1009 Java OOP Project Report – Team Work

Student Name	Student Number
Ng Kiang Ann Roysten	1801228
Muhammad Najib Bin Bahmat	1800874
Chua Wei Ye	1801554
Tan Kah Wei	1800769
Chua Woi Zhao	1801961
Yong Jian Ming	1801586

Table of Contents

Objectives

To create a Web Crawler application to crawl data about MRT breakdowns from two different platforms that is heavily based on Object Oriented Programming(OOP) design principles.

OOP Design Principles

In the webcrawler package, the extractor classes uses abstraction to share the functionality needed for the subclasses. It requires a concrete model type to be defined in order to extend the class. The use of abstraction rather than interface in this case is due to the fact that getData() method is overloaded and contains a default method to call the abstract method that has to be implemented in the subclass. The type returned in getData() method is based on the concrete model type defined in the subclass.

TrainServiceDisruptionSheetExtractor and TwitterTweetsSearchExtractor class extends the Extractor class. They are extended with TrainServiceDisruptionData and TwitterTweetData as the model type respectively. These two classes will download and scrap data from a webpage and return useful information to the user. The information will be stored in a json file respectively and read the data from there to reduce the bandwidth and the time it needs to display the information to the end user.

TrainServiceDisruptionData, TwitterTweetData and TwitterSearchData are used to encapsulate the data and methods used in our program.

Limitations

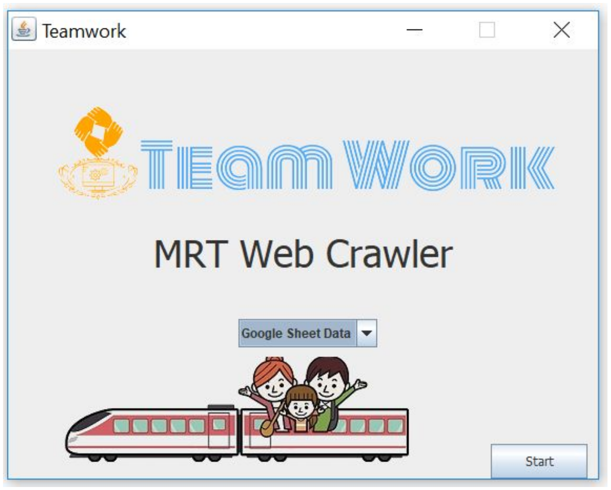
1) As the data that we are crawling is based on MRT Breakdowns, users are not able to specifically crawl customized data. Only MRT breakdown data is crawled.
2) When the data is downloaded to a JSON file, it will take time for the program to download and no progress download bar is implemented. So the user have to wait till the program successfully finished the download in order to proceed.
3) For the dataset from Google Sheet, the data available is only up to 2017. It does not contain current 2019 data.
4) In addition to point 3, the data crawled are not based on real-time data. On top of that, due to the ever-increasing data available on the web, it is tedious to process the specific required data out from the respective sites.
5) For the twitter data, we used a non API call to retrieve tweets. Twitter is actively blocking the use of such API and thus it does not work in the long run. The algorithm that we used to generate a random User-Agent generate a random key of 30 characters. This is just a way to avoid blocking real browsers and made it work for this particular project.

Technologies Used

Technology	Description
Windows builder(Graphics User Interface)	WindowBuilder is built as a plug-in to Eclipse and the various Eclipse-based IDEs (RAD, RSA, MyEclipse, JBuilder, etc.). The plug-in builds an abstract syntax tree (AST) to navigate the source code and uses GEF to display and manage the visual presentation.
JSoup	Jsoup is a Java library for working with real-world HTML. It provides a very convenient API for extracting and manipulating data, using the best of DOM, CSS, and jquery-like methods.
JFree Chart	JFreeChart is popular for its efficient chart creation and user-friendly installation setup. In this project, we will be implementing JFreeChart in the GUI. User can view the data analyzed from the web crawler in chart representation, providing better insights for research purposes.
Jcommon	A Java class library used to support JFreeChart. With the aid of serialization utilities, user interface classes for displaying information about applications and various custom layout managers.
Json	Json uses lightweight data-exchange format. Text can be read easily and used as data format in Java object oriented language. Hence, we are using it in our project mainly for storing data crawled from websites.
Gson	Gson library is a tool to serialize/deserialize objects into json format and vice versa. It can handle collections such as ArrayList<> as long as it is given a concrete type during compile time. Java known limitation thus far is the type has to be determined in compile time rather than run time or deserialization would fail.

