CS1013 Programming Project I

Group 10 Project Report

Conor Gildea
Tobias Hallen
Seán Hassett
Catalina Rete

Organising & Splitting Up Work:

For communication, the team set up a group on Messenger to enable instant text communication between all members. We also used an app called Wunderlist which allowed us to create tasks that needed to be completed and assign them to team members. As well as this, we organised at least one group meeting per week outside of the assigned lab time.

While there was some amount of combined effort and working across the entire program, our four team members each had a dedicated specialty where the majority of their work was focused. These specialties are outlined below:

Conor Gildea:

My objective consisted of integrating the interactive maps with data visualisation into our project. In order to achieve this, I had to research how to implement and use the Unfolding Maps API in our project. I then had to research how to place and adjust the size and colour of the markers on the map in order for us to achieve our clustermap.

Another feature I worked on was the ability to gather GPS coordinates from just a postcode. To do this I used an open source API (postcodes.io), which allowed us to ping the website with our postcode and receive the GPS coordinates back. This was extremely useful as it allowed our project to display the location of a property on a map, after the postcode had been searched for in the search bar. I also was in charge of setting up the foundations for how we should scale our program for any display, which all members then incorporated into their code going forward.

Tobias Hallen:

My main job for the project was the design of each of the types of data visualisation. I built the classes for the bar charts, line graphs and pie charts. This is done using almost only the basic processing library, along with some others for text formatting. The charts were all made to be fully scalable and interactive, lending a more tactile feel for the program to the user, and meaning that we could build on that interactivity in order to generate more queries. The scalability was hugely important, seeing as though we used the entire 3.6GB dataset, because it meant that the classes needed to be robust enough not to result in any errors or irregularities across wide ranges of data inputs. I also implemented animations in the way each of these were drawn, along with how for example the bar charts change dynamically when flipping through the year view.

Seán Hassett:

I focused on the implementation of MySQL to handle the data for the program. This involved converting the .csv file to an SQL table as well as indexing the table appropriately to speed up queries. I used Java to generate a precalculated table which contained almost all the results we wanted to access already calculated. I also created a Query class in the main program to handle all calls to the tables in order to retrieve data and worked on the Go() class which initialises the user's query and generates results.

As well as working on the data, I created the Year View interface, which allows the user to scroll through the data on the screen for every year. Finally I created the tables on which the property info is displayed following a postcode search.

Catalina Rete:

My main responsibility was creating a modern and familiar user interface that would be error-proof. Thus, I created all the buttons listed in the sidebar, part of the retractable menu, and the search bar which allowed the user to type in a postcode. I also animated them and used appropriate complementary colors to achieve an aesthetic look. One of my goals was to make an interface that did not only look pleasant, but also gathered the information from the user in a way that made it easy to process afterwards. For this purpose, I made use of HashMaps that mapped Strings to Booleans. Most of the features I implemented can be found in the Interface Guide below.

Interface Guide

- Access the sidebar.

- Search for all results for a specific postcode from the full dataset. Return results by pressing Enter.

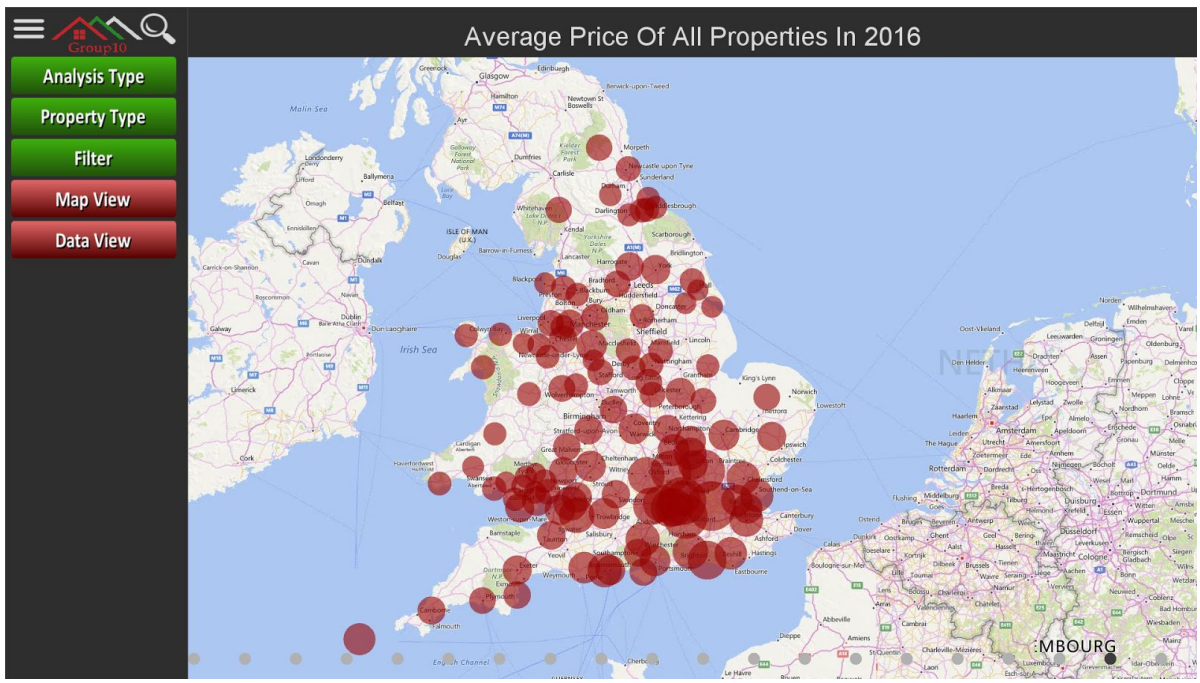
- Select the type of analysis to query from a list of options: Average price, Maximum price, Number of sales and Sales by type.

