in the browser because it is an environment. Flask is a micro framework for Python based on Werkzeug, Jinja2andgoodintentions. The comparison of Flask and node.js breaks down to a comparison between Python and JavaScript. The performance at the back end of any application is important for the speed and response time of the application. Both ﬂask and node.js have good performance however node.js has better performance this is because it is based on chrome V8, a fast and powerful engine. Flask is suﬃcient once application is not memory intensive. Scalability of the application when running on node.js is optimum as node.js supports an asynchronous application. This ensures for smooth, fast scaling. Python’s Flask does not support traditional asynchronous programming, but this can be mimicked with coroutines, making asynchronous processing achievable.

### Python Flask vs Python Django

Django is a full-stack web framework for Python, whereas ﬂask is a lightweight and extensible Python web framework. Django is focused on the ﬁnal product alone, it is an all-inclusive experience. Flask focuses on the experience and the learning opportunities. It provides simplicity, ﬂexibility and ﬁne-grained control. Although Django comes with everything built in, it is diﬃcult to change predeﬁned platforms, libraries etc. Although Django was released ﬁrst in 2005 in comparison to Flask in 2010 the latter has outgrown Django and is a more popular framework today. Although both are easy on the eye, some would say that Django has underpowered templating and Object-Relational Mapping.

### Python Flask vs PHP

Another alternative was the popular general-purpose scripting language that is especially suited to web development, known as PHP. Its Fast, flexible, and pragmatic, PHP powers everything from your blog to the most popular websites in the world. But its inconsistent API, fragmented community and lack of a routing system lead to us choosing Flask. [<https://stackshare.io/stackups/flask-vs-php>]

## **Database**

### **MongoDB**

MongoDB is one of many non-relational databases. It is a NoSQL database that is used across many areas in the technology and business sector. MongoDB is a free open source database management system. As it has evolved over the years it is now one of the most popular document-oriented databases. It is run on a mongo server and can be created and controlled from your command prompt. Data in MongoDB is stored in JSON like documents. A format called BSON which is JSON in binary style issued for document storage in the database. This allows for an easy to read format of the data. Dissimilar to relational databases for example MySQL which uses rows and tables as a database structure MongoDB is formed of collections and documents. Each database in mongo can have multiple collections. ” A collection is a group of documents”. These documents can have many ﬁelds. The Documents in the database are a set of key value pairs. An id will be assigned to the documents making your database easy to edit. As MongoDB is a NoSQL non-relational database it diﬀers in many ways from a relational database management system. A relational database is row and column based as opposed to document and ﬁeld based. MongoDB is considerably easier to set up and is not vulnerable to SQL injection. A RDMS can be more challenging to understand and lets down the database in terms of its hierarchical storage not being as good and how it is open to SQL injection. One of the more popular relational databases is MySQL. Compared to MongoDB, it is quite inﬂexible in terms of the database schema. MongoDB is known for its ability to” handle large amounts of unstructured data” which MySQL lacks in its technical features. Improving databases in terms of storage and their schemas can help to build more eﬃcient and easier to understand technologies. MongoDB runs better than many other database systems used today. It is a high-quality technology which has features that are desirable in having a secure database. Known for its scalability, memory processing and concurrency MongoDB makes for a desired database. All these features make it easy to develop a fast, reliable database which makes it work alongside evolving applications and is sought after by businesses. It is ﬂexible and can be adapted to work with many platforms and can be integrated into software much easier than other database management systems. The most common frameworks that are used in conjunction with MongoDB are NodeJS and a newer one Mongo Flask. It is also supported by cloud services such as Amazon Web services and Google cloud platform.

### MongoDB vs MySQL

There are ultimately two types of databases. These include SQL database and NoSQL databases. MySQL is an example of a SQL database while MongoDB is an example if a NoSQL database. MongoDB is stored in a document like json. It is ﬂexible and dynamic because the structure can be changed to meet customers’ needs and can be scaled horizontally. SQL is written in SQL query language and can be scaled vertically. MongoDB is a much newer server and was released in 2009 vs SQL server which has been published since 1989. MySQL had document orientated structure model while SQL is a relational database management system (RDBMS) model. Joins, concurrency and foreign keys are not supported by MongoDB - only SQL databases. MongoDB’s are created and maintained in an agile development practise while SQL databases are supported mainly by waterfall life cycles practises. This makes MongoDB’s more popular with modern companies who use agile development and many older companies are now switching to use MongoDB and agile also. Data Schemes are Dynamic in MongoDB but static/ﬁxed in SQL databases. MongoDB’s are more usable and ﬂexible as they can be run on Windows, Linux and os X operating systems in comparison to MySQL which can only be run on Windows operating systems.