GUI Overview

1. Home Page



This is the main page of our application. From here, users can select either “Google Sheet Data” or “Tweet Data” then, click on “Start” button to begin crawling.*Warning - Be patient as it will take some time for the application to load the data into the system.

2. Google Sheet Page

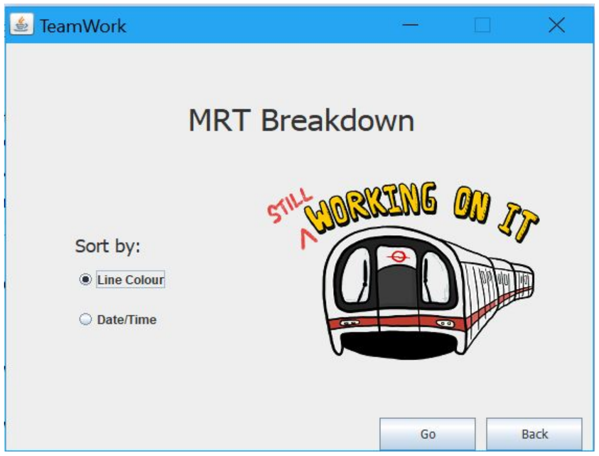
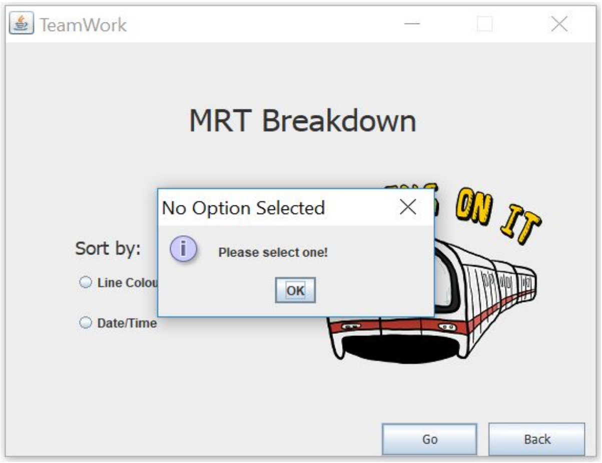
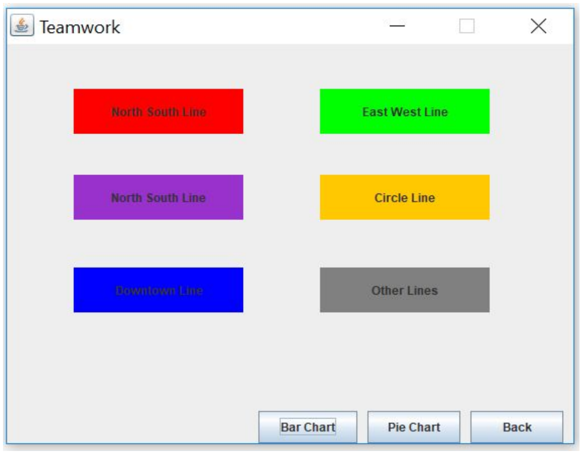


Figure 2.1



In this page, the data extracted is from the Google Sheet. The data is sorted according to two types, by Line Color(Purple, Yellow, Blue etc.) and Date/Time. The user is able to select one of these two types and proceed to the next page. An error message will pop up if none of the option is chosen (See Figure 2.1).

3. Train Line Legends and Charts



In the next page after the user selected Line Colour as their option, a legend will be shown to indicate train lines with respective colours. Users can either display the data in bar chart (see Figure 3.1) or pie chart (see Figure 3.2). In addition, individual chart will be saved as a downloaded image upon clicking the buttons. Also, a “Back” button is also available for users to return back to Google Sheet Page.

