**Data Analysis and Forecasting Tool**

Kamilah Agbaje, Maruf Hoque, Rathusan Jeyaseelan, Trevor Kiggundu

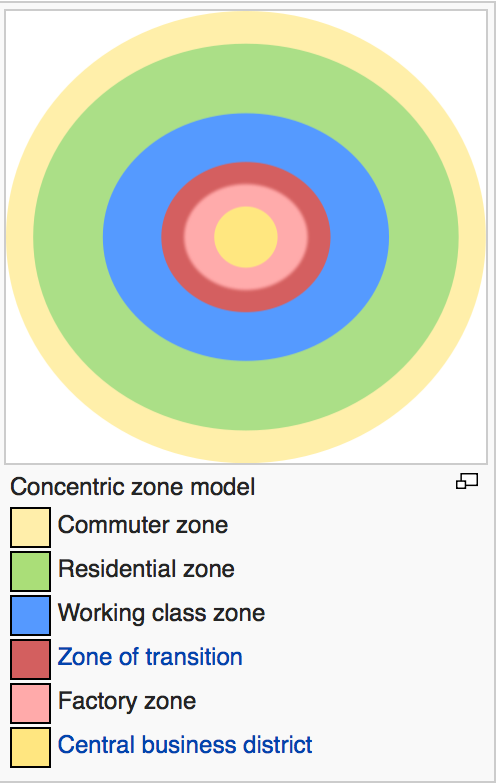
November 2018

A report submitted in fulfilment of the requirements for the module Computer Algorithms and Modelling, Computing and Information Systems Department, University of Greenwich.

**\*\*\*\*\*\*\*Table of contents\*\*\*\*\***

**Definitions:**

* Central Business District: The centre of an urban area, containing a high percentage of shops and offices. A large majority of modern cities follow this city structure (Ex: London, New York).
* Burgess concentric zone model:Model that uses zone/rings to explain the structure of urban cities. The centre-most part of the city is the central business district (CBD), and the corresponding outer rings become less urban as one moves away from the CBD. The other zones include the transition zone, inner suburbs, outer suburbs and the commuter zone.

****

****

* JFreeChart- JFreeChart is a 100% free and open source Java chart library that makes it easy for developers to display professional quality charts in their applications.

**The design of the GUI and model including any assumptions made:**

The model’s solution is obtained through the use of algebra and in particular, the simple linear regression equation y = β1x + β0. After the simulation of the model, the results are output in a graphical and tabular manner. For the graphical output, a software package called JFreeChart is used, and for the tabular display the JTable class, which is present in the Java Swing package is implemented.

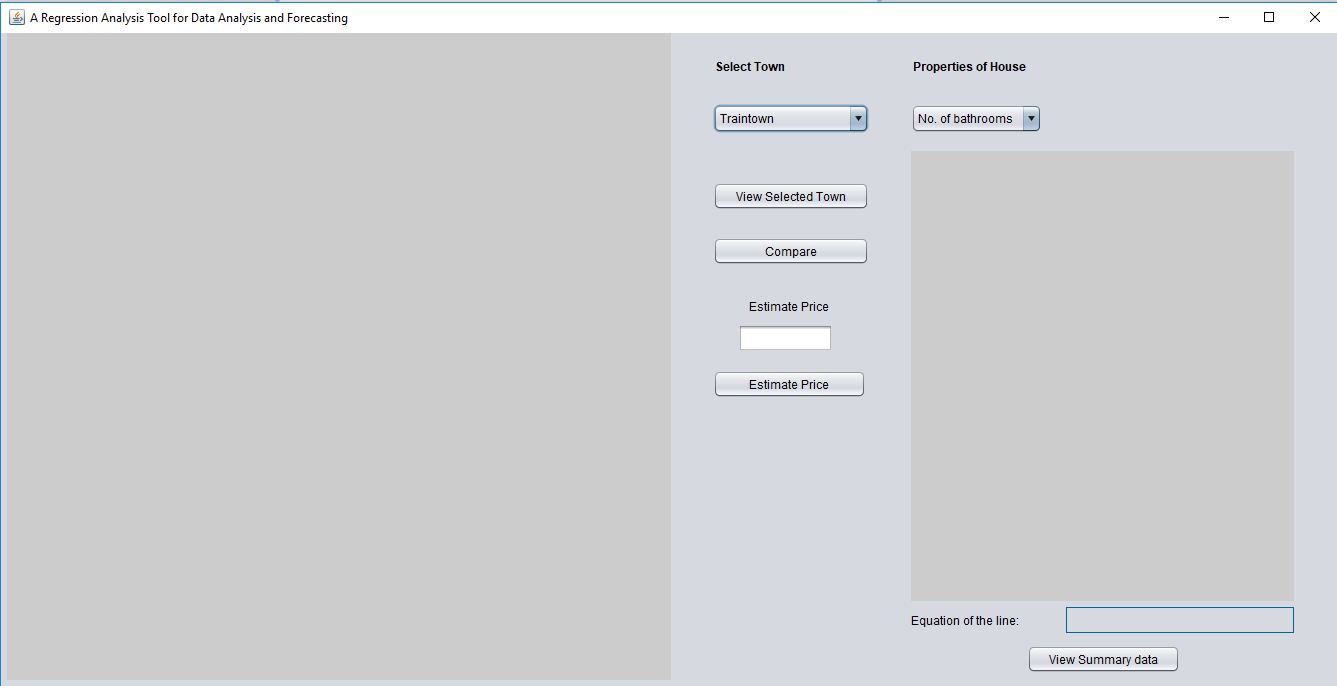
JFreeChart is the ideal choice for this task because it is a very popular charting library that was created over ten years ago. Over 57 versions have been released as a result of bugs being reported and fixed, as well as it being backed up by a community on GitHub. Therefore, it’s reliable enough to use due to its robustness and easy availability. The fact that the library is open source also means that its performance can be assessed and debugged efficiently. This ensures that the library will not affect the quality and stability of the application.

The application definitely makes use of the open-source Java libraries. In general, displaying details of the regression analysis and plotting the coordinates of data and the regression line is essential, but the JFreeChart API still has to be adapted to meet the requirements. So to adapt JFreeChart to the solution, one dataset is for each town and then each town had two series, with an extra series for prediction. It is known that JFreeChart allows for line charts to be displayed from dataset regardless of whether they have one or more series. The library then allows for straight lines to be drawn through each (x,y) point provided within a series. The first series contains the coordinates of the scatter plots, and for this series the points are plotted without a line, allowing it to serve its purpose as a scatter graph. The second series contains two coordinates, the first one being (0,y-intercept) and the second being (x̄, ȳ). Drawing a line through these coordinates serves as a representation of the line of regression. The final series consists of a single coordinate, which visually displays a point at a user input x-value and the corresponding y-value after the input has been substituted into the regression equation.

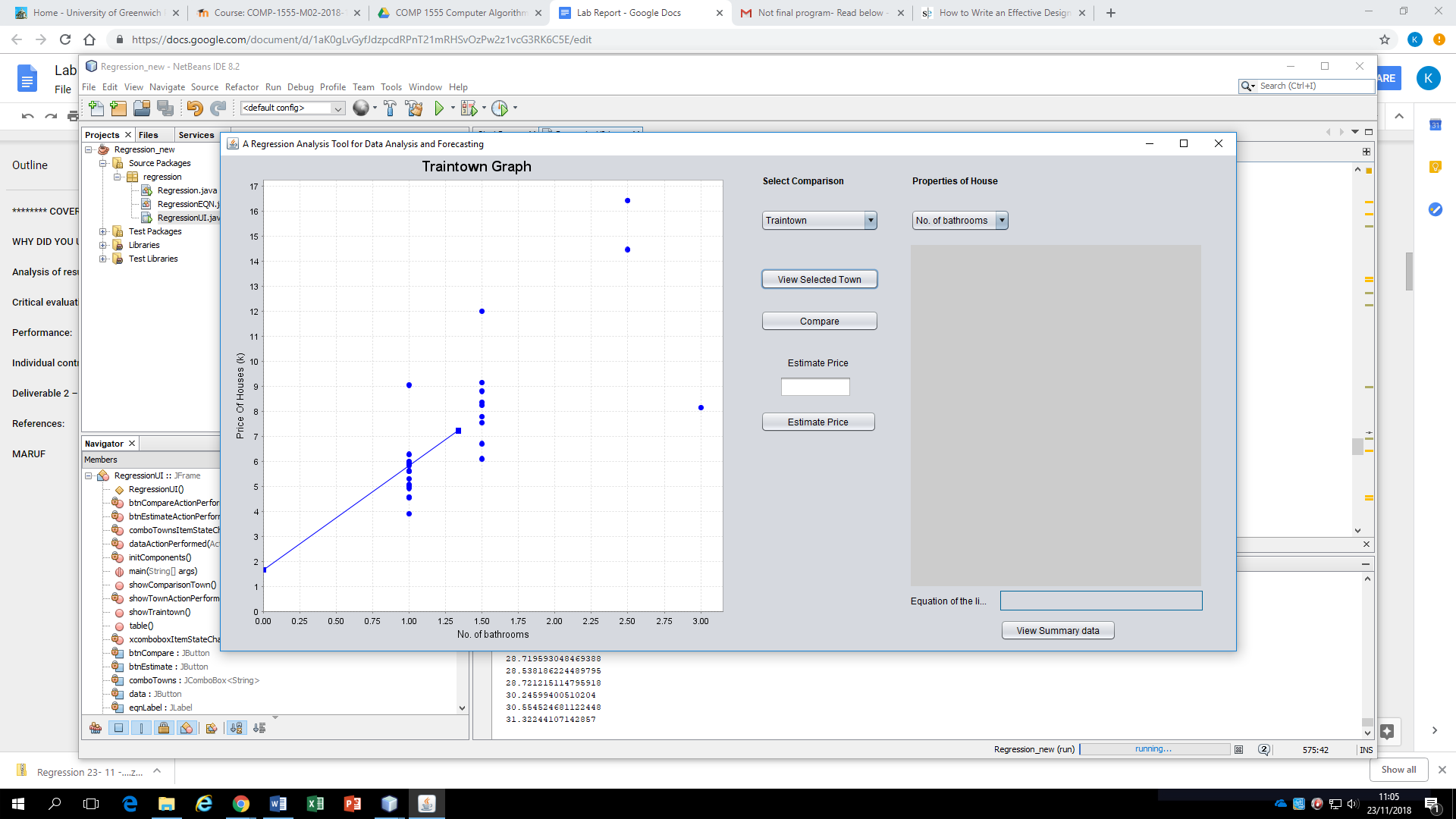
The goal was to make a visually pleasing yet simple GUI design for the user. The GUI consists of:

* 2 JPanels
* 2 JComboBox(es)
* 3 JButtons
* 2 JTextFields
* 4 JLabels
* 2 JOptionPanes

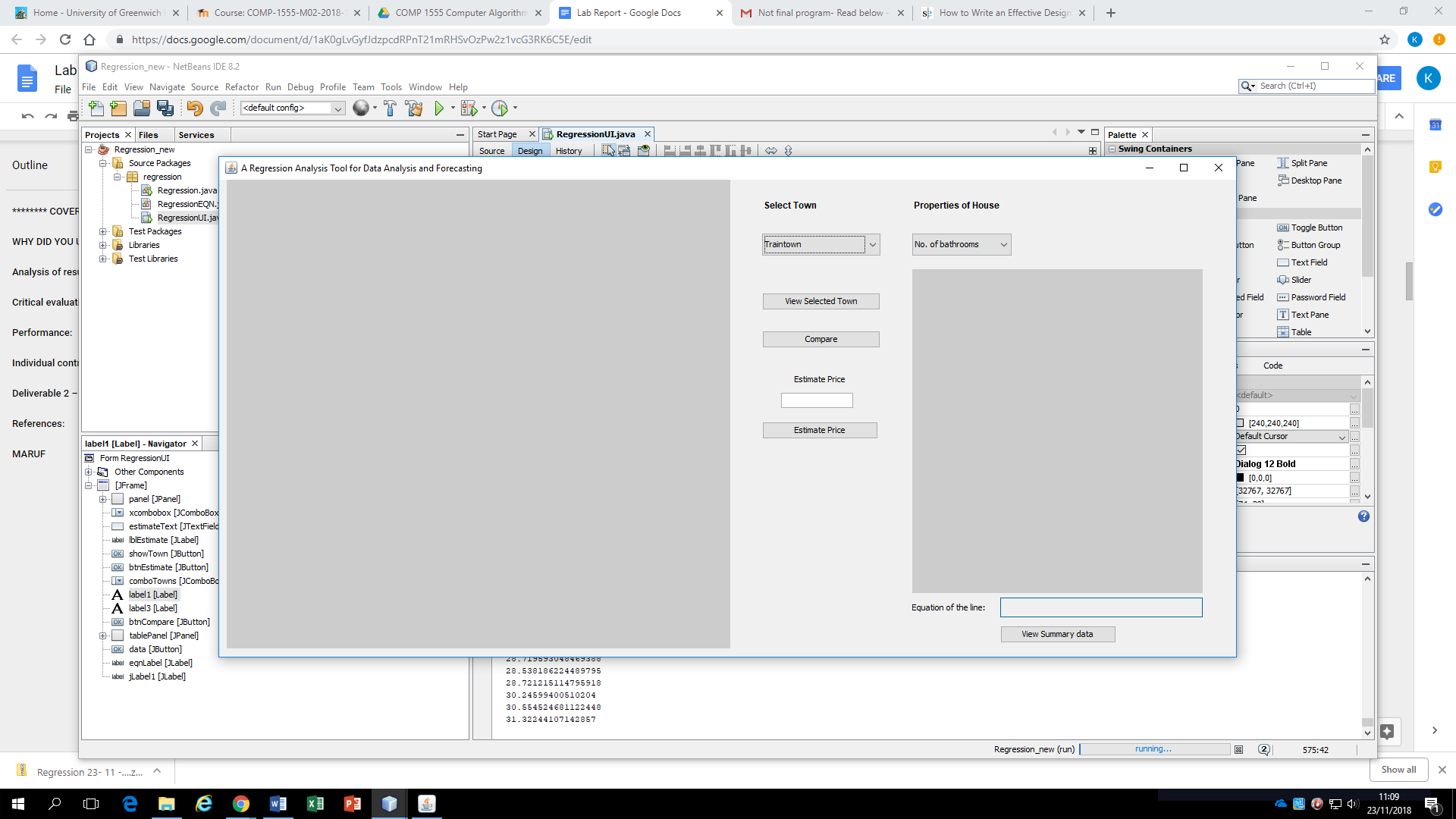
The JPanel to the left displays the X-Y grid, and the one on the right displays the summary data.