### MongoDB vs Cassandra

Both MongoDB and Cassandra are NoSQL databases. Large companies such as Facebook use both databases. Cassandra can handle large amounts of unstructured data e.g. Instagram has about 80 million photos uploaded daily to its Cassandra database. MongoDB, because it is schema-free, documents can be created without creating their structures ﬁrst, in comparison to structures having to be predeﬁned in Cassandra databases. Cassandra databases are queries with CQL querying language. This is similar to SQL querying language. MongoDB currently had no support for any querying languages. MongoDB has queries which are structured as JSON fragments. Cassandra was released in 2008 by Facebook but currently being maintained by Apache software foundation. This is a newer NoSQL database than MongoDB and already has become more advanced and popular among large companies.

### MongoDB vs Microsoft SQL Server

Our final option was a relational database management system developed by Microsoft, called Microsoft SQL Server, and is a database management and analysis system for e-commerce, line-of-business, and data warehousing solutions. Whereas MongoDB is the database for giant ideas. MongoDB stores data in JSON-like documents that can vary in structure, offering a dynamic, flexible schema. MongoDB was also designed for high availability and scalability, with built-in replication and auto-sharding. Microsoft SQL Server and MongoDB can be primarily classified as "Databases" tools. "Reliable and easy to use", "High performance" and "Great with .net" are the key factors why developers consider Microsoft SQL Server; whereas "Document-oriented storage", "No SQL" and "Ease of use" are the primary reasons why MongoDB is favoured. MongoDB is an open source tool with 16.3K GitHub stars and 4.1K GitHub forks. Here's a link to MongoDB's open source repository on GitHub. Microsoft SQL Server also had expensive licensing which there were cheaper alternatives for. [<https://stackshare.io/stackups/microsoft-sql-server-vs-mongodb>]

# 3. Methodology

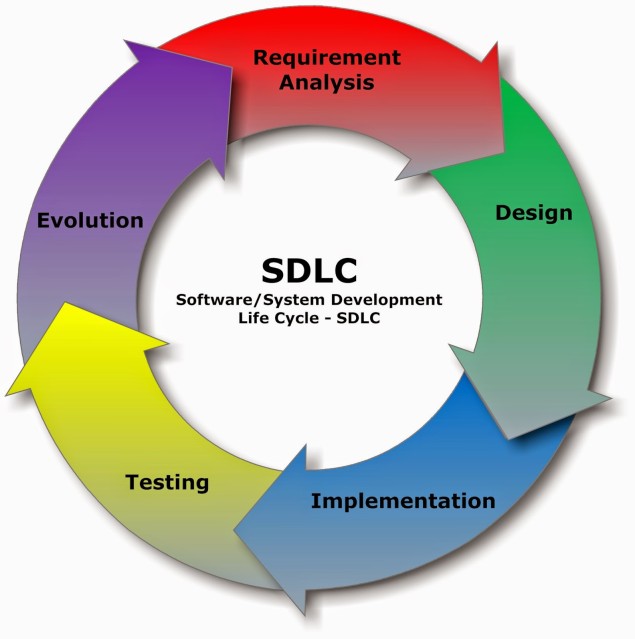
## Project Management

Project Management is key element to ensure that the project is laid out correctly and each component part is complete at a given date etc. Because this project was completed by a team it was important both members completed certain aspects of the project.

In order to manage this project, GitHub was used as it was very useful due to it being a team project. It enables both members of the team to work on separate individual branches. After each iteration, these branches were merged into the master branch. After each integration, the project was at a working state of the project. This could be used for industry standard as continuous delivery. For this project it meant that after each Agile sprint there was a working version to present to customer and supervisor. Although it is a working version is may still need changes in the next sprint. At the beginning of each sprint all branches pull from the master so as every team member is working oﬀ the same latest release of code.

### Agile

The project used Agile project management methodologies. An iterative approach was taken. This meant that each week certain tasks are completed. The project team met once a week to discuss what has been completed since the last meeting, what must be completed before the next and discuss any difficulties experienced throughout the duration. The meetings are short and concise. Once every week to two weeks, the team met their mentor and explained what was completed since the last meeting and is to be done before the next. These weekly meetings were the iterative approach. Each week the process would be repeated to meet, plan, design, develop, test, and evaluate.