- Select the type of property to run the query for, including an option to run the query for all property types.

- Select which counties to run the query for by first specifying England, Wales or both, and then selecting counties from a list.

- Map View: Displays the data on an interactive map, represented as clusters positioned over the selected counties and sized according to the data for that county.

- Data View: Displays the data as bar charts with the option to click on a bar and view a trend graph for that county. Also displays the pie chart if 'Sales by type' is the selected analysis type.



Features

- **Data**

The data for our program is held in two SQL tables. We are using the full dataset which covers data for 23 years (1995-2017) with over 22 million transactions. The majority of the queries access a precalculated table which contains the results of the queries we would be running. We managed to compress 22 million rows down to 14,605, with each row containing the average price, max price and number of sales for a particular county in a particular year and for a particular property type. This table also stores GPS coordinates for each county to save from overloading the postcodes.io website. This precalculated table makes queries almost instantaneous, while we use the full table to retrieve data for specific properties. Swapping between a local and a remote server is done by modifying a boolean, with the final submission set to run remote.

Note: The program slows down when connected to the remote server. Although it can handle queries on every county, it is recommended to use a smaller amount of counties for quicker testing.

- **Year View Interface**

When the user runs a query, the data for all 23 years is returned and passed to 23 different screens. The default screen is 2017 and the user can either use the arrow keys or the radio buttons at the bottom of the screen to view the results of the query for the different years. All of the data views and the map view are displayed on this interface, giving the user easy access to information for all years for every query.

- **Bar Charts**

Our bar charts allow the user to select any amount of counties they want. They can also specify a type of property or use all properties. We then pass in all the values that were selected by the user, as dictated by the hashmaps. The bars grow to the correct height dynamically and we can see values over time using the year view interface. The charts are uniformly scaled across all 23 years which aids the user in seeing the changing values as they scroll through the year screens. Hovering on a bar shows the exact figure the bar represents, as well as the name of the county.

- **Trend graphs**

Clicking on the bar in the chart displays a trend graph that shows the increase/decrease over all the years in one screen. The line graph is also animated and provides interesting real-world insights. For example, we can see the impact of the economic recession in the drop in number of sales in 2008 followed by a drop in average price in 2009 for nearly every county. Pressing backspace on the trend graph returns the user to the barchart.

- **Pie Chart**

If the user wishes to see a breakdown of the percentage of each property type sold, they can select "Sales by Type", then select a county and click "Data View". This data will be shown on an animated pie chart with a legend attached. The user can also view this analysis for the entirety of England or Wales. In this mode, the sidebar changes to prevent the user from selecting more than one county for this data, as well as greying out the Property Type and Map View buttons, as they are not compatible with this analysis type.

- **Maps**

We have implemented the Unfolding library and we use two maps for the project, Microsoft.RoadProvider() and Microsoft.AerialProvider(). The user can press '1' and '2' to switch maps. Clicking on the Map View takes the user to a road map which displays clusters of red markers. The size of the markers is dependant on the magnitude of the result of the query chosen. If markers overlap, we can zoom in to more easily identify them. Hovering over a marker will highlight the marker and show the exact result for that county. The year view interface is also implemented in this view.

- **Search**

If the user is looking for a specific property, they can enter a postcode in the search bar. This action displays extended information on the property or properties at the postcode, and also zooms in on the location on the aerial map. If the user enters an invalid postcode, the program will display an empty table.

- **General**

We have created methods to scale everything according to the height and width of the display, thus making our program fully resizable and assuring the same aesthetic quality on all types of screens.

Problems Encountered

Below is a list of some of the bigger problems we encountered during the course of the project:

- Rescaling text

We initially used Processing's font creation tool to create a variety of .vlw files to use for font. This became a problem however when we looked at scaling the program for different resolutions as these fonts were of a size irrelative to the display and immutable. We solved this by creating new fonts in the code and using the scaling functions to define the font size. Thus we get different sized fonts for different resolutions.

- Preventing mouse-scroll interference

We had difficulty in getting the map and the sidebar to co-exist as they both made use of the mouse scroll. This meant that if you tried to scroll in the bar you would zoom on the map and vice versa. This was solved by locking the mousewheel interactions for the map while the mouse cursor is over the sidebar and vice versa.

- Selecting only one county for Sales by type

As the Sales by type analysis is unique in that it can only be run for either one county or one country, we needed to come up with a way of preventing the user from picking too many locations and crashing the program. This was done by converting the checkboxes for the country and county selectors to radio buttons. At the same time, the Property Type button was greyed out for this mode as it serves no function.

- Making markers visible

We had a problem with highlighting some of the markers as the drawing order of the markers was fixed. This meant that some markers appeared behind others meaning not all of the markers are easily visible even when highlighted. This problem is alleviated by zooming in on the map to separate the markers, but unfortunately it remains unsolved in the final program.