Figure 3.1

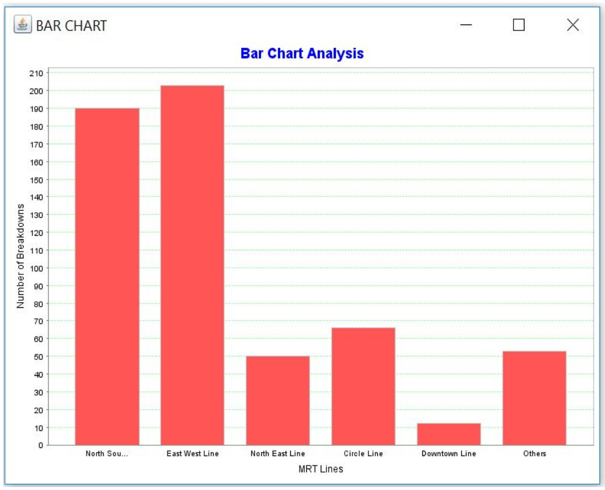
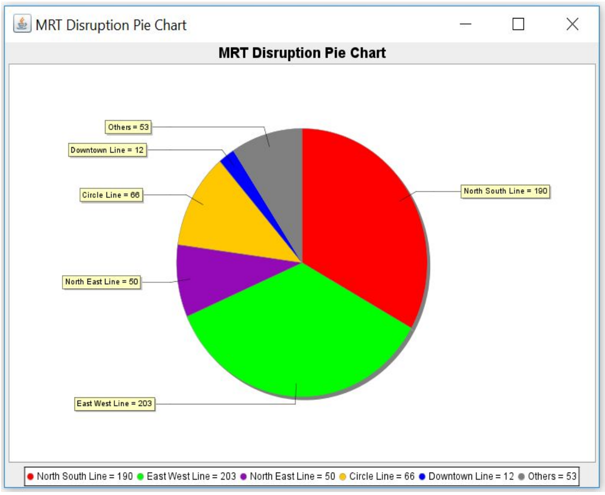
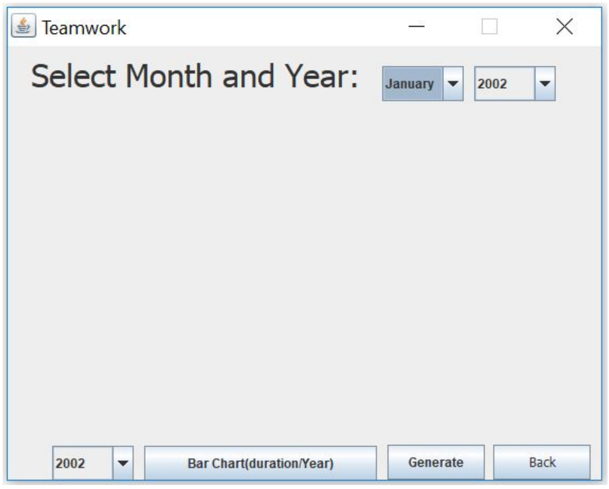


Figure 3.2



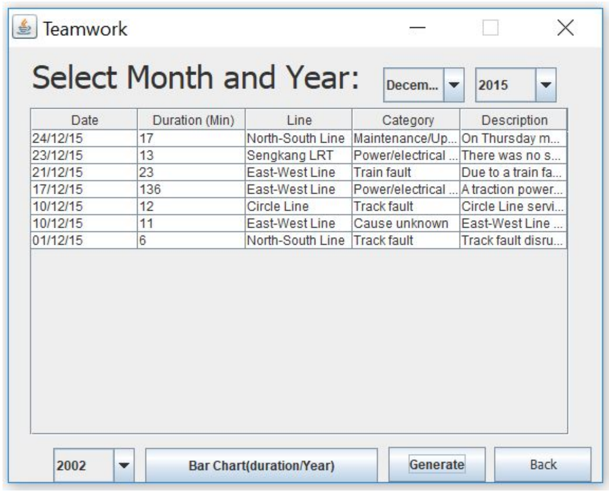
The bar chart from Figure 3.1 shows the number of breakdown occurred over the years from respective train lines. Figure 3.2 shows the similar results but in represented in pie chart.

4. Date / Time Menu and Charts



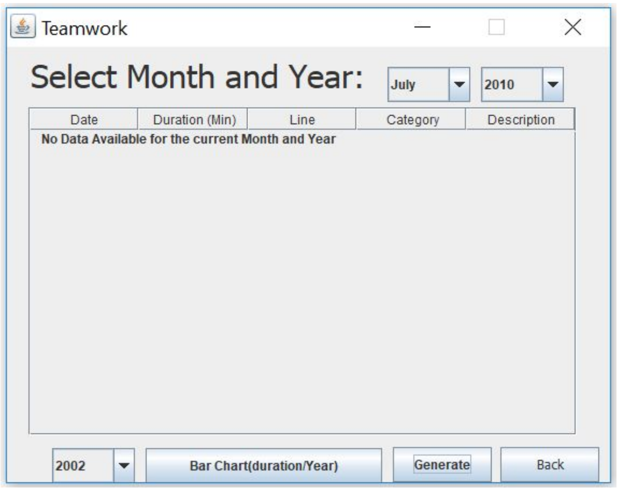
First of all, this will be page for users to view a selection of data represented in either date format or by the duration of train fault within the selected year.