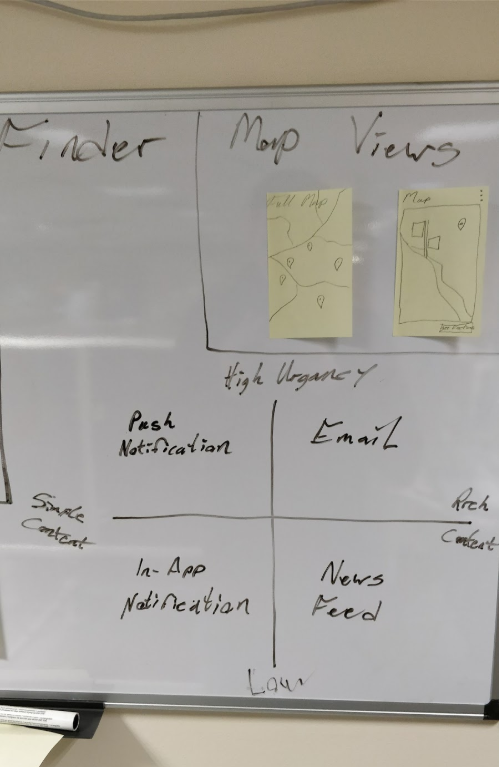
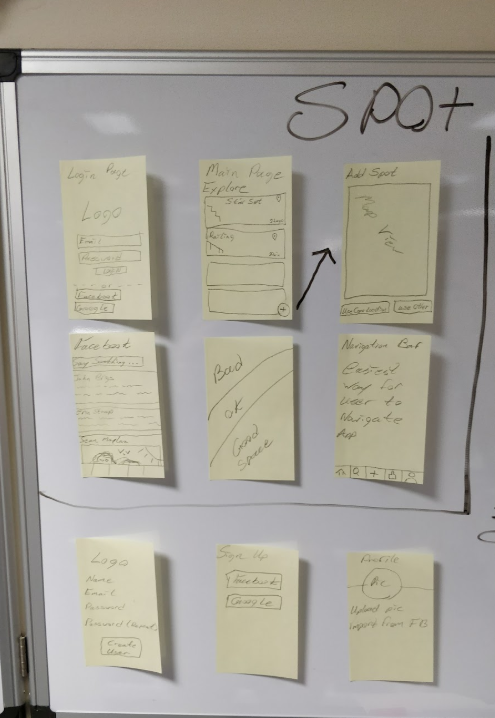
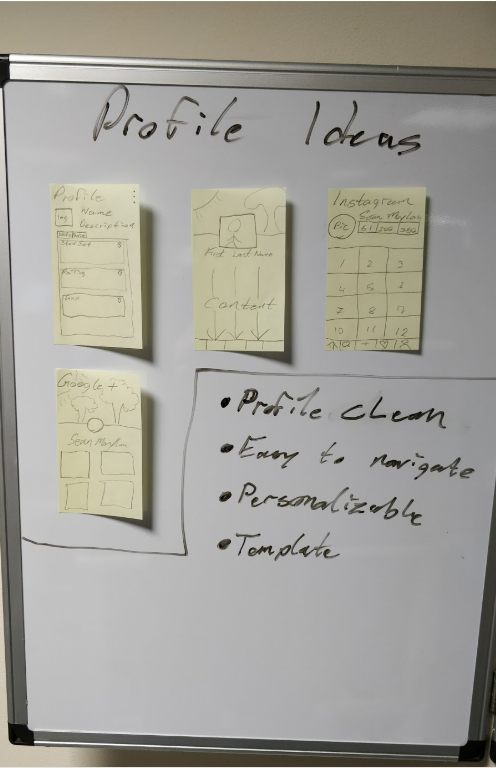
[This Photo](http://wccftech.com/jack-software-developed-part-ii/3/) by Unknown Author is licensed under [CC BY-SA-NC](https://creativecommons.org/licenses/by-nc-sa/3.0/)

### Planning

During the meeting and planning phase, the milestones for the project were identiﬁed and broken down into simpler, more manageable tasks. These are the tasks which must be complete before each meeting and can be found in the issues section of the GitHub repository. Each sprint started after the weekly meeting with the supervisor and the aim is for the work to be complete before the meeting the following week.

## Design

Throughout the agile development life cycle design is a step in every iteration. The design is gradually built on and is never deﬁned at the beginning of the project. The gradual evolution of the design enables taking advantage of new technologies that come on stream as well as meeting any new requirements brought forward by users. The user experience is key when designing an application and to make sure of that, it was to be made user friendly. The type of end user of the item must be considered when designing.