****

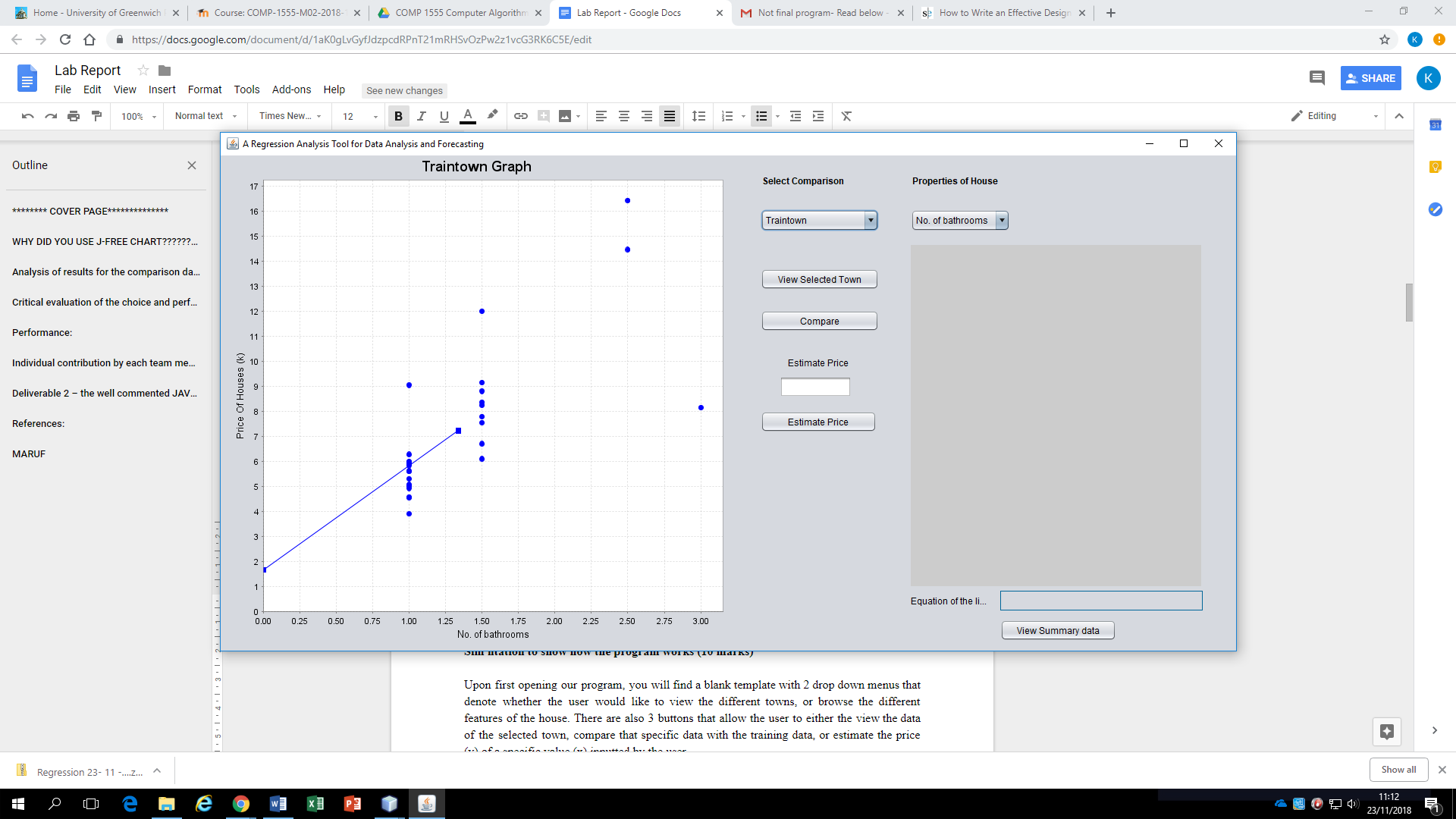
The JFreeChart API is implemented into the program by using the panel to display the axis titles, scale and graph title.



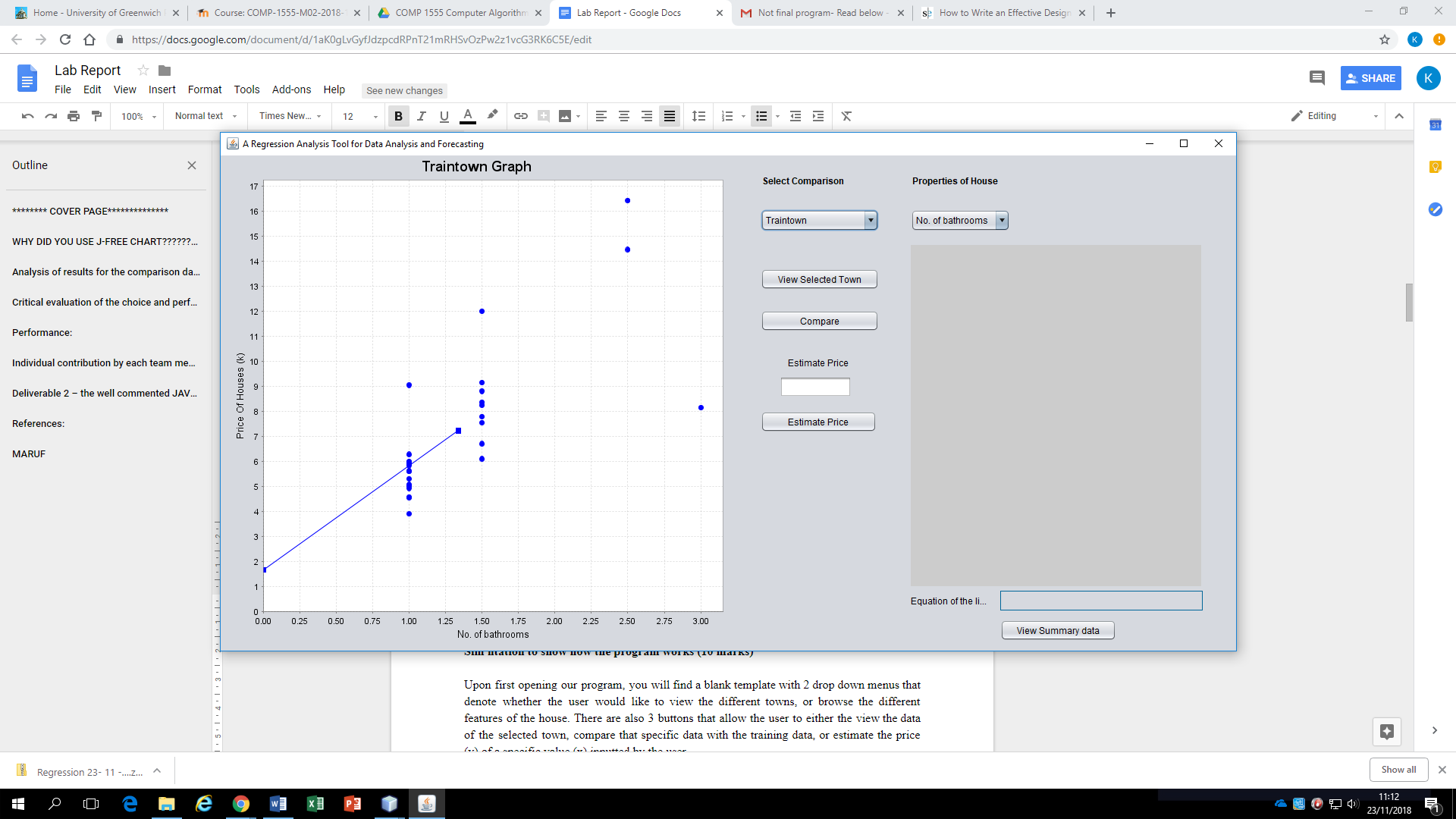
Two combo boxes are also created to relay the options given by the corresponding JLabels, which are named ‘Select Town’ and ‘Properties of House’.



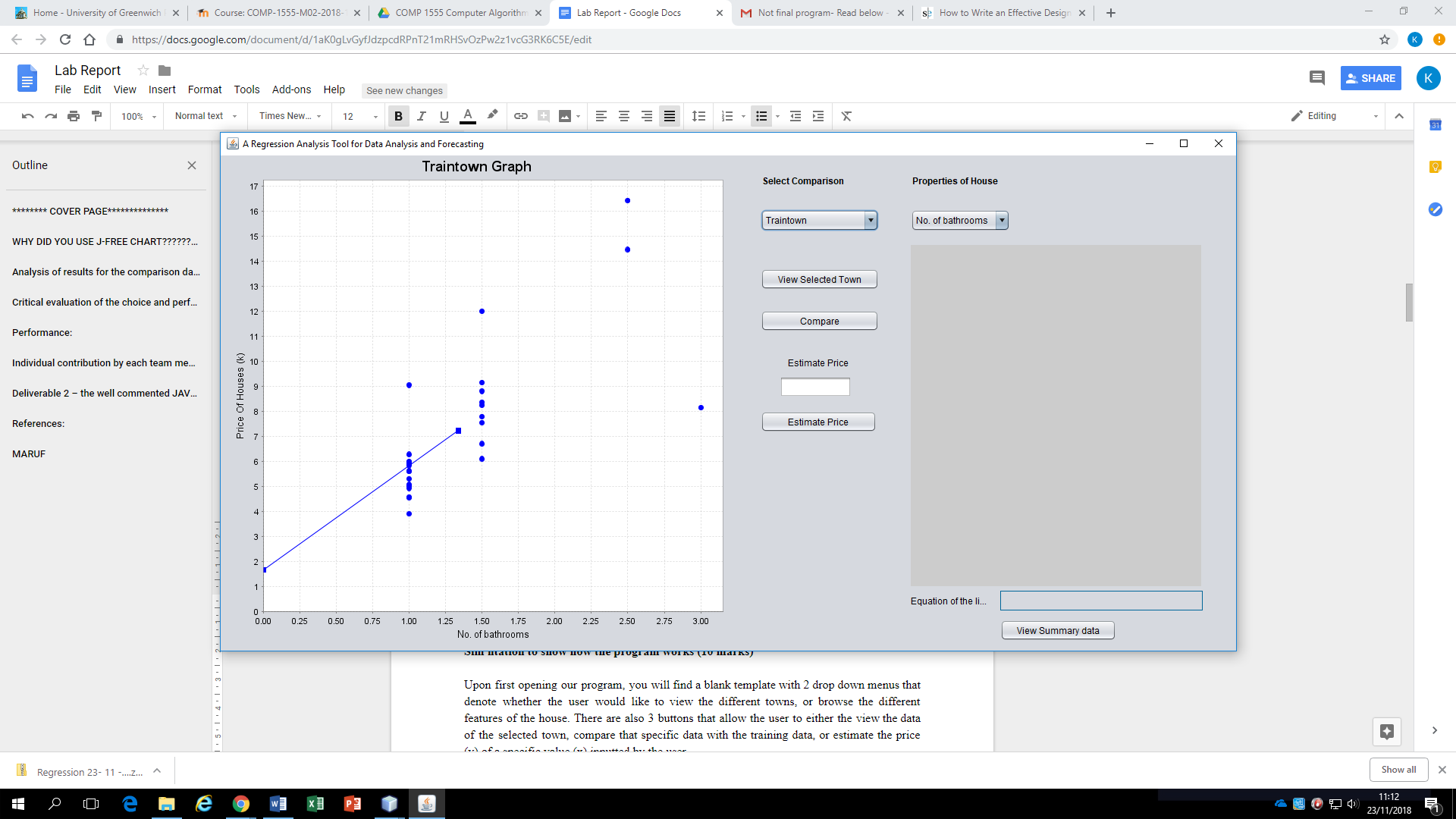
4 different buttons are created for the program, each individually named ‘View Selected Town’, ‘Compare’, ‘Estimate Price’. Clicking the ‘View Selected Town’ button displays whichever town and parameter is selected by the user on the chart within the left-most JPanel (X-Y grid).



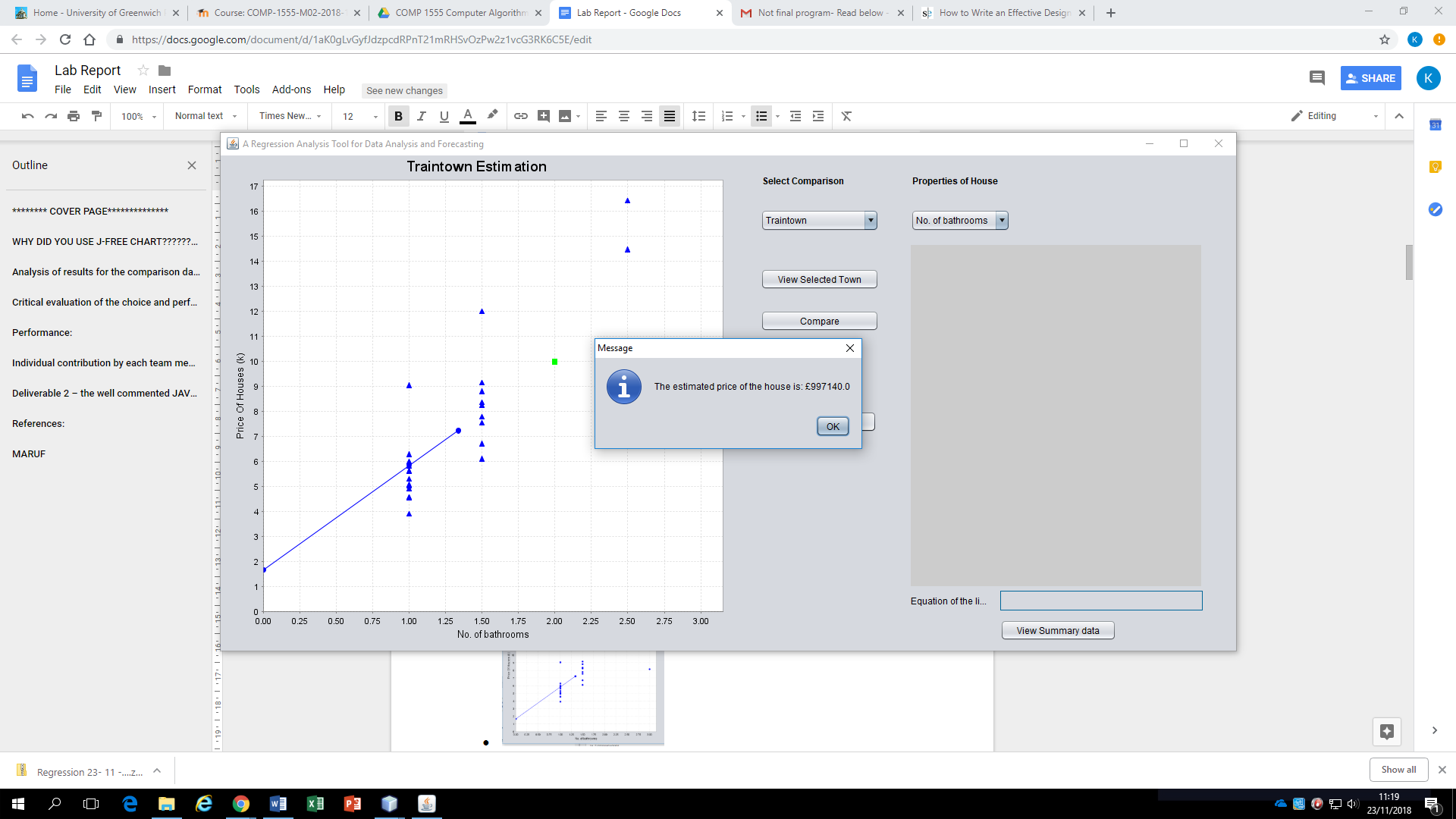
Clicking the ‘Compare’ button results in the chart showing a side-by-side comparison between Traintown and whatever town has been selected from the ‘Select Town’ combo box.