Figure 4.1



Date	Duration (Min)	Line	Category	Description
24/12/15	17	North-South Line	Maintenance/Up...	On Thursday m...
23/12/15	13	Sengkang LRT	Power/electrical ...	There was no s...
21/12/15	23	East-West Line	Train fault	Due to a train fa...
17/12/15	136	East-West Line	Power/electrical ...	A traction power...
10/12/15	12	Circle Line	Track fault	Circle Line servi...
10/12/15	11	East-West Line	Cause unknown	East-West Line ...
01/12/15	6	North-South Line	Track fault	Track fault disru...

Figure 4.2

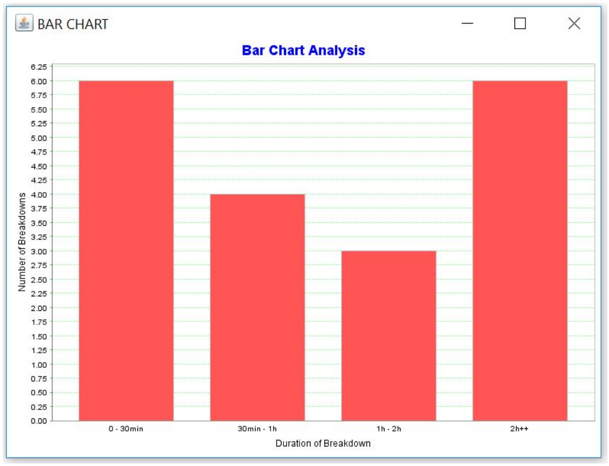


Date	Duration (Min)	Line	Category	Description

No Data Available for the current Month and Year

User is able to select the month and year from the dropdown list at the top right corner, and by pressing the Generate button at the bottom, the jtable will display the details of the breakdowns that happen at that specific month. If there is no breakdowns at that specific month, an error message No Data Available for the current Month and Year” will be displayed to notify the user.

Figure 4.3



User is also able to view the Bar Chart analysis of the data by year basis. Firstly, the user will select the year dropdown list at the bottom left of the application as shown in Figure 4.1, which user will then click the Bar Chart button right beside it. The program will then generate a bar chart analysis as shown above showing the occurrence and duration of the breakdown.

5. Twitter Data

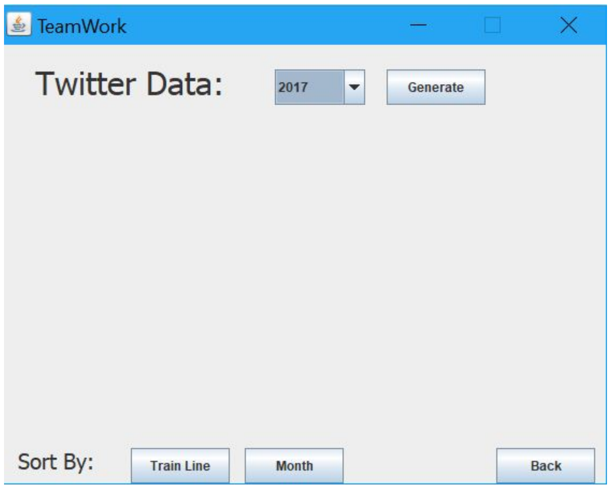
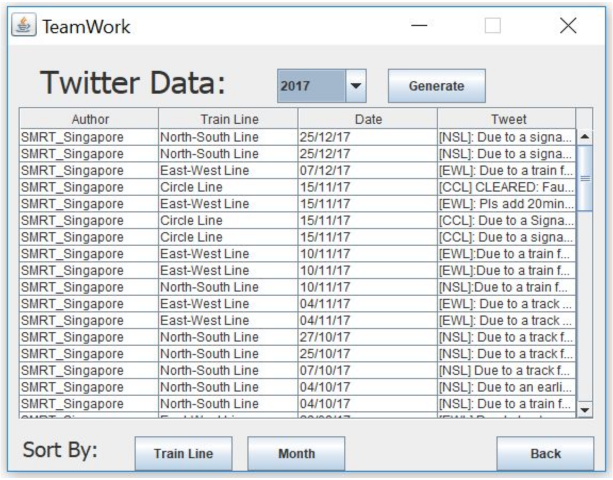