Project Design

## Testing

Testing was carried out continuously. When tasks were complete that were stated in the test plan the tests were carried out. If possible, user tests were carried out when meeting with the supervisor. The supervisor used the application and commented on any changes that should or could be made for the next meeting.

## Evaluating and Managing

After each sprint before the meeting phase a review was carried out on the tasks completed in the last sprint. If there were changes to make these were completed in the next sprint. In the evaluation section of the agile development cycle the code by both developers was mergers into the master branch of the GitHub repository. The evaluation process is also used to look back, see how much of the previous iteration that was not complete and change the workload applied for a single iteration. This could also apply at an industrial level where continuous delivery is needed. After each commit, a working version was present for our supervisor. But this might mean that future changes are still forthcoming.

# 4. System Architecture

All static text is stored in the MongoDB hosted at http:// . This information is accessed by the Android application though the Python flask server which hosts the Mongo database. Each row of the database is called individually to display the appropriate information, update, and delete a different spot. When creating a spot in the database, a new row is added to the end of the database for each entry. The web application both sends and receives information to and from the database while the Android application only receives information from the database.

## Types of Technologies

### Server

Mongo and Flask To develop the back-end server for the project it was completed using the web framework Flask. It is coded in Python and connected to the MongoDB database. The database was setup in MongoDB using atlas. The MongoDB database contains a collection called info. The collection then is made up of documents called id and description. There is an id and description for every location and piece of informative text about the location in the app. The description contains the long pieces of text in the database. Each id is intended to have a description. The Flask web framework that was set up was launched on a virtual machine. Flask was used as a web framework in this case to allow access to the database and display and edit its contents. It can be viewed externally on its own IP address that was created from the VM “”. This was achieved by connecting MongoDB and Flask to implement CRUD operations. The ﬂask server is run from the ‘FlaskServer.py’ ﬁle. This ﬁle contains the route to show the running server. The Flask server was then connected to MongoDB. The PyMongo api allowed the connection between these two frameworks. PyMongo is imported in Flask. In the FlaskServer.py ﬁle, it calls the Mongo URI. This allows it to recognise the database. The data types stored in the database are converted to Json string also in this ﬁle. This allows readable interpretation of the data. The CRUD implementation in the Flask allows speciﬁc routes to access the database. The collection ‘info’ is referenced to ensure the correct data is being accessed. The CRUD functionality is implemented in the Flask through methods Get, Post, Delete and Patch. The Get method does a query to ﬁnd the id and description. This is accessible through the route data. The IP address allows the route to be added to it “’’. This IP address will display the contents of the database in JSON string format. This is the read functionality in the CRUD working using the speciﬁc method. These methods are programmed in the ‘users.py’ ﬁle. The data can then be accessed in the Android application and read in to display the data by referencing the IP address. The ﬂask framework lets the data be accessed eternally from a virtual machine, so the data is secure and will not be lost. This framework is used for the create update and delete to allow the data to be edited by the administrator. This is again accessed by the IP address and the look front-end is completed in the Login.java. This references the users ﬁle and the methods that allow the queries to delete add and update along with the read which is also used in the Android application.

#### Connecting Android with the Database

The information displayed in the Android Application is being stored in the database and not the app itself. The text information is being read into the app from the database, so it is more secure. This was done so the application does not contain large amounts of text. This allowed testing to be done to see how Android works with reading from a database as it is a modern technology that has been used in the app development sector. Due to androids’ vast libraries and amount of open source code, little research and testing many diﬀerent URL readers to fetch the data and be able to display it in the correct way. The database was changed many times. In the most recent database, it contains a name, id, and a description. Each id is in its own document and is numbered in the database. This allowed for each id to be read in separately so the exact text could be speciﬁed in the application.

This database connection is done in Location.java, Login.java, Tools.java and User.java. The same string readings are done in the classes. The URL is read in as a string. The URL is of the Data that is displayed in a JSON format when you search the URL. A method is then used to read any data that is in JSON format. This then will use a set state in the application that reads the data contained in the ‘multi’ document that is in the database. The set state is used to rebuild what the user calls inside it in Android. This means all the data is read in the app to the console. This also decodes the JSON to be able to display it as plain text. To display the text in the app it is completed in the widget. Android uses a list view builder. For the data that has been read in to be transferred to the list display an item count is used. This reads the data variable from the json data and add the length of the database. It has been changed to one in this case, so it only reads one piece of information when it is called. A container in the app is what displays the text. The variable is called and has an int and the name of the ﬁelds in the database being used. The int is referencing the number each document in the database is given. When you open the app, there is . This can be seen in the container of the class, it calls and description so will display the correct piece from the database.