After the values are inserted into the JTextField here, The ‘Estimate Price’ button is pressed, allowing for the prediction to take place.

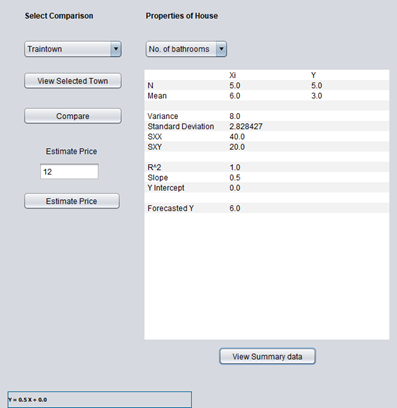


The firing of the ‘Estimate price’ button results in a JOptionPane that displays the estimated house price.



After an estimation, clicking the ‘View Summary Data’ button results in a table being displayed in the JPanel above it as well as the regression equation being shown in the JLabel below the JPanel on the right that displays the summary data.

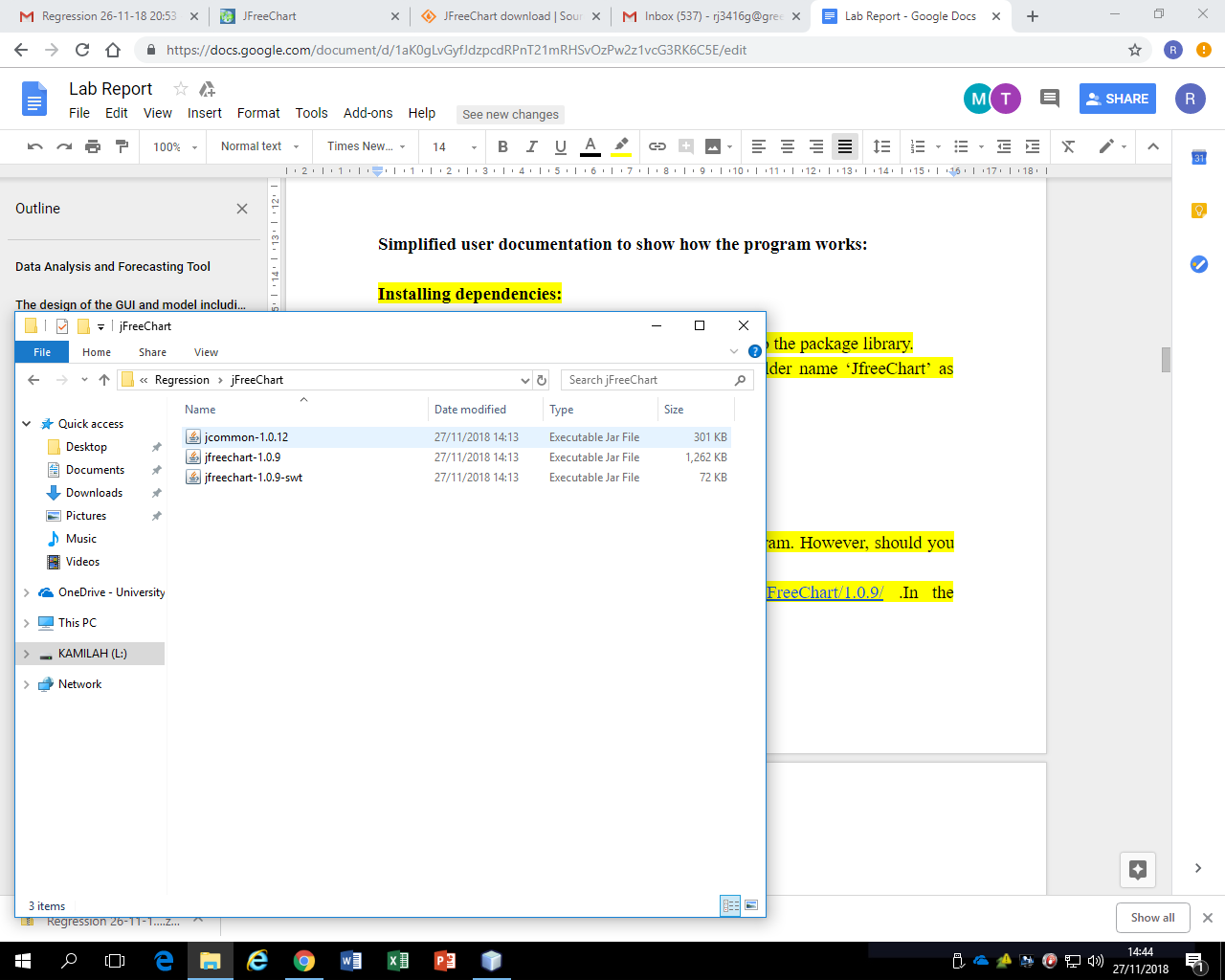


****

**Simplified user documentation to show how the program works:**

**Installing dependencies:**

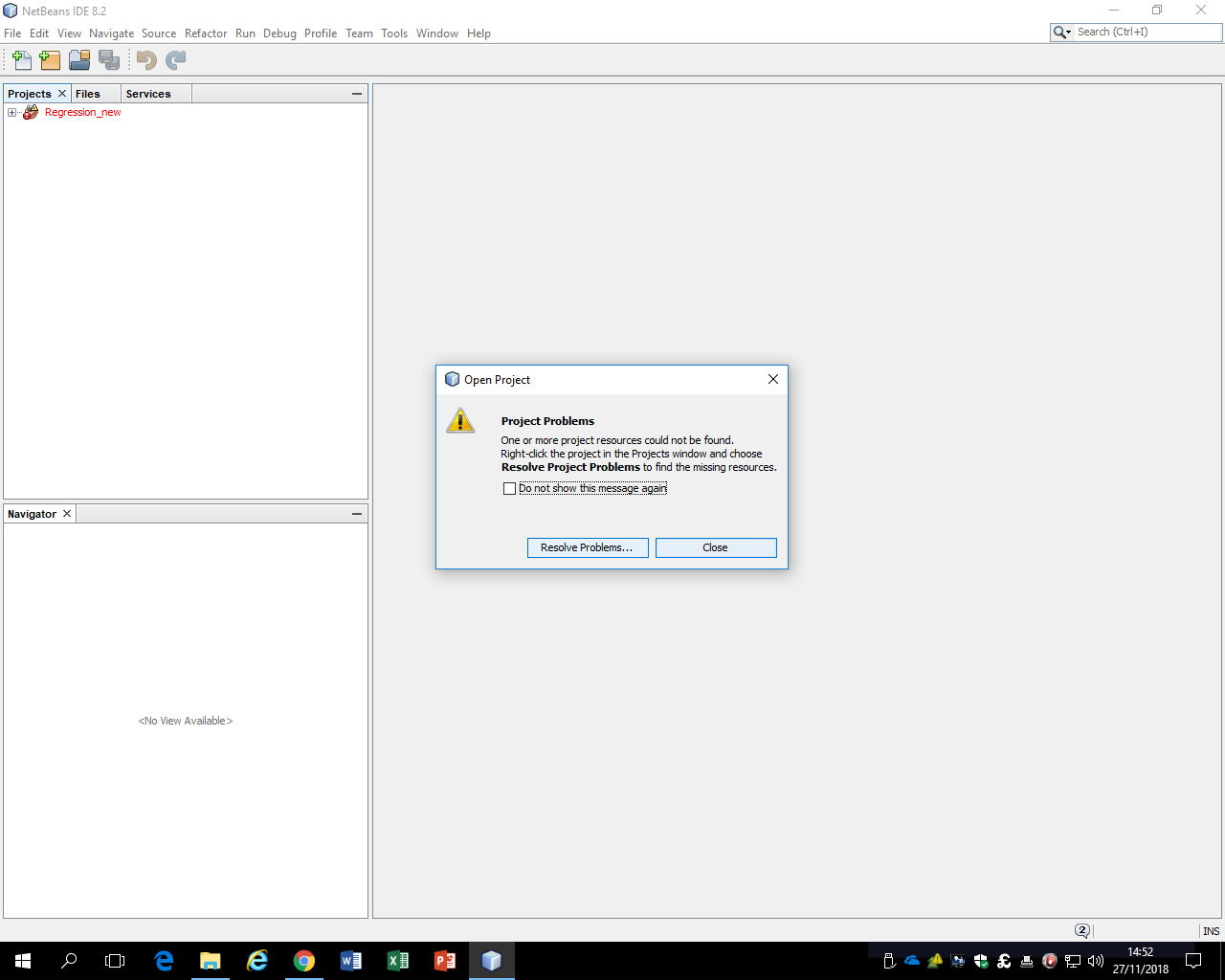
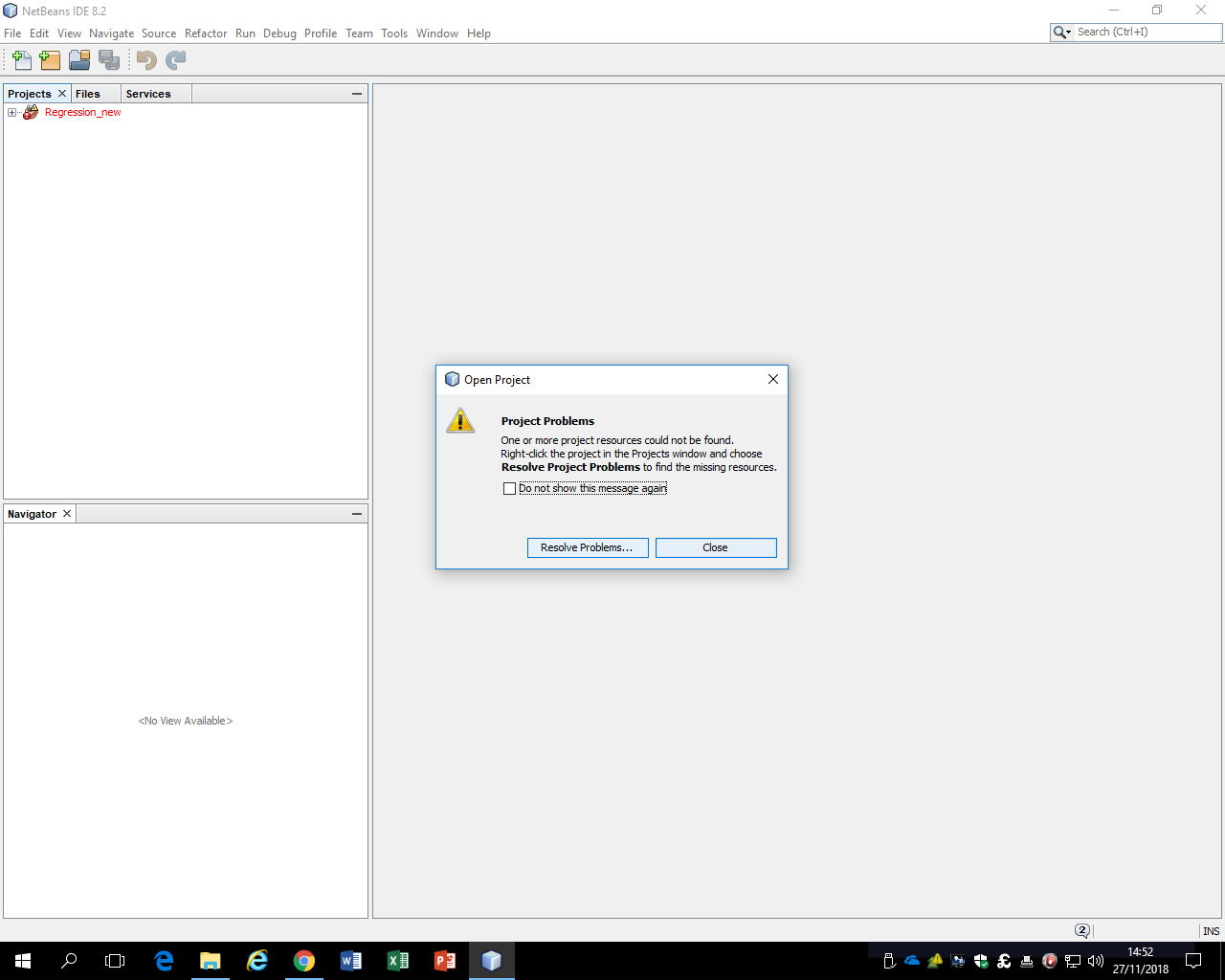
To run the program, three .jar files (jcommon-1.0.12, jfreechart-1.0.9, jfreechart-1.0.9-swt) need to be linked to the package library. They have been provided in the project files in folder name ‘JfreeChart’ as below.