Figure 5.1



Author	Train Line	Date	Tweet
SMRT_Singapore	North-South Line	25/12/17	[NSL]: Due to a signa...
SMRT_Singapore	North-South Line	25/12/17	[NSL]: Due to a signa...
SMRT_Singapore	East-West Line	07/12/17	[EWL]: Due to a train f...
SMRT_Singapore	Circle Line	15/11/17	[CCL] CLEARED: Fau...
SMRT_Singapore	East-West Line	15/11/17	[EWL]: Pls add 20min...
SMRT_Singapore	Circle Line	15/11/17	[CCL]: Due to a Signa...
SMRT_Singapore	Circle Line	15/11/17	[CCL]: Due to a signa...
SMRT_Singapore	East-West Line	10/11/17	[EWL]:Due to a train f...
SMRT_Singapore	East-West Line	10/11/17	[EWL]:Due to a train f...
SMRT_Singapore	North-South Line	10/11/17	[NSL]:Due to a train f...
SMRT_Singapore	East-West Line	04/11/17	[EWL]: Due to a track ...
SMRT_Singapore	East-West Line	04/11/17	[EWL]: Due to a track ...
SMRT_Singapore	North-South Line	27/10/17	[NSL]: Due to a track f...
SMRT_Singapore	North-South Line	25/10/17	[NSL]: Due to a track f...
SMRT_Singapore	North-South Line	07/10/17	[NSL] Due to a track f...
SMRT_Singapore	North-South Line	04/10/17	[NSL]: Due to an earli...
SMRT_Singapore	North-South Line	04/10/17	[NSL]: Due to a train f...

From this page, users can navigate across various functions within the page, such as retrieving data from Twitter by selecting the year from the combobox and hit on the “Generate” button. Next, the jtable will be triggered in response to the user’s request. As shown in Figure 5.1, the Twitter data are presented methodically in the jtable.

Figure 5.1

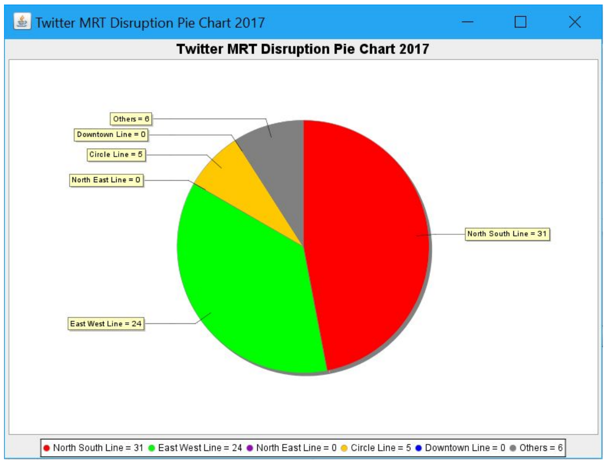
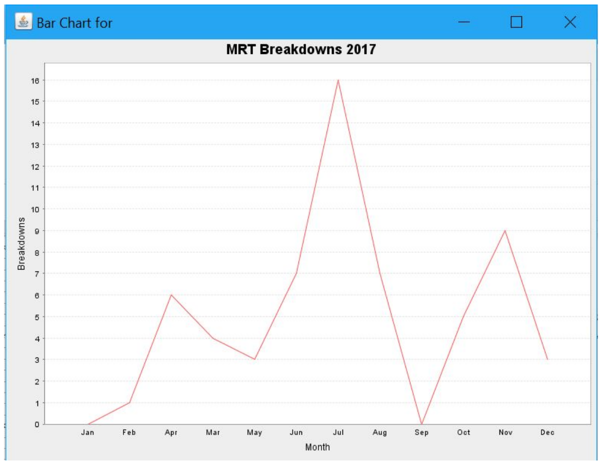


Figure 5.2



Furthermore, users may view the data based on their individual preferences. By clicking on the “Train Line” button, the data retrieved earlier in the jtable will be sorted according to the number of MRT disruption appearing on Twitter from respective train lines in a pie chart (See Figure 5.1). Alternatively, the data can also be manipulated in a line graph, indicating monthly MRT breakdown depending on the selected year (See Figure 5.2).

Team Contributions and Reflections

1) Chua Wei Ye (1801554)

   <u>MainGUI.java</u>
   - GS Pie Chart
   - Twitter Pie Chart
   - Calling data from GS extractor and Twitter extractor to display pie chart

   <u>TrainServiceDisruptionData.java</u>
   - Create variables to store and sort the crawled data from google sheet..
   - Extended by the extractor class to make use of the variables created.