\*\*insert screenshot of homepage\*\*

To use the database in an eﬃcient way with Android the way the text was being called was changed multiple times. With the research involved the only way it seemed possible to display the text was in a list view. This meant there had to be an integer used in calling the speciﬁc pieces of information. This was as the item builder contains the Build context and an int. Without the list view builder an item builder was not allowed. When the data variable was used with a ﬁeld from the database it could only print one description as all the data in the collection uses description ﬁelds. The data variable had to be used so the text was not being displayed as JSON text. The data variable could not be used with two ﬁelds being called as only one is used in the set state method. This meant this was the only way the data could be displayed in a correct way that suited the database. The containers mean the text is displayed in card like paragraphs that allow for an easy to read display in the application. The text documents and pictures will ﬁt nicely beside each other and the text can be aligned nicely.

#### Web App

As part of the project, we felt the web application’s functionality should at least support basic CRUD operations which has been developed for the administrator to edit the data. The Flask web framework was set up for running the database and to have a list of options available for the administrator to change the database. They would have been able to add a name, id, description to the database and edit the information that is in the database by using the id ﬁeld. There would also be an option to input the id the administrator would want to delete; a remove button would then remove the speciﬁc ﬁeld of the id you were about to enter. Methods were used to control these in the ”FlaskServer”.py ﬁle. The Post, Get, Delete and Patch were used. To view the page, you can use http:// view. To view the database, you can click the view button, and this will bring you to the database. It is only viewed as JSON. This is a URL link in the HTML page. It references the user deﬁnition This displays the database using a get method. This uses a Mongo query to ﬁnd the information. The delete function is also working, this is using a post method to delete the information. The delete is done through a MongoDB query in Python code. The edit button is on the page as it is a html page calling the URLs. HTML pages are used to display the information and link the buttons to the deﬁnitions which are in the Python ﬁle.

### Client

#### Mobile App

Android studio was used to develop the mobile application intended for Android devices. Research was carried out to learn java programming language and learn the syntax of Android along with understanding how widgets work correctly. When creating the application, initially all pages were split up into two classes. This made the code very long and caused a lot of confusion. At this stage the developers decided to break up the code into a class for each location or spot. The main page holds the home page of the application along with the splashscreen of the application. The spashscreen “widget” has a “” method. This method calls the timer function which is an async method.

The maps page contains all elements for the Google maps used within the application. The map view library is imported at the beginning of the ﬁle. This enables the creating of a Google maps image in the application. This Google maps image however is places on top of the widget. This is a key feature in the development of the project. The map view is a separate widget that in linked to directly from Google maps api. This is called from within a stateless widget. When the back button is selected the used is returned to a empty page. For user friendliness a message to go back one more page is presented to the user. Therefore, a speciﬁc location stands out on the map when clicked, and a label appears over the location if visited; if not, they have the option add a spot. The “” method takes the parameters (String name, String id, double longitude ,double latitude).The maps page is called by each of the location pages.

Each page containing the locations of the spots are laid out in similar format. One of which page contains multiple photos in a scrolling slideshow. This user must select previous/next button to navigate through the images. The images are stored in a list and list is controlled by a photo Index to see what number of the list is currently being displayed. The text and URL location for each page is taken directly from the MongoDB Flask database.

## System Integration

### Sublime Text

This popular cross-platform text editor with a Python interface was used to code the server which can found in the FlaskServer.py file and is started via the terminal using “export FLASK\_APP=FlaskServer.py”.

### Postman

Postman [<www.postman.com>] then tests the requests from the server. Before testing, an interface of requests must be created. Then, we can call them from whatever activity the information is needed

### Retrofit

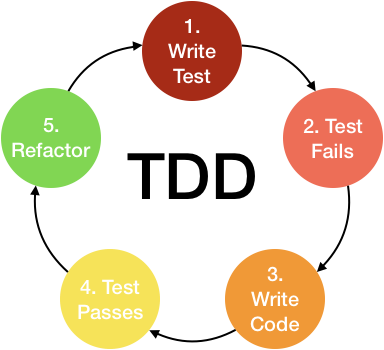
Firstly, what is Retrofit? Retrofit is a REST Client for Java and Android. It makes it relatively easy to retrieve and upload JSON (or other structured data) via a REST based webservice. In Retrofit you configure which converter is used for the data serialization. [<https://www.vogella.com/tutorials/Retrofit/article.html>] Why it was used? Once the requests worked, I used to retrofit to make calls to the server from the app.