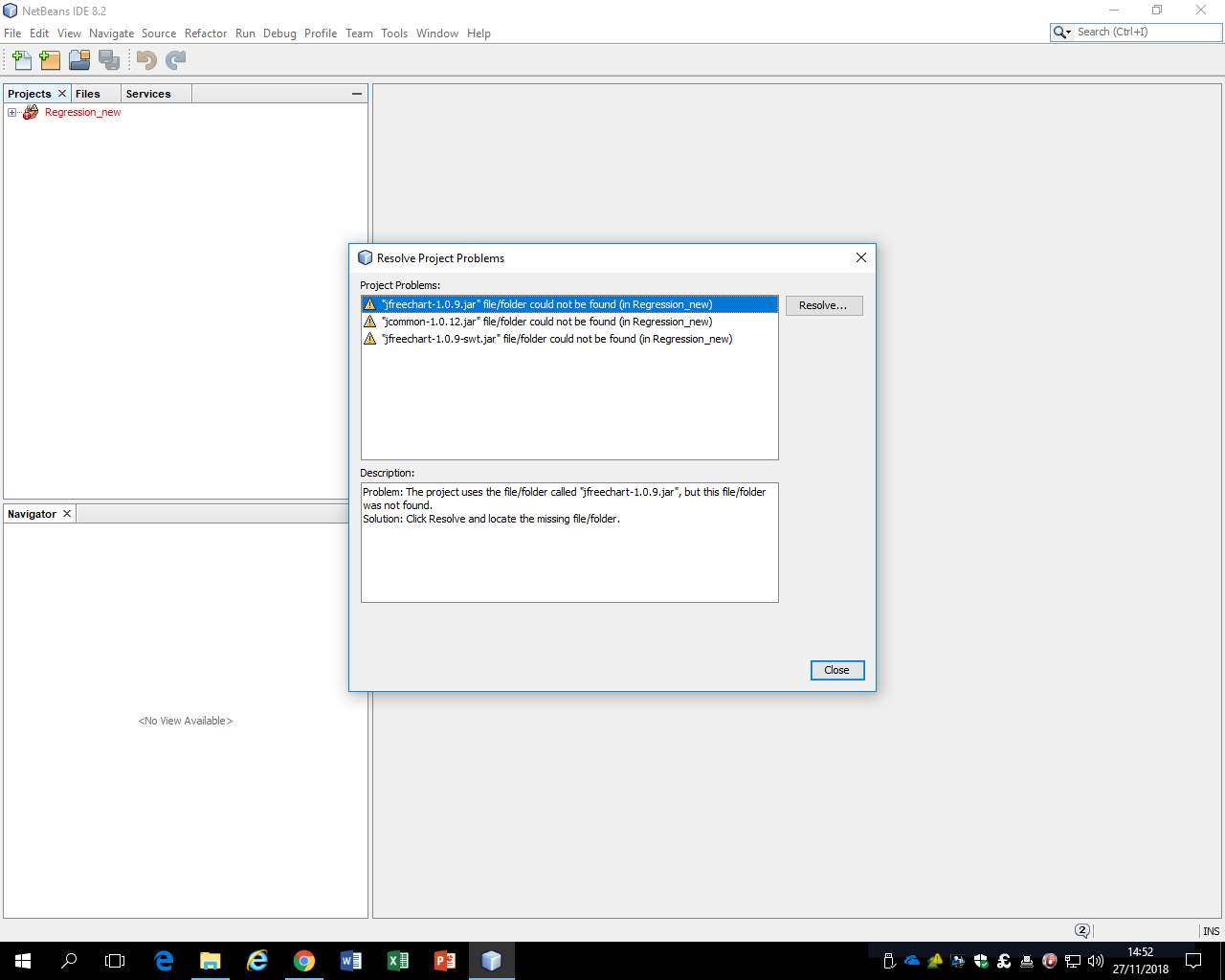
We have provided a detailed list of steps required to solve any issues that may occur when opening the project for the first time.

**How to link dependencies to the package library:**

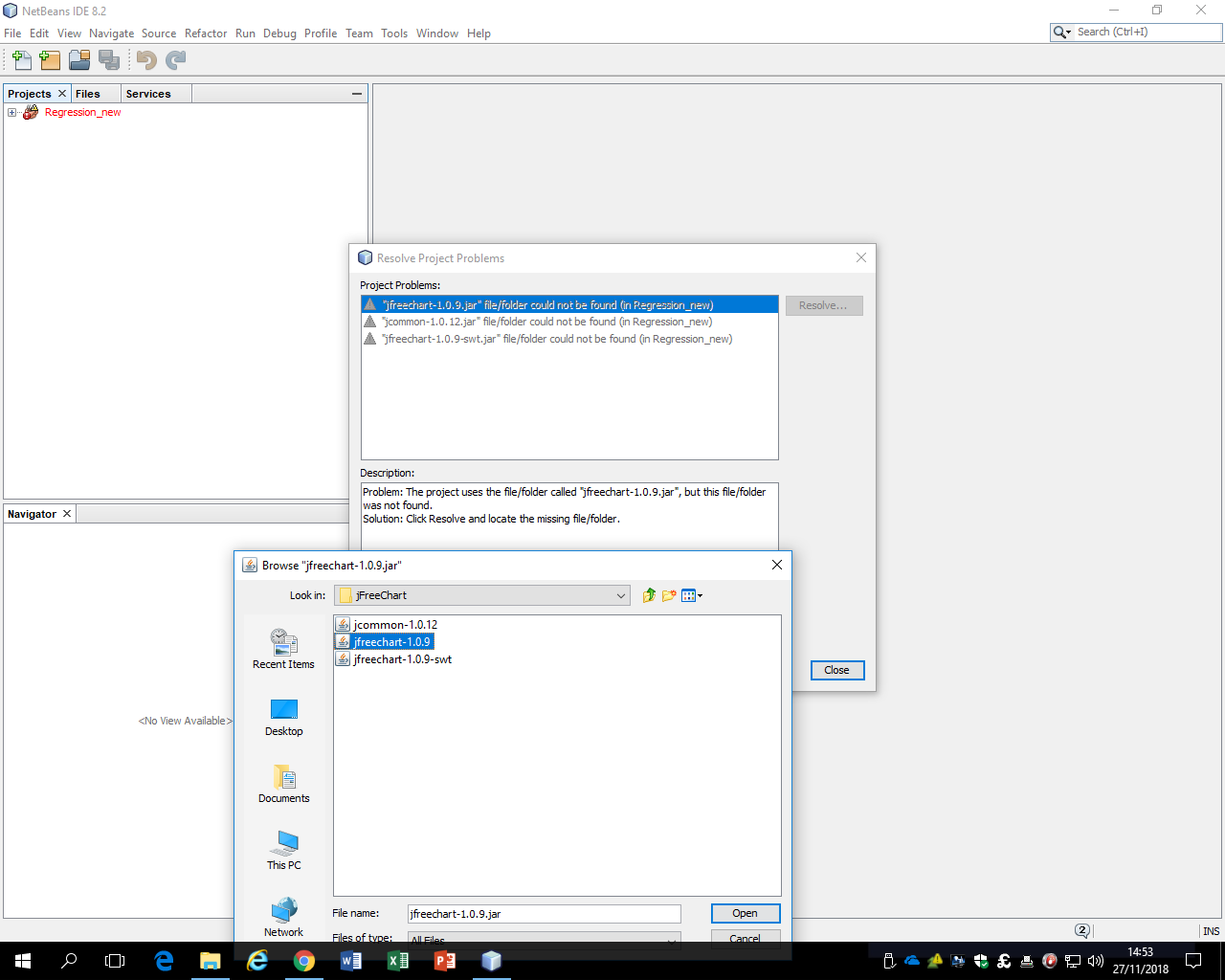
Once the program is opened in NetBeans, there will be a error message stating it cannot locate the JFreeChart files. To resolve this issue, click on ‘Resolve Problems…’



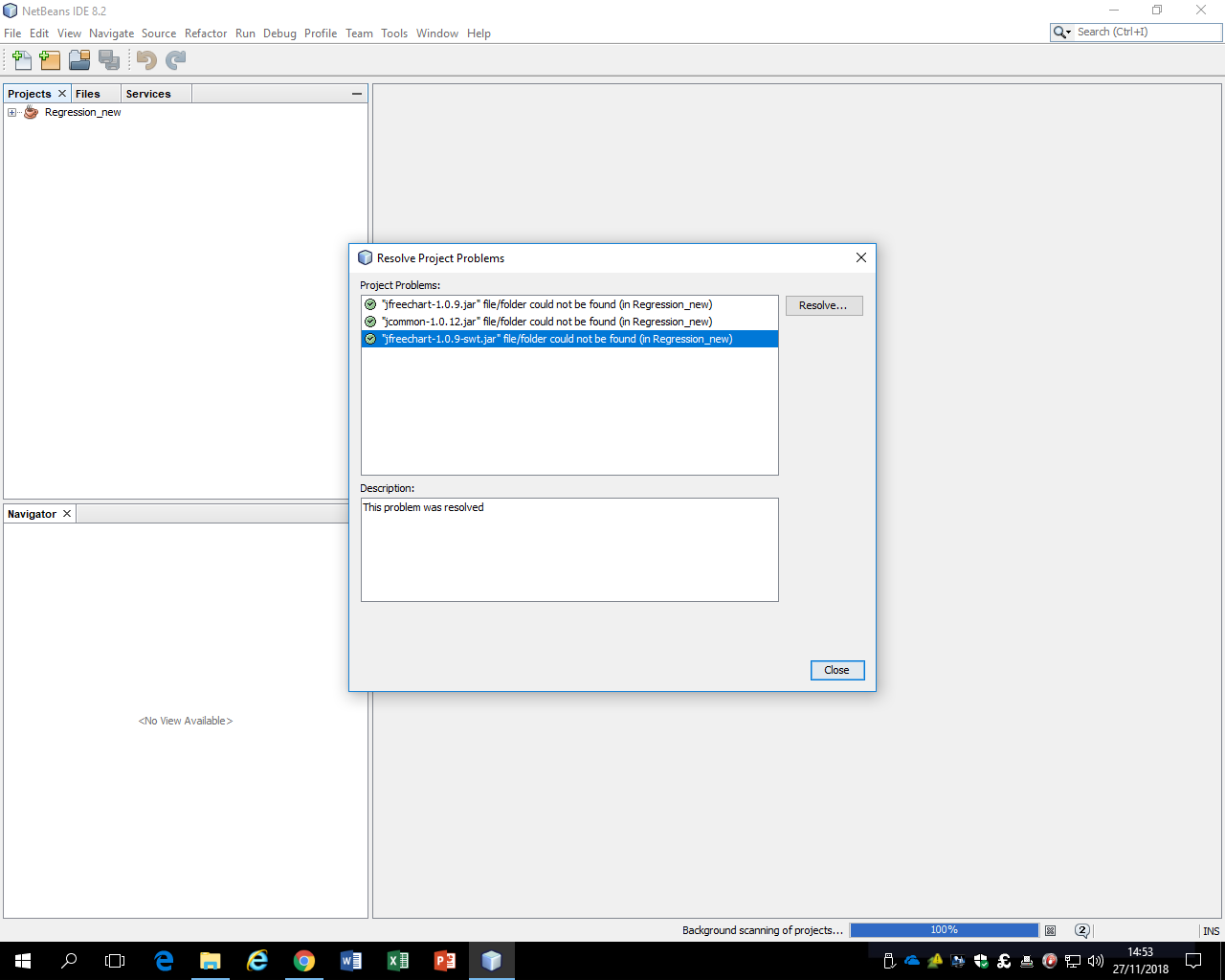
This will lead to a pop-up box being presented on the screen. Click on any of the 3 files and click the ‘Resolve...’ button.

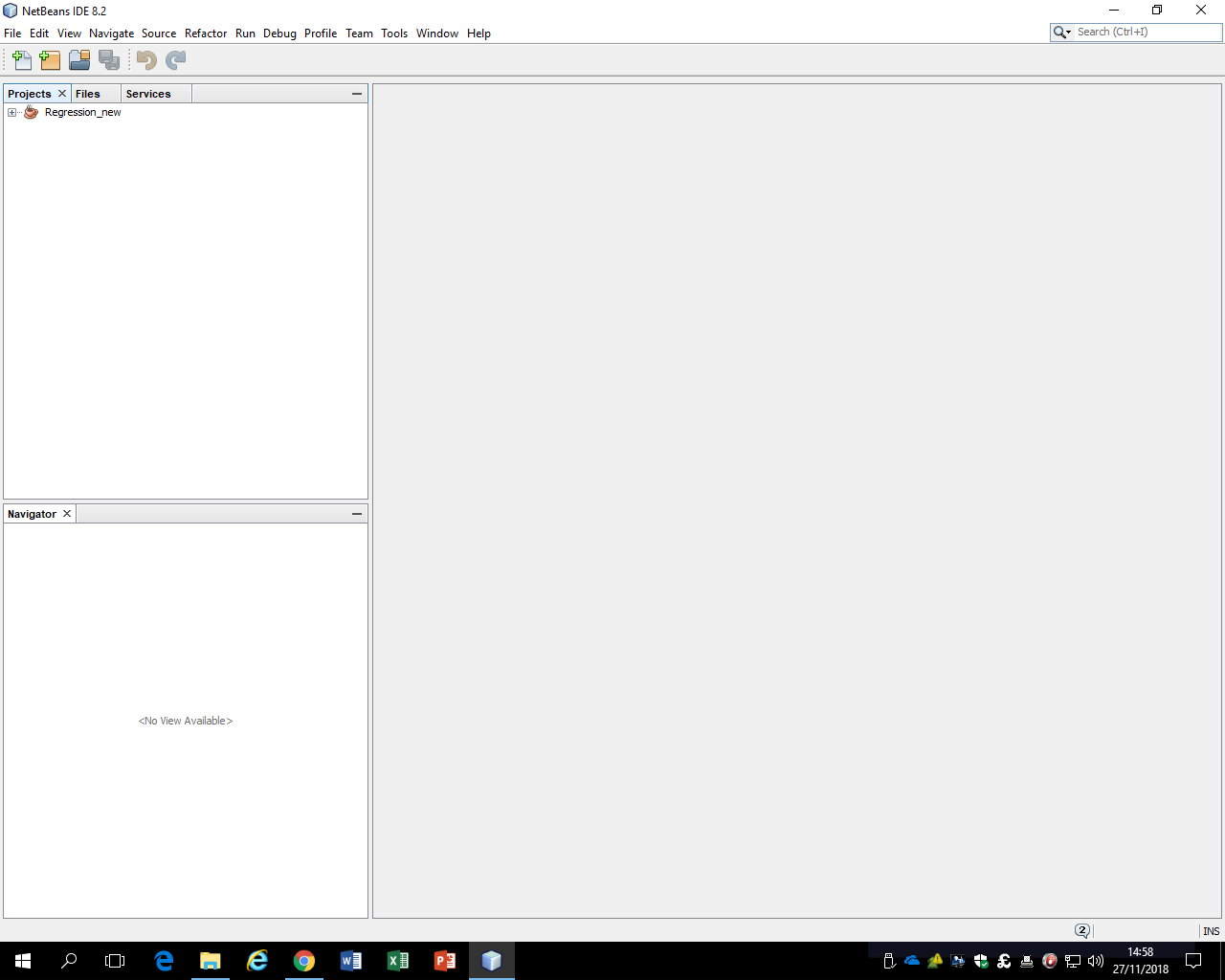


Next, locate the corresponding file which are already included within the project folder under ‘JFreeChart’. Repeat this step until all three files have been successfully located.



