*Programming Project Documentation*

Overall Outline of Program:

This program was designed to make property searching a lot more visually pleasing. The main menu is very simple. It contains a button for an advance search, map search and then the general search using the queries for all of uk.

One of our main aims when designing the program was to separate queries based on when they are needed to be run, so we can preload most of the queries at the start up of our program. This allows a much smoother user interaction, as well as providing a more organised structure.

Before drawing anything on the screen, our program reads in the input file of 50k entries. It could handle a larger database at the cost of starting up slower. The only requirement we made is that the data should fit comfortably in Ram, so that our program can still run smoothly. For this reason, the max database size our program could handle is about 100 MB on an older computer.

For reading from the file, we use a processing library, which actually has a csv reader method, negating the issues with reading a csv file that can contain commas inside it’s fields. de

After reading the data, the queries are pre-loaded.

For all the query processing we are using one class, which deals with the information that was read in, except for the dynamic queries done by the user. They are handled by a separate class that’s specifically made for processing runtime input information. (SearchEngine).

After the data has been initially processed, the user interface shows up, with a simple welcome screen presenting the options to choose a query. User input, and navigation is handled mostly by the ControlP5 imported library. The main aims when designing the program was to have a well distinguished boundary between the separate features, so that’s why at the start only 3 general categories are presented. A simple scrolling search, an advanced search, and a map, all of which navigates the user to a new screen presenting further possible options. The simple scrolling search is used for choosing just one property of the estates, and showing further information regarding that property. The advanced search is a more interactive dynamic searching feature, where the user can enter a text and choose categories, which will be searched for in the database, as well as showing the chosen results on a map. Internet connection for using the map is required, as it is not saved statically in the project. And finally the map option will allow the user to have information on each county in GB, as we felt like the county based information is the most generally summarizable for the user. This is the overall layout of our program.

Individual Contribution:

|  |  |
| --- | --- |
| **Name of team member** | **Work assigned** |
| Irene Ann Tony | * Display of data(Bar Charts, Pie Chart) * Main menu: Back buttons, create each screen for each query on the scroll bar. * Edit the overall visual layout of the program. |
| Michael Adebusuyi | * Uk Map * Advanced Search Query |
| Sharon Olorunniwo | * Map * Colour scheme of overall Program * Queries |
| Szabolcs Gal | * Reading data * Handling data * General program structure design * Leading the team |

Features Implemented:

**Screen Layout (Irene):**

The main menu was designed so that we have one main screen that branches into other screens depending on the different queries chosen. The main scroll bar displays the different types of statistical information that the user can view of just England. When the user selects a query, they are brought to the screen with the charts. Each screen has a “back” button which will bring the user back to the Main menu, this includes the advance search and map search screen. I structured the screens “Advanced Search” and “Search by Map” so that they were user friendly and looked visually pleasing.

**Advanced Search Screen(Michael):**

The advanced search screen allows the user to search for certain housing properties under a given search type. The search types were “County”, “District”, “Property” and “Town”. The search type is controlled by four radio buttons, which is selected by the user. The user then has a text field at the top of the screen that allows the user to enter the in area that they would like to search for. As the user types, a list of relevant houses is generated in real time. When the user selects a property from the generated list, the map on the left on the screen will point to where the house is geolocated and the center of the screen will show some statistical information about that property like its price rank or district rank. The user can choose between 3 different map types.

**Map Screen**:

The map screen displays a map of the UK separated by its counties. When a county is selected, statistical information is displayed about that property for example, a pie chart representing the proportion of houses in that district that are old or new.

**Map(Sharon):**

So for the map I used in *“Search by Stats”* made use of the *giCentre geoMap library*. I created an interactive map so that when the user placed the mouse over a certain county on the map the name of the county will appear over the mouse. When the mouse is placed over an area, that area is then highlighted. So in order to get the County Name I had to read in a *.shp, .dbf file.* The ShapeFile which was named *GBR\_adm2* was downloaded and then read in so that it could be used. The .csv file is separated in a table and each column is given a title. In order for us to display the county name we must read in the strings under the column titled *"NAME\_2"*. Each county of the map also had its own *COUNTY\_ID*, which is used to locate a specific area on the map.



This line of code finds the COUNTY\_ID of the county that the mouse has pressed. The the name of the county is then stored in *String County.* Which is then displayed. This string *String county* is then used to to display the right visual data corresponding to the the to county that is being displayed on the map.