### Types of Testing

#### Test Driven Development

Test-driven development is an evolutionary approach which combines test-first development where you write a test before you write just enough production code to fulfil that test and refactoring. What is the primary goal of TDD? One view is the goal of TDD is specification and not validation. In other words, it is one way to think through your requirements or design before your write your functional code. Implies that it is both an important agile requirement and design technique. Another view is that TDD is a programming technique.

[Beck, K., 2003. *Test-driven development: by example*. Addison-Wesley Professional]



Before writing implementation code, the developer writes automated unit test cases for the new functionality they are about to implement. After writing test cases that generally will not even compile, the developers write implementation code to pass these test cases. The developer writes a few test cases, implements the code, writes a few test cases, implements the code, and so on. The work is kept within the developer's intellectual control because he or she is continuously making small design and implementation decisions and increasing functionality at a relatively consistent rate. New functionality is not considered properly implemented unless these new unit test cases and every other unit test case written for the code base run properly.

Some professed benefits to TDD are:

* In any process, there exists a gap between decision (design developed) and feedback (functionality and performance obtained by implementing that design). The success of TDD can be attributed to reducing that gap, as the fine granular test-then code cycle gives constant feedback to the developer. An often-cited tenet of software engineering, in concert with the "Cost of Change" [<http://www.pauldee.org/se-must-have/boehm-SE-Economics.pdf>] is that the longer a defect remains in a software system the more difficult and costly it is to remove. With TDD, defects are identified very quickly, and the source of the defect is more easily determined.
* TDD tempts programmers to write code that is automatically testable, such as having functions/methods returning a value, which can be checked against expected results. Benefits of automated testing include the following:

1. Production of a reliable system
2. Improvement of the quality of the test effort
3. Reduction of the test effort
4. Minimization of the schedule.

* Test cases create a thorough regression test bed. By continuously running these automated test cases, one can easily identify if a new change breaks anything in the existing system. This test bed should also allow smooth integration of new functionality into the code base. With XP, developers do little or no up-front design before embarking in tight TDD cycles consisting of test case generation followed by code implementation. However, many of the benefits listed above can be realized in essentially any development process simply by shifting from unit test after implementing to unit test before implementing.

[“Assessing test-driven development at IBM” EM Maximilien, L Williams

25th International Conference on Software Engineering, 2003. Proceedings. - 2003

ieeexplore.ieee.org <https://dl.acm.org/ft_gateway.cfm?id=776892&type=pdf>]

#### Behaviour Driven Development

Behaviour Driven Development is an agile software development approach that encourages collaboration between all project participants. It is an evolution of Test Driven Development [Section 4.2.1] and Acceptance test-driven planning [Section 4.2.3] . The most important core principle of Behaviour Driven Development is that “business and technology people should refer to the same system in the same way” [<http://behaviour-driven.org/>].

In order to achieve this objective, a common Language is needed for specifying system behaviours, allowing customers to specify requirements from a business perspective, businesses analysts to attach concrete examples clarifying the system behaviour, and developers to implement the required system behaviour in a Test Driven Development manner.

The second core principle of BDD says that “any system should have an identified, verifiable value to the business” [<http://behaviour-driven.org/>]. Organizing the development effort around the system behaviour could achieve the first objective, but how could we know when we have delivered a behaviour? If the behaviour is described using executable scenarios (executable acceptance tests) then the software can be automatically verified through successful passing of the tests.

When Dan North introduced this concept, he proposed an ubiquitous language for analysis, so that the requirements can be captured into the codebase. This language represents the requirements as user stories, and the acceptance criteria as scenarios attached to user stories. He proposed the following standard form for writing scenarios: given some initial context, when an event occurs, then ensure some outcomes.

##### Comparison

TDD is a test of component in isolation and is an example of Unit testing, whereas BDD is a system test. It is also a behavioural test.