Once the three files have been located, the pop-up box should look like the screenshot below.

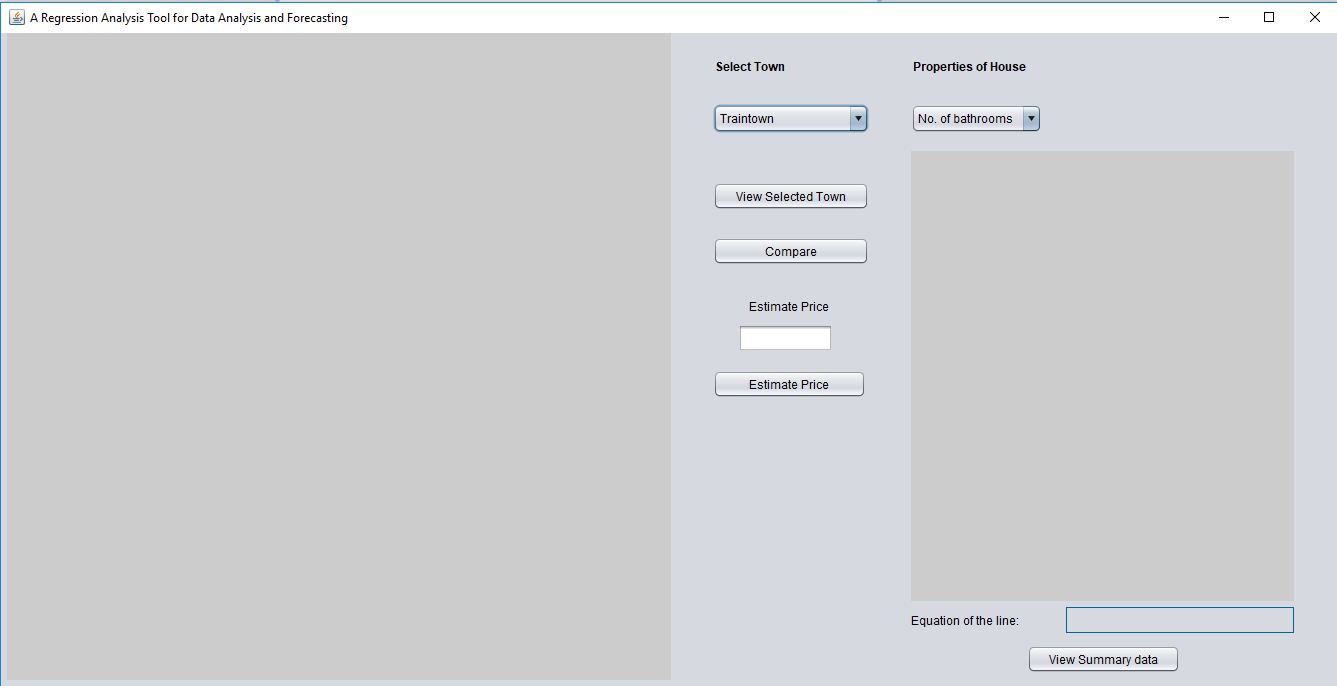




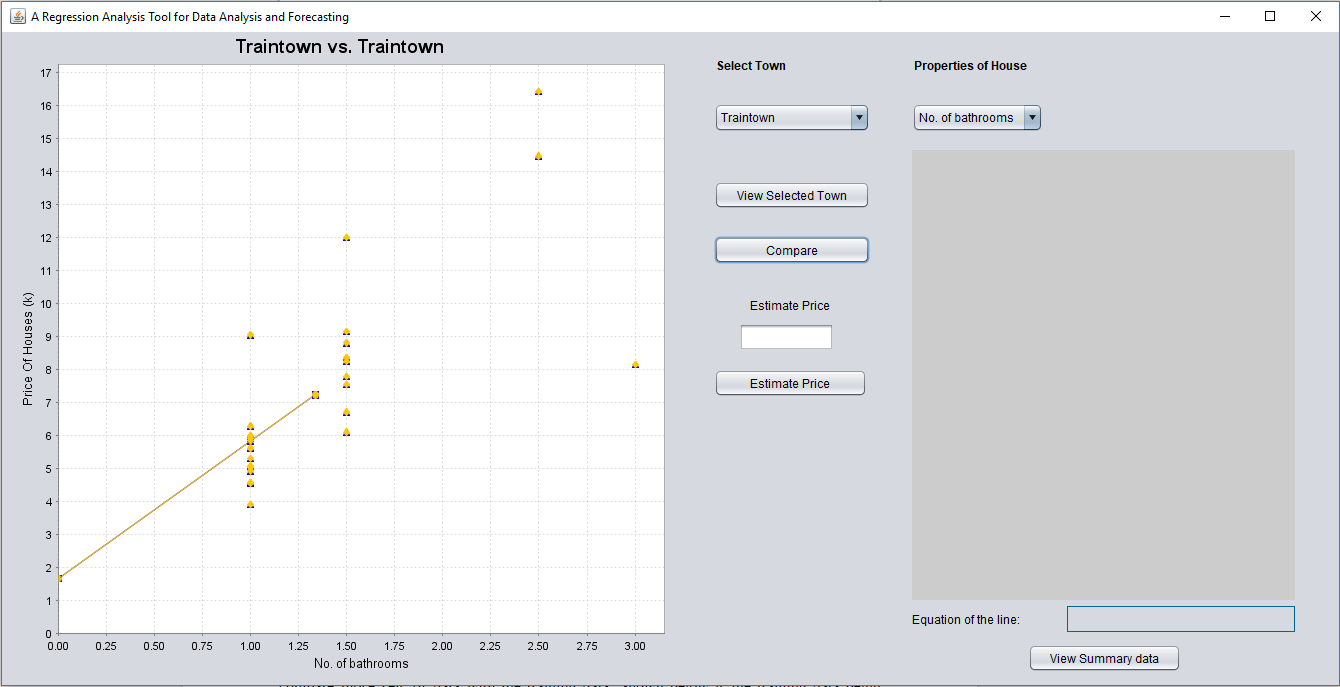
Once completed, the project file should no longer appear in red, indicating the problems have been resolved.

**How to use the data analysis tool:**

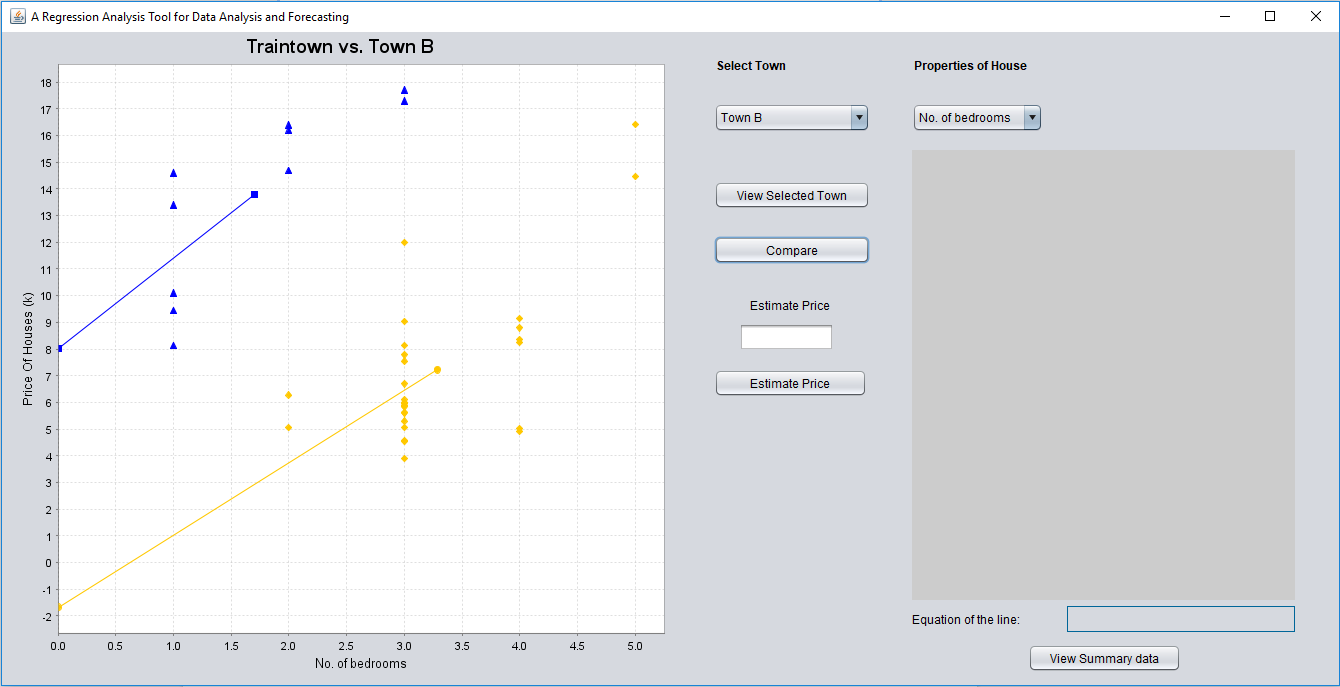
Upon first opening the program, the user finds a blank template with 2 drop down menus that denote whether the user would like to view the different towns, or to browse the different features of the house. There are also 4 buttons that allow the user to either the view the data of the selected town, compare that specific data with the training data, estimate the price (y) of a specific value (x) inputted by the user or view the summary data.

****

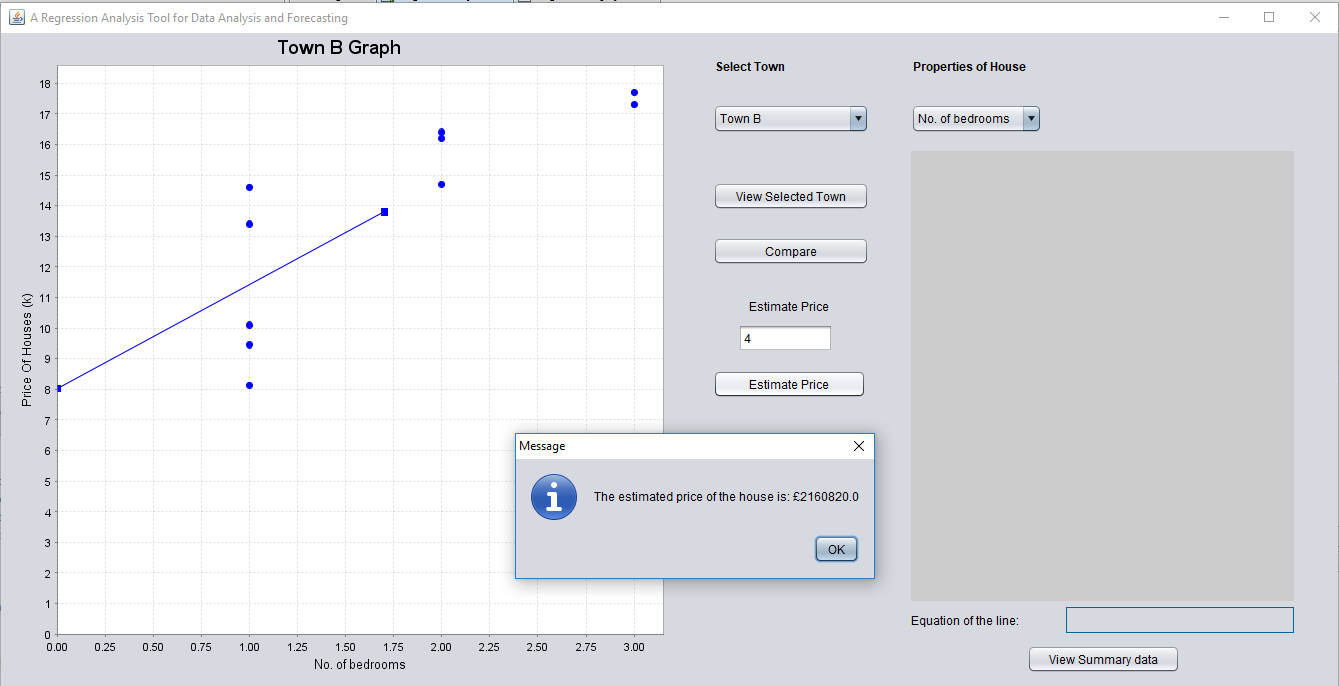
Clicking the ‘Compare’ button displays the training data on the graph by itself if and only the Traintown data is selected on the drop down menu (shown below).



