Advanced Software Engineering Team 3 Report

In this Advanced Software Engineering coursework assignment, our team was assigned a series of tasks that completed in sequence gradually progressed towards an android app that displays the Land Registry Price Paid data for nearby properties on a Google Map fragment. This report chronicles, in our own words, the work that each individual contributed toward the completion of the project in its final state. Below you will find approximately five hundred words from each team member describing the challenges, triumphs and lessons that each one of us encountered during the course of completing our sections of the work. First off, Grant will describe the development of the Front End in Android Studio. Following this, Dan will describe the development of the Back End in Java Netbeans, before Mehmet describes the Database Systems initially in Java Derby and finally in Heroku and PostGre SQL. To finish everything off, Alex will describe her efforts to test the system.

The project’s front end is an Android application running at API level 24. The application employs Google locations services to fetch and display location coordinates (latitude and longitude) for the user; as well as plot both the user’s location and nearby properties (within the user’s current postcode) on a Google map.

Android programming has proved to be a challenge above and beyond that of basic java programming. Whilst sharing all of the same underlying tools, Android introduces several new concepts to the aspiring programmer, such as the activity lifecycle of applications, requiring the developer to understand what Android is doing behind-the-scenes at any given time. The developer must also be mindful of how the application’s computations interact with the visual XML layout, designed for the user, mapping each element of the display to its background code appropriately.

The following figure is an approximation of the application’s functionality, intended to illustrate the core sequence of events involved in its lifecycle:

**MainActivity**

**Client – Asynchronous task**

App fetches user location (lat/lng) from Google location services.

User’s current location is displayed on a Google map below their lat/lng coordinates.

The Asynchronous Client class is instantiated and executed. User location is passed to Client via constructor.

Application plots properties received from back end on Google map, displaying the postcode and sale price on-click.

Markers are colour coded according to price bracket.

Has the back end responded?

**NO**

**YES**

Opens a network socket and connects to the application’s back end.

Sends user’s postcode location to back end in order to be queried against land registry database.

Receives multiple nearby property sale details from back end.

Price, postcode, address, transaction date.

Stores nearby house sale details in ArrayList that can be used by MainActivity class.

The greatest challenges in developing the front end of this application were thread management, communication between classes and attempting to implement various 3rd party APIs. The application’s main thread (UI thread) performs a great deal of work each time it receives a Google location update. This computational requirement occasionally caused the application to crash unexpectedly and frequently made the interactive map unresponsive whilst it was ‘thinking’. Calling methods between classes also posed its own issues. Due to the inherent nature of Android programming it is impossible (or at least poor practice) to create an instance of the MainActivity class, making it became very difficult to implement MainActivity methods in the package’s other classes.

It was our hope to reduce the computation requirements of the application by moving all geocoding and reverse geocoding to our back end virtual machine. Unfortunately, time and group experience prohibited us from achieving this in time for the task 5 deadline. This improvement should have removed a significant amount of workload from the phone’s processor, allowing for other upgrades such as additional map marker placement and more frequent location updates, boosting the application’s performance. Given more time, this would be worth re-attempting.