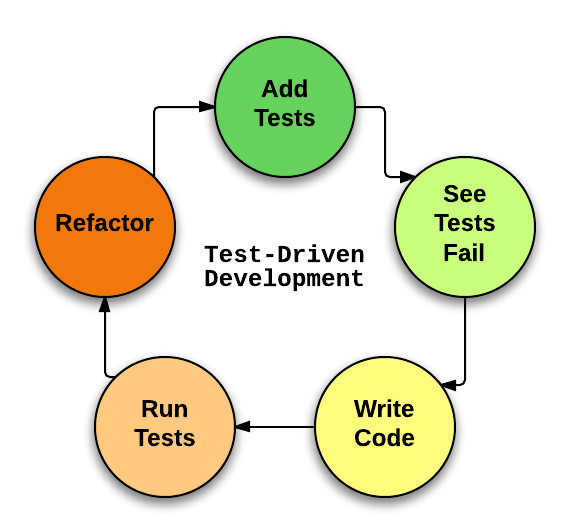
While BDD focuses on the value, TDD cares about the quality. In TDD, tests are written at the same time as the code; so, the developer writes and reads the tests, the testers also read the code. Whereas in BDD, the person that understands the customer best writes the tests and almost anyone can read the tests, such as developers, testers, stakeholders, business and product owners. With Unit tests, you know specifically what did not work unlike BDD where you don’t. Unit tests tend to be quicker, due to behaviour tests being done as a whole. Although, Behavioural Tests tend to be high, but 100% code coverage is not uncommon. Regarding maintenance, Unit tests change first; whereas with BDD, the high-level nature of Behavioural Tests mean that they change infrequently. BDD testing is portable unlike Unit tests as it not coupled to the code.

[D. T. Pays, “Test driven development vs behaviour driven development + free cheat sheet.”]

[<https://www.developmentthatpays.com/files/DevelopmentThatPays-TDDvsBDD-CheatSheet-1_5.pdf>]

#### Acceptance Test Driven Development

Acceptance Test Driven Development involves team members with different perspectives collaborating to write acceptance tests in advance of implementing the corresponding functionality. The collaborative discussions that occur to generate the acceptance test is often referred to as the three amigos, representing the three perspectives of customer (the problem needed to be solved), development (how we solve the problem), and testing. These acceptance tests represent the user’s point of view and act as a form of requirements to describe how the system will function, as well as serve as a way of verifying that the system functions as intended. In some cases, the team automates the acceptance tests.



##### Benefits

Just as TDD results in applications designed to be easier to unit test, ATDD favors the creation of interfaces specific to functional testing. (Testing through an application’s actual UI is considered less effective.)

##### Pitfalls

Even more than the use of automated acceptance tests, this practice is strongly associated with the use of specific tools such as Fit/FitNess, Cucumber, or others.

One major risk, therefore, is that the tool chosen will hinder rather than advance the main purpose of this practice: facilitating conversation between developers and product owners about product requirements. Tools should be adapted to meet product owners’ needs rather than the other way around. [<https://www.agilealliance.org/glossary/atdd/>]

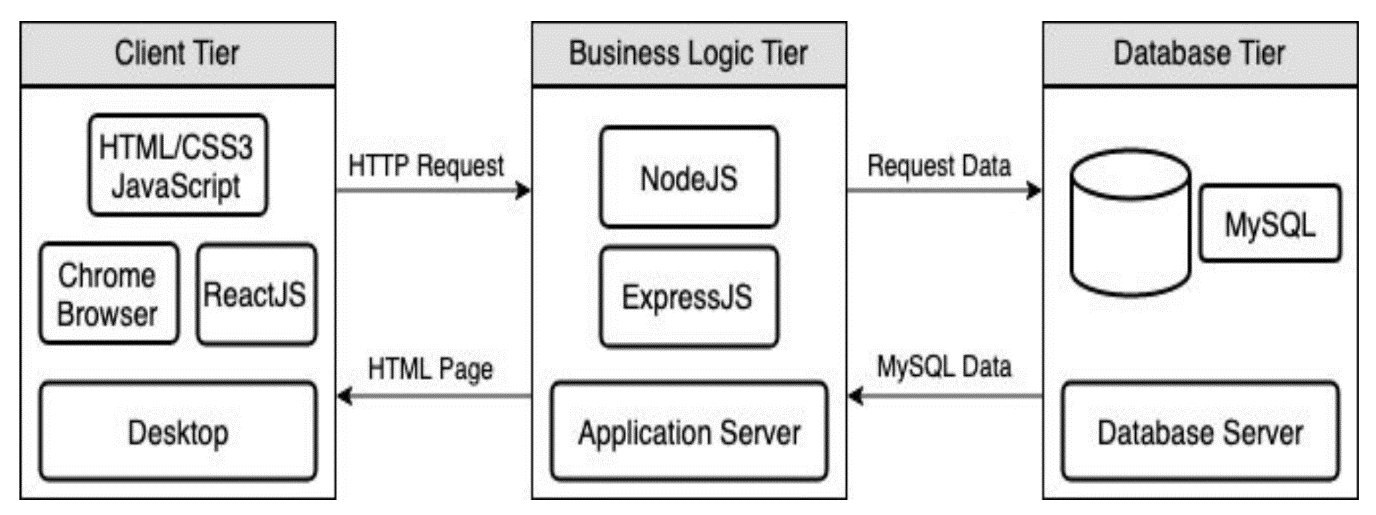
# 5. Results

Since the first meeting with our supervisor back in October, we can say that we have set met the objectives set out for this project; some of which can seen above. [1.2. Objectives]

## Testing

Firstly, we chose our technologies that would be used; Therefore, we chose the appropriate three-tier architecture. We needed a backend server, a database, and an IDE or framework to develop the app.