However we did encounter an issue with this. We experienced lagging when the county name was being displayed. To combat this issue I used an online tool on[*MapShaper*](http://www.mapshaper.org/)to edit and simplify the .csv, .dbf and the .shp file so that the lagging is reduced and almost eliminated.

**Main Screen(Michael):**

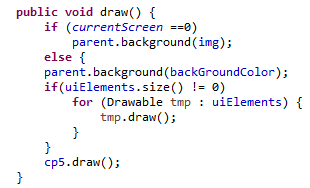
The main screen implemented a scroll bar that when an element from the scroll bar was selected it will direct the user to another screen which will display a query which corresponds to the element that the user selected. The main Screen, on the top left had a button that redirects you to the “Advanced Search” screen. On the top right of the screen, there is a button that redirects you to the “Map” Screen.

**Main Screen colour scheme/design(Sharon):**

For this project I was very particular about having a plan and an overall layout for our program. I was responsible for sketching the basic layout of our program including some basic queries and suggestions for our visual representation of data. So in order to make our program a lot more pleasing to the eye for the user we needed to make our program eye catching. I was in charge of giving of a set color scheme to our program. I decided to set the screen background to white as it makes everything legible and it gives the program a fresh and simple look. I decided to use purple and blue set our text labels and lists and fonts. For the home screen however I decided to set the background as an image of "Westminster". In order to do that I had to use the processing method *PImage*. In order to set the screen background for the home screen to an image I needed to modify the screen class. The screens were stored as an arrayList, and its original constructor did not accept a PI mage. Instead of just adding that parameter to the screen constructor I created another screen constructor.



This then accepted a PImage instead of taking in a screen Color background. In order for this to have any effect, I had to modify the draw function in the screen class. So I just added a simple if statement.



However after this was implemented I ran into an issue with displaying the image as the screen. I received an error saying that the image should be same size as the screen. As we were using the *fullScreen()* method in Processing.

I also modified the font used in the program. This was done by downloading a .ttf file and using the processing method createFont. I firmly believe that the font must be legible for the user, so I used a suitable font Size for the the various text Labels used in the program. As well as that I positioned some of the UIElemnts on the screen the screen to give our program a neat layout. To get the background of the scrollable list transparent I used the Processing function *color(int,alpha)* for setting color. As well as assigning the color an integer value the variable alpha controls the opacity of the color.