   Reflection:
   Through this project, I gained more knowledge on the implementation of OOP using java programming language and the flow of the whole project. The complexity of trying to link every classes together, making sure there's no conflict between different variables of the same name. The uses of data classes to create variables and make the program more organised and efficient

2) Chua Woi Zhao (1801961)

   <u>TwitterTweetsSearchExtractor.java</u>
   - Filtering of crawled twitter data; remove repeated entries for display
   - Extracted train service from crawled data

   Filtering of crawled data is done with the assumption that no train service would have more than one breakdown in a day. The filtering code requires standardized input from the TwitterTweetData.getTweetTrainLine() method, thus line 156 to 171 is necessary. It also makes the output data look nicer. Filtering works by identifying the train line in current iteration of crawled data and setting its flag to true. When flag of respective train lines are true, their data would not be displayed, thus only storing one entry of train breakdown for that train line, in that day. The flags are reseted when the date changes.

   Reflection:
   After this project, i learnt how to program in OOP style and understand the purpose and need for OOP. I also learnt how to crawl data by selecting its html tag and working with regular expression (regex). I have learnt much from my teammates about the usage of APIs, generic types, and the creation of the GUI. This project have provided me with valuable experience in working as part of a development team.

3) Muhammad Najib Bin Rahmat (1800874)
<u>MainGUI.java</u>
   - JTable to display Data
   - Sorting of data crawled(GoogleSheet)
   - Error Popup message
   - Project Integration

   <u>TwitterTweetData.java</u>
   - Data structure for Twitter Data
   - Blueprint for Twitter Data object

   Reflection:
   Throughout the project, I have learnt that OOP design principle is useful in designing a project that have multiple objects working together. I have also learnt how crawling data from websites can be useful in solving real-world problems. Data is crawled through specific searches or keywords to provide up-to-date data of the specific sites. This is often used by search engines like Google.

4) Ng Kiang Ann Roysten (1801228)
<u>MainGUI.java</u>
   - Line Chart (twitter data)
   - Sorting the data crawled into a chart to display useful information(based on GUI buttons)

   <u>Extractor.Java</u>
   - saveIntoCache() - to save the crawled data into a json file.

   <u>Video Making</u>

   Reflection:
   I realise that time management is very important during a project,and in order to finish a project well,we need to work well with our teammates.Also,i learned that OOP is a good concept to know to code realistically to analyse data and to come out with good coding practices and programs.Especially when it comes to data analysis and web crawling,if we do not keep to the standard practises of OOP our codes will be messy and it will also be longer than one that keeps to the practices.

5) Tan Kah Wei (1800769)
   MainGUI.java
   - Bar Chart (Google Sheet)
   - Images and designing of GUI

   TwitterTweetData.java
   - Create a blueprint for the crawler (e.g. variables)
   - Ensuring data are serialized in JSON
   - Sort and store tweets into the class based on the information crawled

   Reflection:
   I generally feel that this is a good opportunity for me to learn about data crawling because it is a norm. In order to complete the project, everyone has to play their part by adding meaningful values. Data crawling can be difficult at the beginning because there are many restrictions such as the fact that data can only be retrieved for the past seven days on Twitter. Hence, being able to overcome the first step of data crawling will be a great success. All in all, it can be very problematic to delegate the tasks because it will become very unorganized for the reader after integration of coding and there are additionally numerous constraints in our program regardless of achieving our primary objectives. With that, I sincerely appreciate everyone’s hard work.

6) Yong Jian Ming (1801586)
   TwitterTweetsSearchExtractor.java
   - getTwitterSearchData() Method (It reads the twitter json response and return TwitterSearchData object for futher extraction)
   - generateRandomUserAgent() Method (It is a method to avoid getting block by twitter or returning no result)

   TrainServiceDisruptionSheetExtractor.java
   - getData() Method (Download the website and using jsoup to process the DOM elements and store into TrainServiceDisruptionData class)

   Reflection:
   I have learn different ways to crawl a website. It is generally not easy due to the limitation put in place in many sites. Using OOP concept, I could come out with useful class to crawl specific sites. I have also learn new limitation within Java with generic types vs concrete types. Extractor class uses a special generic type to bind the return type for the methods in the class. The deserialization of the

json data for some reason could not resolve generic types in runtime and made debugging rather difficult. That results in the ugly code written to change the generic type to a concrete type for Java to resolve. After all, web scrapping is an interesting way for people to analyse raw data into meaningful data for organization to look into especially looking on how well they are doing, what are the best product and so on. This project touch onto that field which is a good experience for the future.