3-Tier Architecture

Then, we had to decide over which parts of the project each individual would be responsible and what technologies would be implemented. Initially, regarding the server and database, we had alternatives we were think of using, but opted not to. For example, we thought about using MySQL as our database and Django as our server.

## Problems during Testing process

### Back-End Problems

The server and database were first set up using firebase and then opted for Flask and MongoDB. The database would contain an id and a description, at first. The html webpage was run locally via Flask. It was also accessible externally by an IP address. The ports for Flask were open so it was accessible remotely; these ports were ports “”. The page displayed just some plain text in a header. The MongoDB was then connected to Flask and it used Mongo’s URL and default port (27017) to connect and run it from the browser. When run from the command line it was displayed that the JS and Index were connected to the database.

#### Learning Flask

Learning Flask from the ground upwards was the one of the first steps and challenges we faced, but, being written in python and having lots of online resources available to get started made it that much easier.[<https://flask.palletsprojects.com/en/1.1.x/>]

#### Learning MongoDB

Having studied databases such as MySQL and SQL , we knew that connecting the Flask Server to MongoDB using a RESTful api would be no easy task at first. First off, we had to set up an atlas account and get started by learning the basics, what type of database we would working with and would it suit our app. [<https://docs.mongodb.com/manual/tutorial/getting-started/>]

## Screenshots

\*\*\*Add Screenshots of each page\*\*\*

# 6. Conclusions

# 7. Appendices

## GitHub

<https://github.com/seanmoylan/Applied_Project>

## Video

\*\*Insert Video Link\*\*

# References

1. Objectives [<https://github.com/seanmoylan/Applied_Project/issues>]
2. Overview of Project [<https://github.com/seanmoylan/Applied_Project/blob/master/README.md>]
3. Android studios pros and cons [<https://stackshare.io/android-studio>]
4. Android Developers guide [<https://developer.android.com/guide/topics/ui>]
5. “Android SDK vs Flutter” [<https://stackshare.io/stackups/android-vs-flutter>]
6. “How Benefits Of Android App Can Help Businesses” [<https://www.rishabhsoft.com/blog/5-advantages-of-android-app-development-for-your-business>]
7. **“Introduction to Flask “ [**<https://pymbook.readthedocs.io/en/latest/flask.html>]
8. **Flask Example [**<https://palletsprojects.com/p/flask/>]
9. Flask vs PHP [<https://stackshare.io/stackups/flask-vs-php>]
10. Microsoft SQL Server vs MongoDB [<https://stackshare.io/stackups/microsoft-sql-server-vs-mongodb>]
11. Download Postman [<https://www.postman.com/downloads/>]
12. “Using Retrofit 2.x as REST client – Tutorial , Lars Vogel, Simon Scholz, David Weiser“ [<https://www.vogella.com/tutorials/Retrofit/article.html>]
13. “Software Engineering Economics” [<http://www.pauldee.org/se-must-have/boehm-SE-Economics.pdf>]
14. [“Assessing test-driven development at IBM” EM Maximilien, L Williams 25th International Conference on Software Engineering, 2003. Proceedings. – 2003 ieeexplore.ieee.org <https://dl.acm.org/ft_gateway.cfm?id=776892&type=pdf>]
15. [<http://behaviour-driven.org/>].
16. TDD vs BDD Cheatsheet by Development That Pays, [<https://www.developmentthatpays.com/files/DevelopmentThatPays-TDDvsBDD-CheatSheet-1_5.pdf>] ,

YouTube video: [<https://www.youtube.com/watch?v=4QFYTQy47yA>]

1. Acceptance Test Driven Development (ATDD) [<https://www.agilealliance.org/glossary/atdd/>]
2. “Flask User’s Guide” [<https://flask.palletsprojects.com/en/1.1.x/>]
3. Getting started with MongoDB [<https://docs.mongodb.com/manual/tutorial/getting-started/>]
4. GitHub repo [<https://github.com/seanmoylan/Applied_Project>]