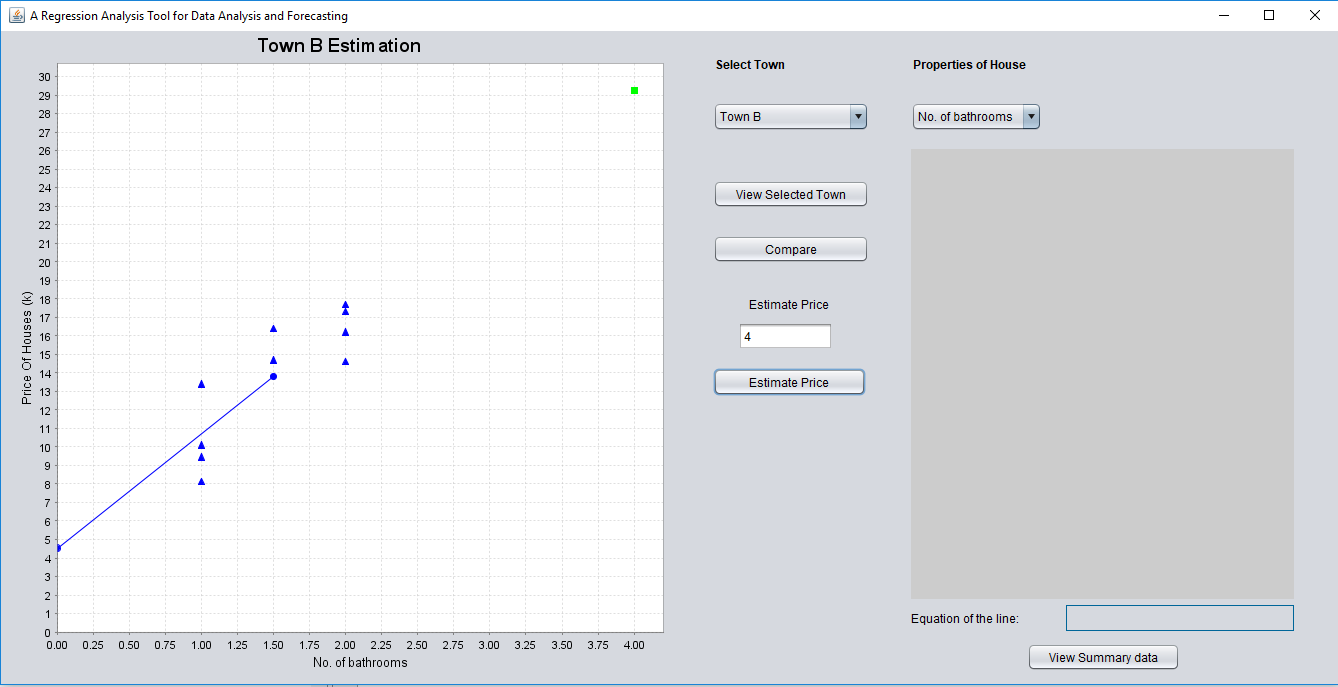
When other sets of data are selected using the menu, clicking the ‘Compare’ button compares those sets of data with the training data. Shown below is the training data being compared with the data from Town B, with the specific x-value compared being the number of bedrooms. The blue data points represent Town B and the yellow data points represent the training set.

****

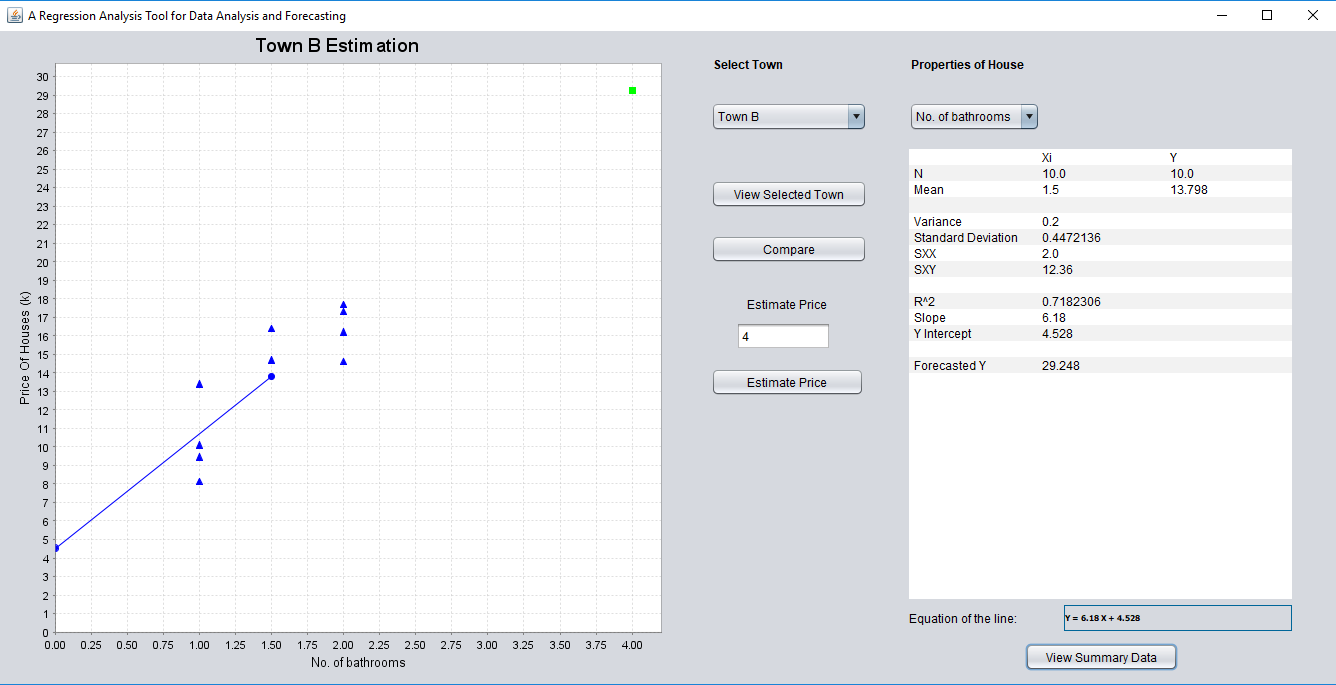
The prices are the dependent variable in this algorithm, meaning that they change depending on whatever x value is inputted by the user. They are then shown on the y-axis of the graph. The coursework specification asks for a feature in the program to give a house price prediction based on a user inputted x-value for one property. Due to this, a button labeled ‘Estimate Price’ is implemented, and when clicked, it gives the user a prediction on the desired enquiry. Shown below is the bedroom data for the houses located in Town B. The user inputted x-value is 4 and the y-value prediction given by the program is £2,160,820.

****

Clicking the ‘OK’ button will then return the user to the original page and plot the new data point on the graph so it can be compared with the other data from Town B. The title of the graph will also change to ‘Town B Estimation’ so that the user knows that it is a modified set of data on the graph. ADD ARROWS

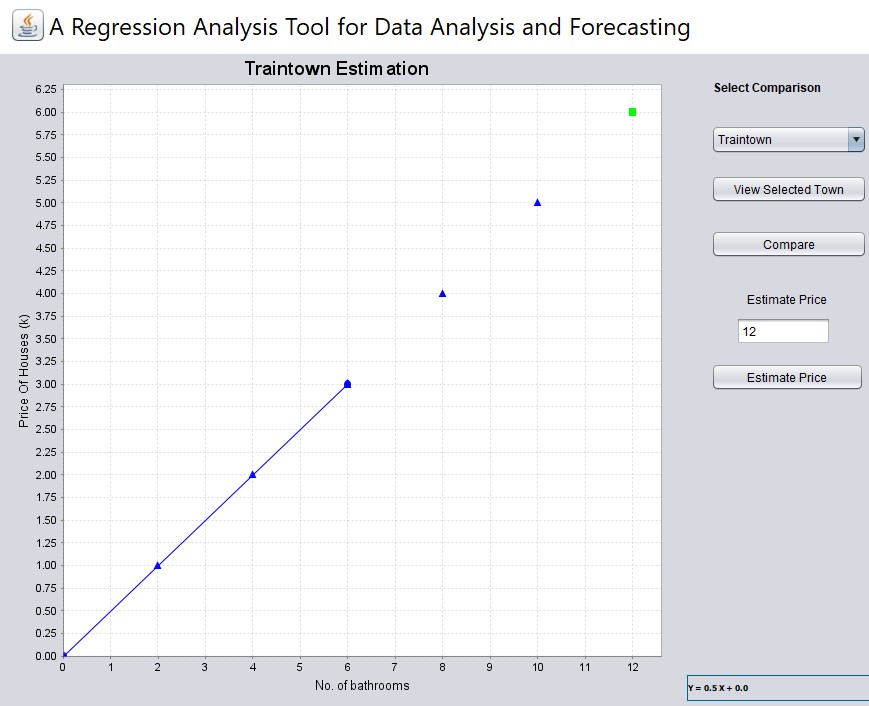
****

The user can also choose to view the summary data for the new point that has been plotted by selecting the ‘View Summary Data’ button. This displays the statistical data such as the slope and y-int, on the right side of the interface. The equation of the line is also displayed underneath the statistical data.

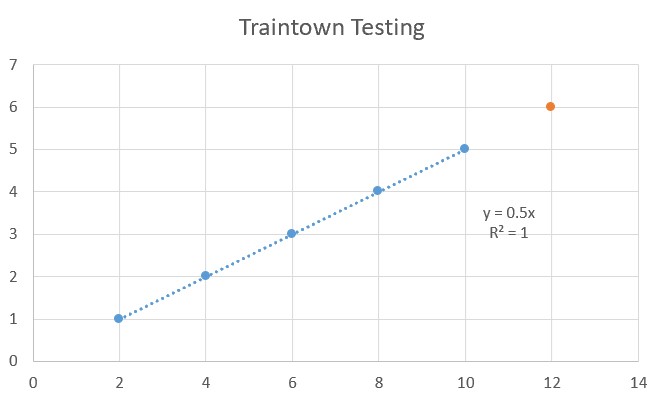
****

**Discussion for the choice of test data you provided and a table detailing the tests and the outcomes:**

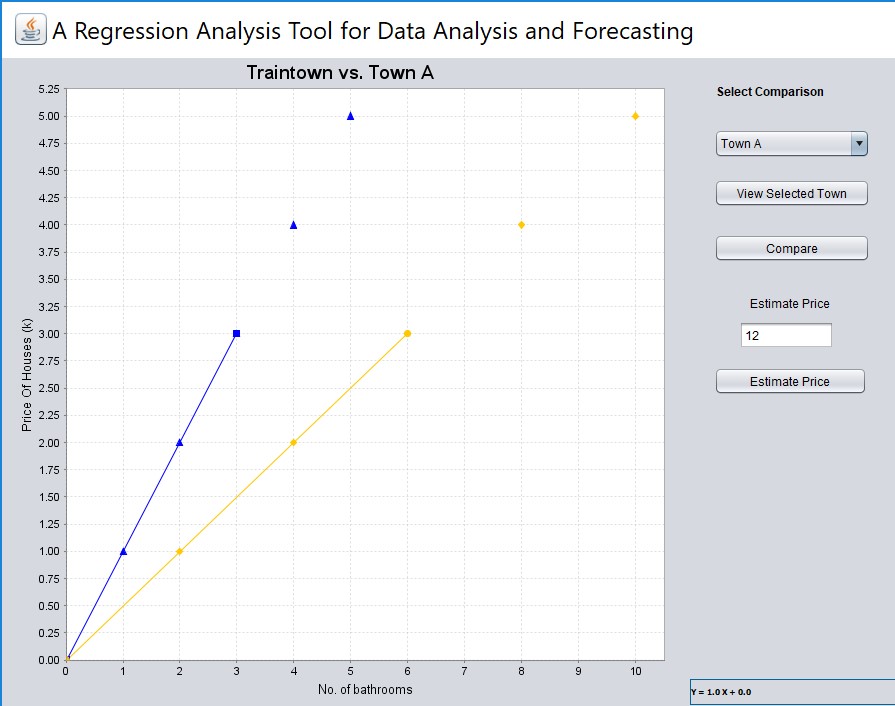
**Image A showing test data on our program with prediction:**

****

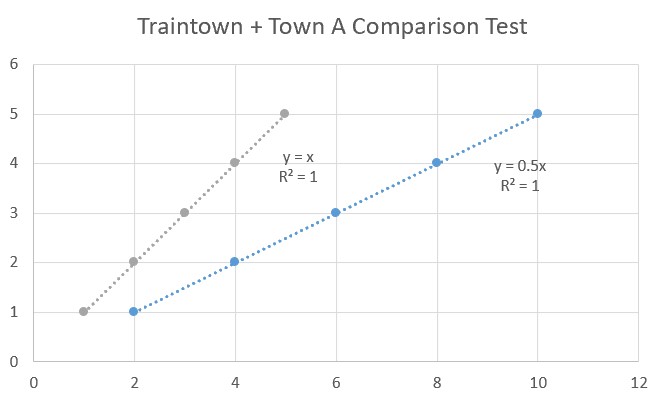
**Image B showing the test data plotted on a graph with prediction in excel:**

****

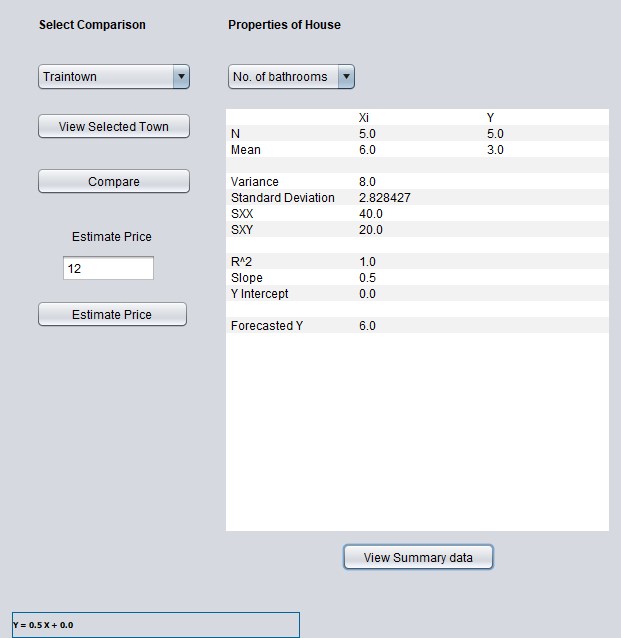
**Image C showing test data for Traintown and town A in our program for comparison:**

****

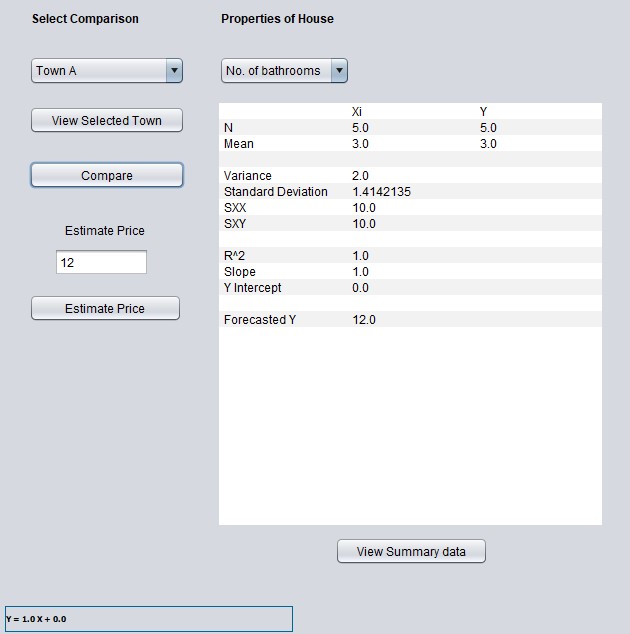
**Image D showing test data for Traintown and town A for comparison plotted in excel:**