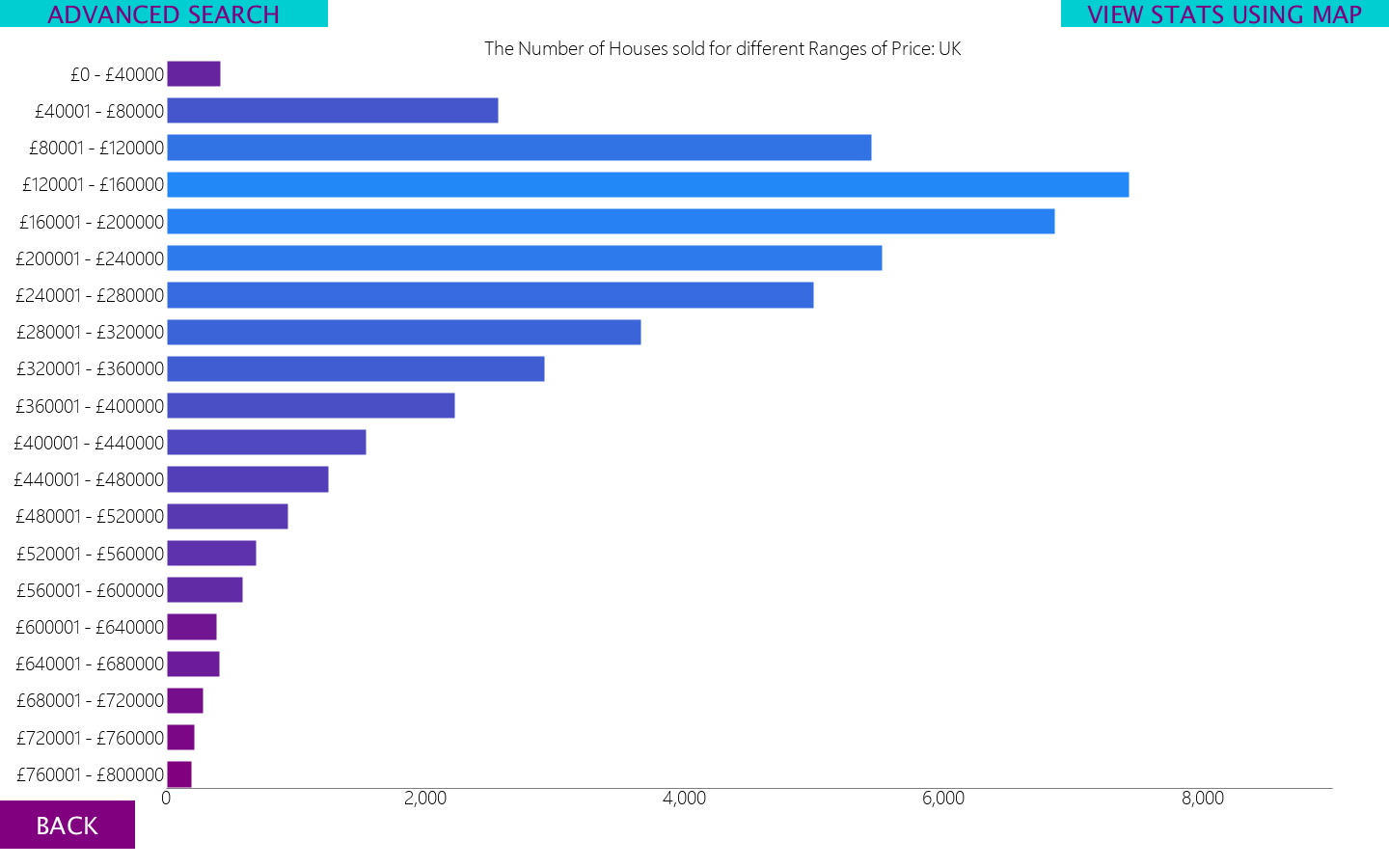
**Queries(Sharon):**

Another one of my tasks was to make sure our code made logical sense. We initially had a dropDown List with the following query to search the data by: Price, Date Of Sale, Postcode, PropertyType, Old/New, Num Name, Street, Locality, Town, District, County. However we I then realized that some of these options were not necessary and did not fit the label of “Search By”. I then decided to remove some of the options in the dropDown and replace the former county option with a new button “View stats using Map”. In order to do that I had to make some changes to the Menu enum class. I had to reorder the screens in that class. I then had to then make the necessary changes in the main Class. It was a very tedious task as I encountered multiple issues with screens not appearing, or wrong buttons presenting the wrong screen or buttons having no effect.

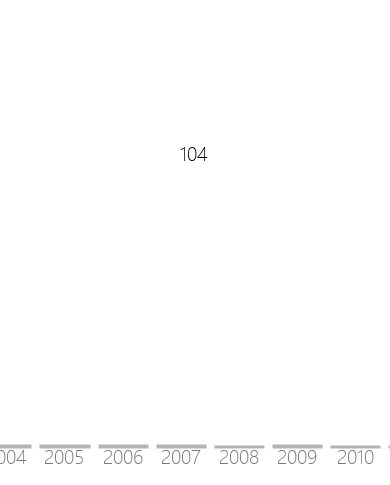
**Bar Chart (Irene):**

The bar chart in this program represents the number of houses in each price range and the number of houses sold in each year. The BarChart class uses the gicentre bar-chart library. The barcharts in this program are visually aesthetic. The colours of the bars are based on two main colours that are chosen by the programmer. The two colours are put into a colour table generator, which generates a table that contains all the colours required to get from the start colour to the end colour.Screen Shot 2017-04-07 at 14.47.12.png

The higher the value the bar represents, the higher the colour on the colour generator table. The bar chart displays a title of the information that it represents as it takes it in as a parameter. The bar chart can also be represented sideways to take into account longer labels.



The bar chart also displays the value that each bar represents when the mouse hovers over it. This is very helpful to the user as some y-axis may be large due to one or two values of data. This results in the bars of the smaller values becoming extremely small and not visible. The same problem was solved in the pie chart class by displaying the values with the key.



Screen Shot 2017-04-07 at 15.04.33.png

**Loading Screens:**

The loading screens are artificial. They algorithmically generated programs that were taken from the website <https://www.openprocessing.org/>.

Problems Encountered:

* Reading the actual data represented by a bar or an area in the charts. This was solved by displaying the data when the mouse hovered over the bar and displaying it on the key in the pie chart.