****

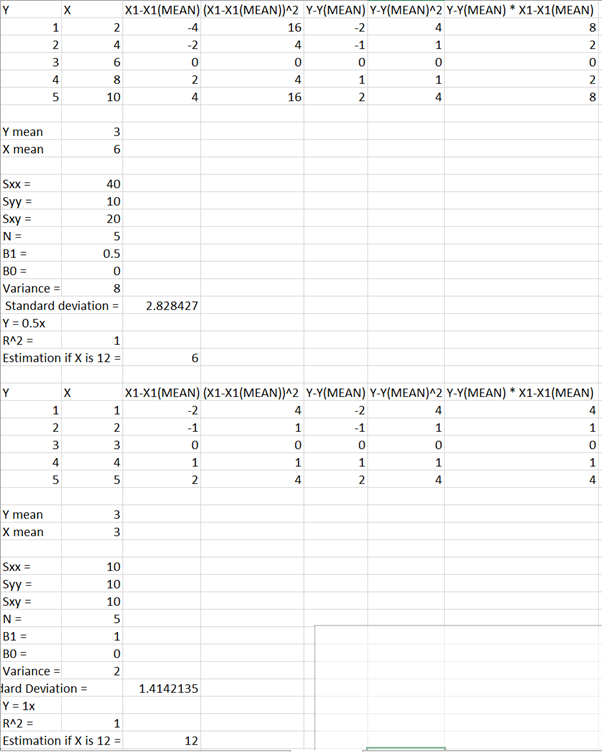
**Image E showing the statistical values calculated by our program for Traintown:**

****

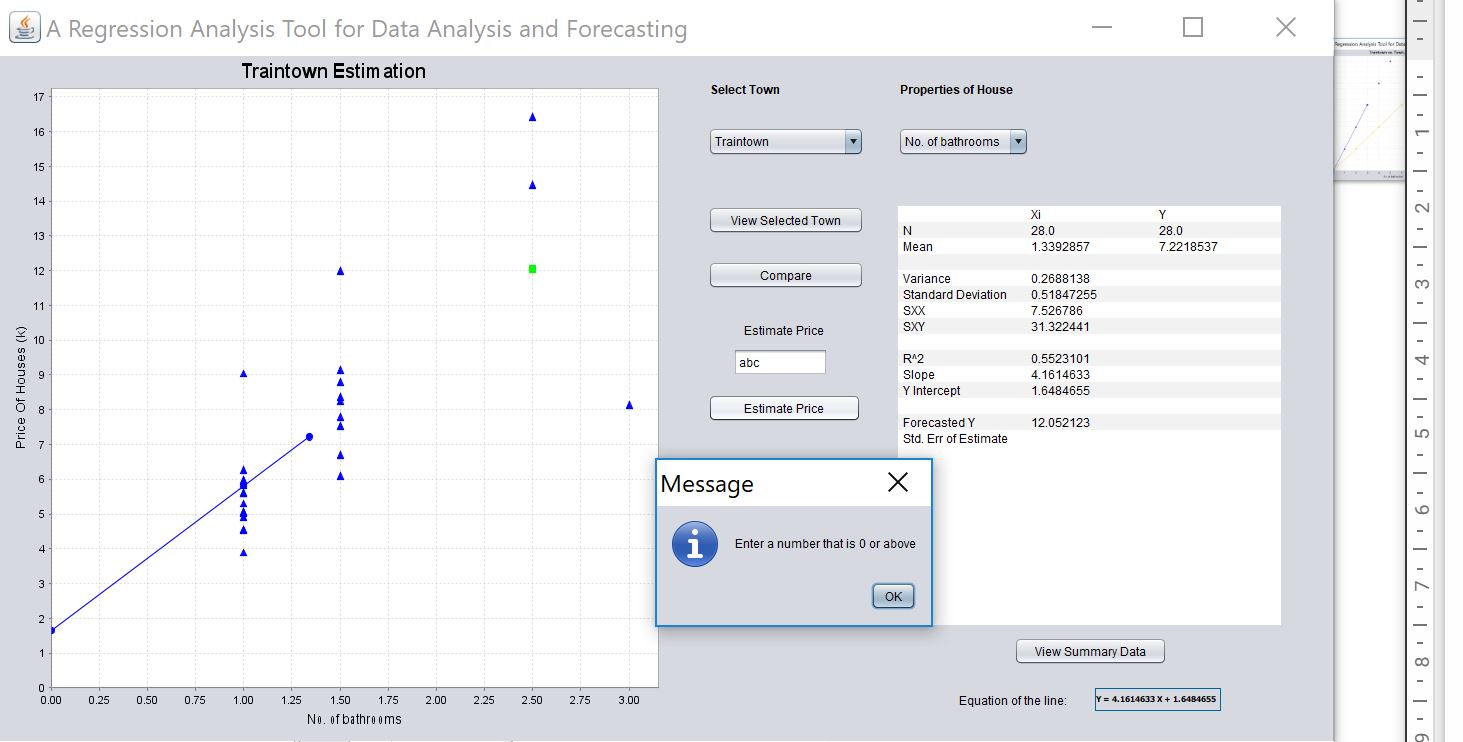
**Image F showing the statistical values calculated by our program for Town A:**

****

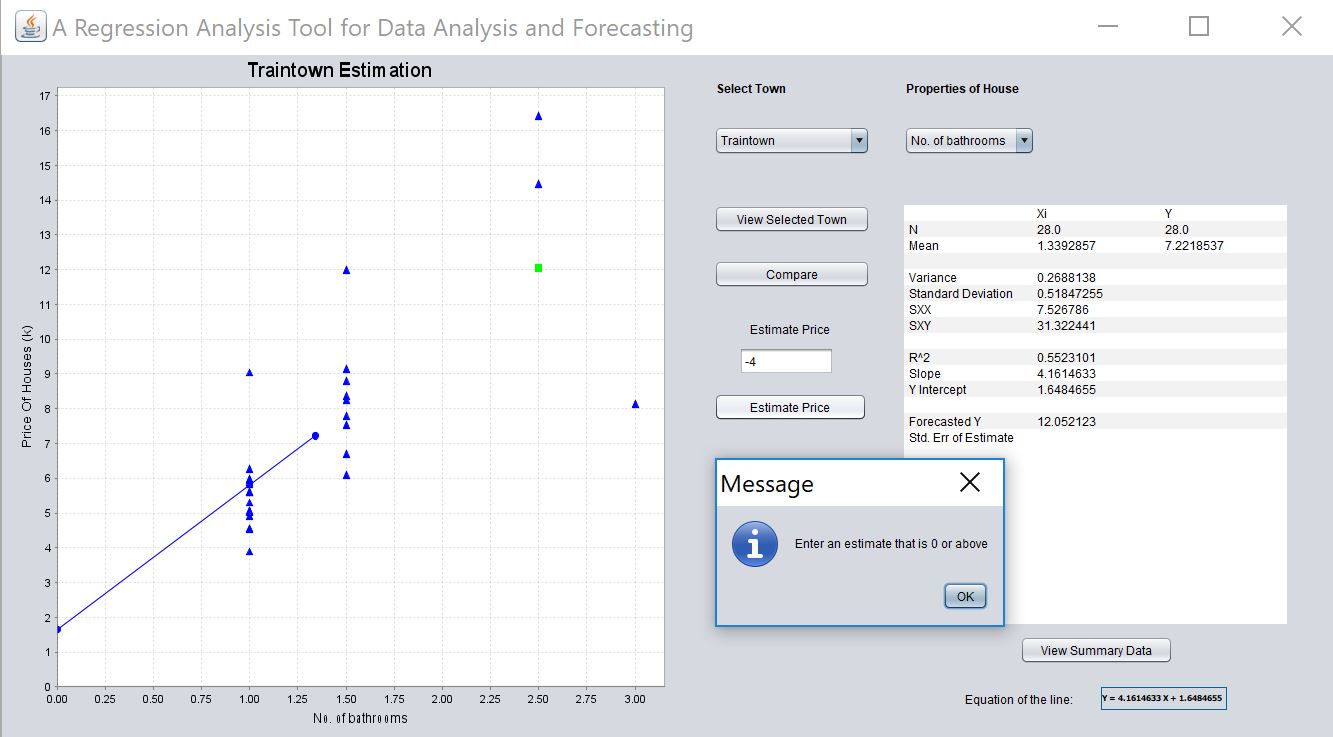
**Image G showing the calculations done in an Excel file containing test data:**

****

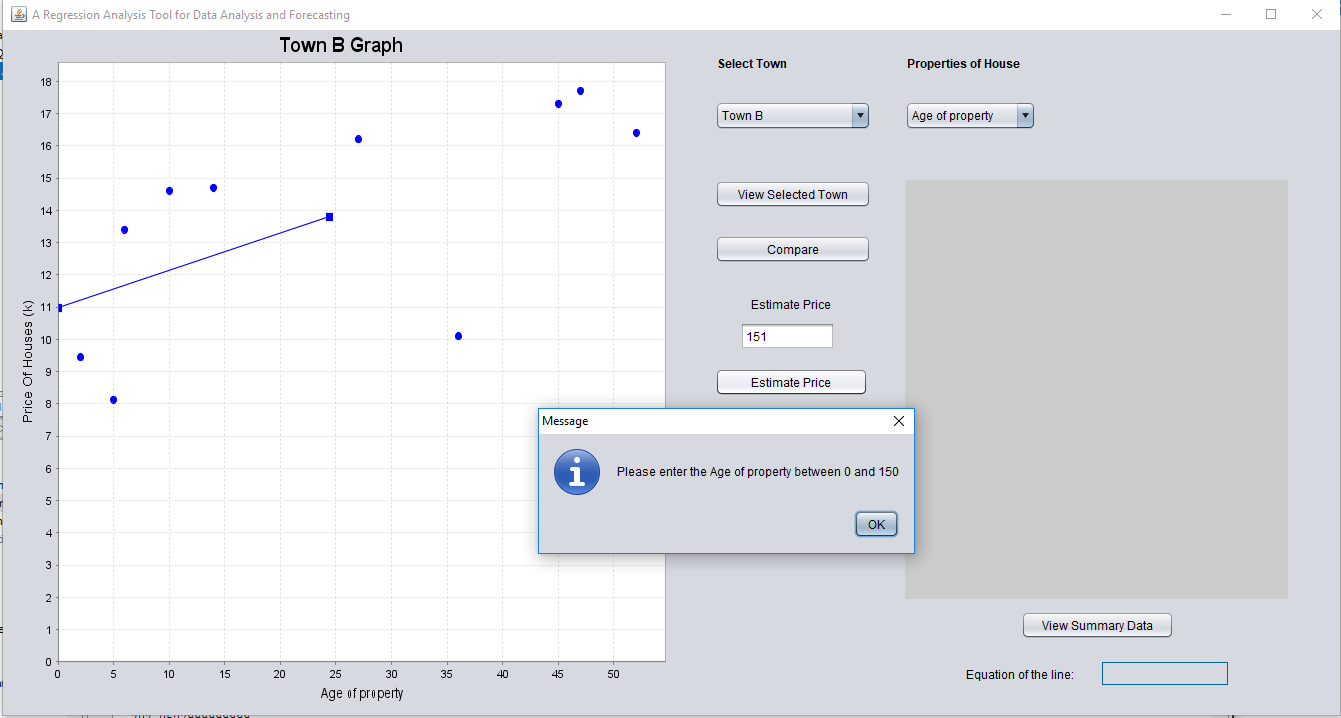
**Image H showing what occurs when the user types letters into the estimate price box:**

****

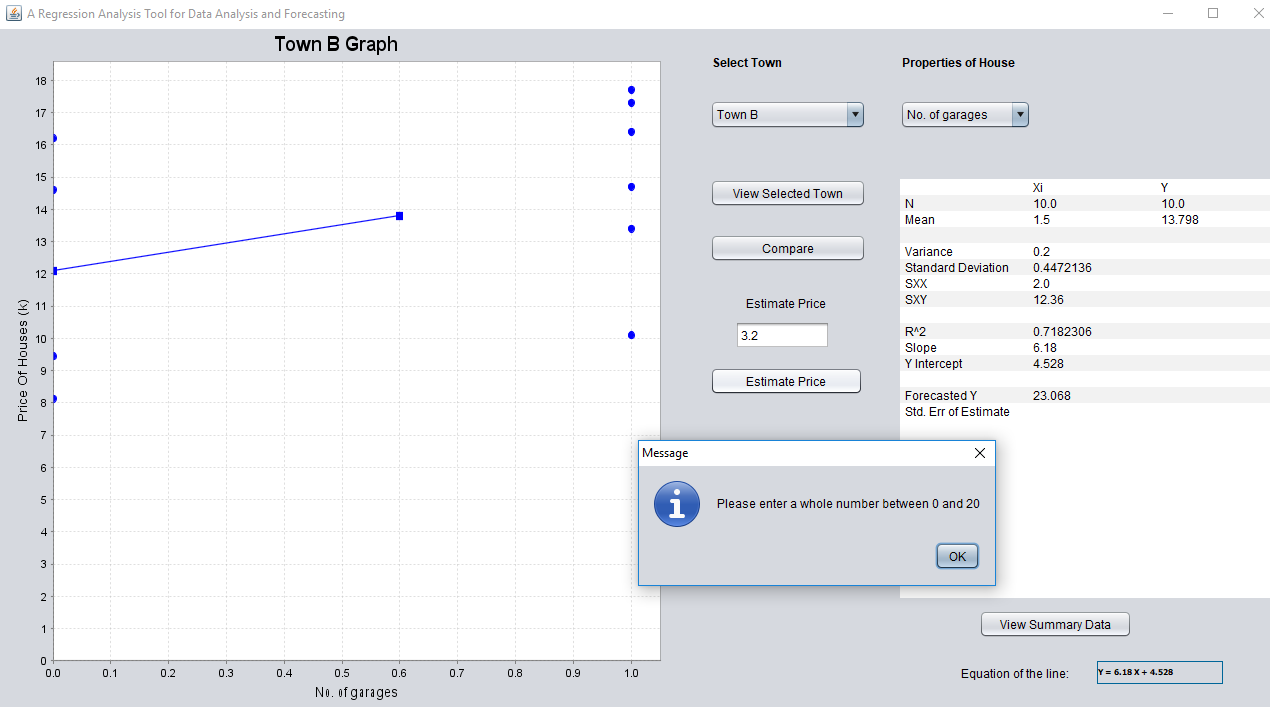
**Image I showing what occurs when the user types a negative number into the estimate box:**

****

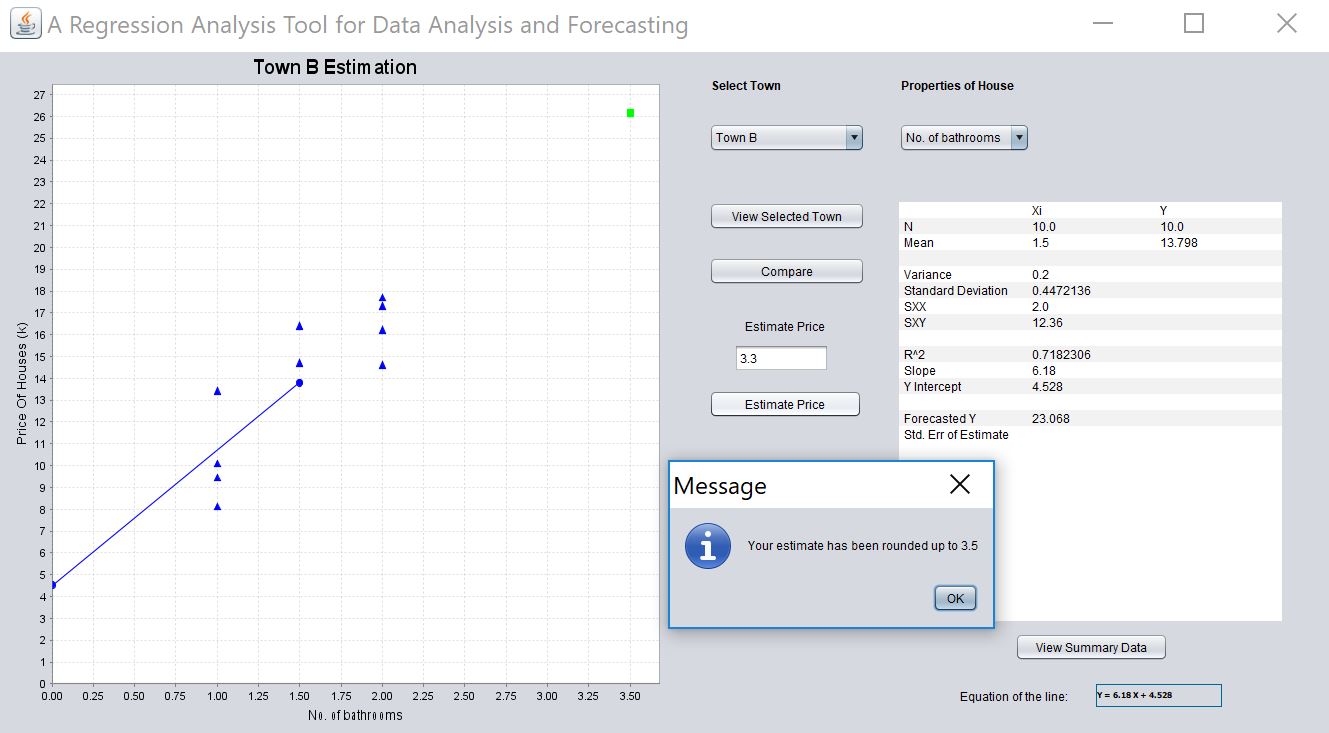
**Image J showing what occurs when a value outside the limit is entered:**



**Image K showing what happens when you enter a number which is not a whole number for variables X4, X5 and X6:**

****

**Image L showing what occurs when a value is typed in the estimate box for variable X2:**

****

We decided to do our testing using a different data set to what was provided. This would allow us to see if the program would still work if a different set of data was used. The values we ended up using for the test data can be seen in the tables in Image G. The first table provides the data for Traintown and the second table in the image provides the data for Town A. The reason we used these exact values is because it was easy to determine the equation of the line and the regression value will be 1. This allows us to easily compare the results created by our program to the manual calculations done in excel. We decided to only change the Y-values and the X1 values for Traintown and Town A since the code we wrote is the exact same for each of the variables and towns. If we change the X and Y values for the other towns, the program would work as expected.

We decided to begin our testing by producing an excel file, so we could calculate the values needed to work out the equation of the line and regression. Then, we created graphs using the test data and used the excel features to print the equation of the line and regression value, which can be seen in Images B, D and G. This provided us with accurate data in which we could compare the results produced by our program with. Then we replaced the original data provided to us with the test data within the programs text files for the X1 and Y values for Traintown and Town A.

As you can see, when comparing Image A with Image B, our program was able to plot the X and Y values correctly for Traintown. We also decided to test the prediction, which was plotted correctly. If you compare the values from the excel file in Image G with the table in Image E, all the calculated results produced by our program match. The same test was carried out for Town A, as shown in Image F, and produced the correct results. Another test was to compare Traintown with Town A. This is shown in images C and D. The graphs produced in our program are plotted correctly.

We also made sure that our program only accepts positive numbers when the user inputs a value to produce an estimate. Image H provides insight into what happens when any characters other than numbers are printed in the estimate price box. We decided not to allow letters as this does not represent a true value for our variables to predict a price for a house since all our data is numbers. Also, the program prevents the user from inputting negative numbers since it does not make sense to buy a property with -1 bathrooms which is demonstrated in image I. We also added a range to prevent the user from entering an unrealistic value in the estimate price box. For example, as shown in Image J, the limit for age of property is set to 150 years. If the user inputs any number greater than 150, they will get a pop-up message saying, “Please enter the Age of property between 0 and 150.” We set limits to the other variables which are 10, 20, 10, 20, 30, 15, 150 for X1-X7. We decided on these values as they seem realistic for each of the different variables. It also provides enough of a ceiling for extra data to be added later.

Another error check we did was preventing the user from inputting values that are not whole numbers. This is demonstrated in Image K. This occurs in the variables number of rooms (X5), number of bedrooms(X6) and number of garages(X4). We did this because it is unrealistic to purchase a house with 3.2 garages. The program will ask the user for a positive whole number. However, we allowed decimal values for the variables site of area(X2), living space(X3) and age of property(X7) because for those variables decimal values make sense. For the variable number of bathrooms(X2), we decided to round the number the user inputs to the nearest 0.5. Since the data provided to us includes some properties with 0.5 bathrooms (1.5, 2.5 etc). We integrated this into our own system to allow the user to input 0.5 bathrooms if desired. This is demonstrated in Image L.

**Analysis of results for the comparison data you were provided:**

**Traintown:**

Traintown consists of a collection of fairly old houses, with the average age being 36 years. The number of bathrooms in the houses are all similar, with most houses having between 1 and 2.5. With this data we can infer that most of the houses are medium to large family homes, as the average prices graciously increase as the average number of rooms and bedrooms increase. These homes are definitely for the middle to working class, more family oriented population based on the average price, number of rooms and bedrooms, which are £722,000, 7 and 3 respectively.

Using London as an example, the houses in the training data set would be synonymous with areas in between the centre of the city and the boroughs close to the outskirts. The site area for these homes (6460 square feet) is also significantly larger than the houses in Town B, but also smaller than the ones in the remaining towns, leading us to believe that they are not in the heart of the city. The fact that bigger houses can be bought at a much lower cost than most of the other towns means that this area would definitely be outside the central business district of that particular city. Our real world-comparison for this area would be places on the outskirts of of the city. Based on the concentric model, these houses would be located in the commuter zone.

**Town A:**

The data for Town A consists of a collection of mostly new houses, with the average age being 8.30 years old. The number of bathrooms in the houses are all similar, with most houses having between 2 and 3 bathrooms. With this data we can infer that most of the houses are large and new and as the prices tremendously increase as the number of rooms increase. Using London as an example, the houses in the Town A would be synonymous with areas near West-Central London. These homes are definitely for the wealthier, more family oriented population based on the average price, number of rooms and bedrooms, which are £1,300,000, 10 and 5 respectively. The site area for these homes (11,780 square feet) is also significantly larger than the houses in Town B, leading us to believe that they are not completely in the heart of the city, where it would be hard to aquire large pieces of land for personal properties. These houses are close to the central business district, but not completely enclosed by the area. Based on the concentric model, we can infer that these houses would be located in the factory/zone of transition. Our real world-comparison for this area would be places like Fulham.

The houses found in Town A are on average bigger than the houses found in Traintown. This is evident due to the average number of rooms being 10, whereas in training town it is 7. This directly correlates to the price of a property in Town A to be significantly more expensive than a house in training town. The average price of a house in Town A is roughly £1,251,000 compared to the £722,000 for a house in training town. The reason we came to the conclusion that Town A would be located in west-central areas of London is because the average house prices in Fulham is roughly £1,138,667. This average is quite similar to the average house price in Town C, so it is safe to assume the houses from Town C will be located in areas near Fulham.

**Town B:**

From the data provided, we have come to the conclusion that majority of the properties in Town B are studio flats in urban areas located in Central London. This due to the fact that the average number of bedrooms is 2 and the number of bathrooms is 1.5. This suggests most of the flats are small. Also, the prices of the properties presented in the data are expensive when comparing it to the size of the property. A property of similar size in this location will cost you almost twice as much as a house just outside Central London. Another reason why the properties are from Central London is because quite a few do not have a garage. This is due to limited space available in the centre of London. Also, there is a significant increase in price as the number of rooms increases. This suggests that larger properties are very desirable in this area as Central London is one of the most desirable places to live in the UK. Another trend we recognised was that as the age of the property increases, so does the price of the property. This does not follow the conventional trend of the price of house decreasing as the property gets older. From this, I can infer that the area holds a lot of historical and cultural value. Based on the concentric model, we can infer that these houses would be located in the central business district.

When comparing Town B with Traintown, we realised that houses located in town B are significantly more expensive. For example, if you was to purchase a 3 bedroom house located in Town B it will cost roughly £1,600,000 compared to the £660,000 you will pay in Traintown. This significant increase in price suggests that Town B will be located closer to Central London and training town will be in the outskirts of London. Also, most of the properties contain 1-2 bedrooms with an average living space and site area of 1290 square feet. Most properties in City of London are generally small due to limited space available within the centre of the city.

**Town C:**

Town C consists of mostly large houses with many rooms and bedrooms located on the outskirts of London. This is because the price of the properties are very low relative to their size. On average, there are 11 rooms and 5 bedrooms. This suggests that the houses are quite large, however the prices are low. The fact that bigger houses can be bought in this town at a much lower cost than most of the other towns means that this area would be outside the central business district of the city. Due to the average number of rooms being high, it is safe to assume that most of the houses are detached homes as they usually contain multiple rooms. Another value that supports this claim is that all the houses have a minimum of 1 garage. From all of this we can infer that Town C is located further away from Central London. Based on the concentric model, we can infer that these houses would be located in the residential zone.

Overall, the houses found in Town C are cheaper than the houses from the training data. This suggests that the location of Town C is further away from central london compared to training town because as you go further away from the centre of the city, house prices tend to decrease. Another reason why we believe this is because the site area and living space is greater on average compared to Traintown. From this I can infer that the houses are bigger, containing more rooms and bedrooms. Although the houses are larger than the houses found in training data, they are still cheaper. This suggests that the area is less desirable than Traintown. From this I can tell that the location of Traintown is closer to the central business district than Town C. People are more likely to pay extra to live in Traintown if they work in the city to reduce their commute time.

**Critical evaluation of the choice and performance of data structures:**

The chosen data structures for our program are Arrays and ArrayLists. ArrayLists are a java implementation of Linked List. Therefore, at the most basic level our chosen data structure are Arrays and Linked Lists. Initially the algorithm we devised required the use of multiple arrays but we slowly implemented the use of ArrayLists as well as we did not have to define the size of the array, allowing for more dynamic data storage. The use of arrays was still justified however, as the use of JFree Chart could only be implemented under those conditions.

Time complexity of an algorithm refers to the amount of time taken by an algorithm to run. Some of the factors that affect runtime are: Hardware, operating system and processors. The time taken to run our algorithm using the Netbeans profiler was 223 ms. The space complexity of an algorithm or data structure is the maximum amount of space used at any one time, ignoring the space used by the input to the algorithm.

Using a profiler in the Netbeans IDE it allowed us to see the size of the live instances of our data structures when we ran the algorithm. The ArrayList used 27,336 bytes (0.1%) of space and contributed to 0.4% of the live instances in the program while, the Array used 59,336 bytes (0.2%) of space and contributed to 0.3% of the live instances in the program. From this we can infer that our algorithm performs as expected.

**Individual contribution by each team member and personal reflections:**

|  |  |  |
| --- | --- | --- |
| Name | Student ID | Contribution |
| Kamilah Agbaje | 001004321 | 25% |
| Maruf Hoque | 001006731 | 25% |
| Rathusan Jeyaseelan | 000885570 | 25% |
| Trevor Kiggundu | 001001720 | 25% |

Reflection:

**References:**

**Foxtons Group plc and Foxtons Estate Agents (2018) Fulham, *Fulham Property Market Trends* , London Estate Agents, [online] Available at: https://www.foxtons.co.uk/living-in/fulham/ (Accessed November 26, 2018).**

***Gilbert, D. (2008) The JFreeChart Class Library, Version 1.0.9, Developer Guide, Object Refinery Limited***

***Gilbert, D. and Morgner, T. (2000) JFreeChart, SourceForge, [online] Available at: https://sourceforge.net/projects/jfreechart/files/1. JFreeChart/ (Accessed November 26, 2018).***

***Lang, V. and Junior (2017) What is the Concentric Zone Model?, World Atlas, Worldatlas, [online] Available at: https://www.worldatlas.com/articles/what-is-the-concentric-zone-model.html (Accessed November 26, 2018).***

***Oxford University Press (2017) Central business district, Dominant social paradigm - Oxford Reference, Oxford University Press, [online] Available at: http://www.oxfordreference.com/view/10.1093/oi/authority.20110803095558550 (Accessed November 26, 2018).***

**SuzanneKn (n.d.) *Concentric Zone Model*, *Concentric zone model*, Wikipedia, [online] Available at: https://en.wikipedia.org/wiki/Concentric\_zone\_model (Accessed November 27, 2018).**