

**School of InfoComm Technology**

**Distributed Data Pipelines**

Diploma in Data Science (DS)

October 2022 Semester

**INDIVIDUAL ASSIGNMENT 2**

(40% of Distributed Data Pipelines Module)

**Deadline for Submission:**

**10th Feb 2023 (Friday), 2359 Hours**

|  |  |  |
| --- | --- | --- |
| Student Name | : | Ang Jing Yi Clairer |
| Student Number | : | S10223223K |

**Penalty for late submission:**

10% of the marks will be deducted every day after the deadline.

**NO** submission will be accepted after 17th Feb 2023, 23:59.

Contents

[1. Problem Statement Formulation 2](#_Toc126938378)

[2. Preparation 3](#_Toc126938379)

[3. API Call 4](#_Toc126938380)

[4. Data Display 6](#_Toc126938381)

[5. Enhancement (Optional) 11](#_Toc126938382)

[6. Summary 13](#_Toc126938383)

[7. Reflection 14](#_Toc126938384)

## ****Problem Statement Formulation****

In the second assignment, the user was tasked with demonstrating their knowledge of setting up an Alteryx workflow. The goal of the assignment was to determine the next three bus arrivals for each bus service. For this case, using the school’s BusStopCode (12101).

To accomplish this task, the user would need to gather information about bus services, such as bus routes and schedules, from a reliable source. This information could be obtained from a public transportation agency website or a transportation data provider.

Once the user had obtained the necessary data, they would need to import it into Alteryx and clean and prepare the data for analysis. This may involve removing duplicates or irrelevant data, transforming data into a usable format, and making any necessary calculations.

Next, the user would need to create an Alteryx workflow that would allow them to analyze the bus data and determine the next three bus arrivals for each bus service. This could involve using tools such as the Filter tool, Join tool, and Formula tool to manipulate and analyze the data.

Finally, the user would need to output the results of their analysis in a readable format, such as a table or a visualization. This could be done using tools such as the Tableau tool, or the Browse tool in Alteryx.

Overall, the assignment required the user to have a solid understanding of the Alteryx platform, as well as the ability to effectively gather, clean, and analyze data in order to meet the objectives of the assignment.

This is the overall workflow:

Diagram

Description automatically generated

The user will next dive into each section of the workflow and explain.

## ****Preparation****

The first section of the workflow is the Instantiate Parameters for API Call.

A screenshot of a computer

Description automatically generated with medium confidence

What this section does is that the user first set up the parameters to call for the API.

Text

Description automatically generated with medium confidence Icon

Description automatically generated

Table

Description automatically generated Diagram

Description automatically generated with low confidence

Graphical user interface, text, application, chat or text message

Description automatically generated Application

Description automatically generated with medium confidence

Table

Description automatically generated Diagram

Description automatically generated with low confidence

The Text Input tool in Alteryx allows the user to enter the parameters they want to find from the API (BusStopCode: 12101). This tool is a flexible way for the user to input data into the workflow, and it can be used to gather information about the desired bus stops for which the next three bus arrivals are to be determined.

Once the user has entered the desired parameters into the Text Input tool, it is then connected to the Select Tool. In the Select Tool, the user changes the data type of the BusStopCode column from its original format into a V\_String data type. This is an important step because the API that is being used to gather the bus arrival information may not be able to process the BusStopCode if it is in a different format.

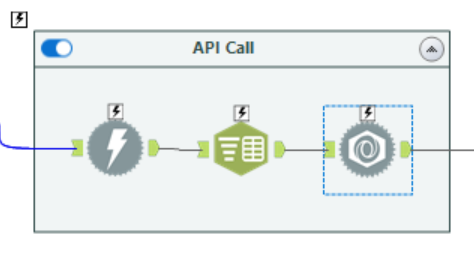
After the BusStopCode has been transformed into a V\_String data type, it is then connected to the Formula Tool. In the Formula Tool, the user sets up the connection to find the desired BusStopCode from the API. This involves writing a formula that will allow Alteryx to communicate with the API and retrieve the relevant bus arrival information.

Once the Formula Tool has retrieved the bus arrival information, it is then connected back to the Select Tool. In the Select Tool, the user unchecks the BusStopCode column. This step is necessary because the user may not want to see the BusStopCode in the final output, and it also helps to keep the output data as concise and relevant as possible. Ignore the orange sections as those columns were previously deleted.

Overall, the Text Input tool, Select Tool, and Formula Tool work together to provide the user with the desired bus arrival information from the API.

## ****API Call****

The second section of the workflow is to Download BusStopCode Data and Extract.



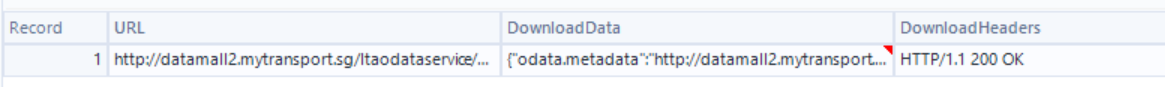
What the second section does is that the user sets up the API Call to allow themselves to be able to access the data.

Graphical user interface, text, application, email

Description automatically generated Graphical user interface, text, application

Description automatically generated A picture containing diagram

Description automatically generated



Graphical user interface, text, application

Description automatically generated Diagram

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated A picture containing diagram

Description automatically generated

Graphical user interface, text, application

Description automatically generated

The Download Tool in Alteryx is a powerful tool that allows the user to retrieve information (BusStopCode: 12101) from a website using an API. The first step in using the Download Tool is to set up the user's API account key. This account key is essentially a unique identifier that gives the user access to the information they want to retrieve from the API.

After the API account key has been set up, the user then inputs the API URL into the Download Tool, along with the API account key. The user also sets the "Accept" row to "OK". This is important because it ensures that the API connection has been established and that the user has access to the information they want to retrieve.

Once the API connection has been established, the Download Tool retrieves the data from the API and outputs it into the workflow. This data is then connected to the Text to Columns Tool, which splits the data into rows using the delimiter "\n". This step is necessary because the API data is often in a format that is not easy for the user to read or manipulate.

Finally, the split data is then connected to the Parse Tool. The Parse Tool allows the user to make changes to the data, such as selecting only specific columns or transforming data from one format to another. After the changes have been made, the Parse Tool outputs the transformed data, which can then be used for further analysis or visualization.

Overall, the Download Tool, Text to Columns Tool, and Parse Tool work together to provide the user with the ability to retrieve information from a website using an API, manipulate the data into a usable format, and make changes to the data as needed.

## ****Data Display****

The third section of the workflow is to display the data to the user’s liking.

A picture containing timeline

Description automatically generated

What the third section does is that after retrieving the information of the BusStopCode (12101) the user wished to explore, the user then cleanse the data and formats them to display the final look of it.

But before doing so as we can see from the above image, before the container named ‘Displaying of final data’, there is 2 tools the user used to do a final cleaning of the data retrieved before exploring.

Graphical user interface, application, table

Description automatically generated Diagram

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated Graphical user interface, application

Description automatically generated with medium confidence

Table

Description automatically generated

The user's approach to cleaning the data in Alteryx before analyzing it is a crucial step in ensuring that the information they retrieve from the API is accurate and relevant. In this particular case, the user used the Select Tool and the Filter Tool to clean the data before further processing.

The Select Tool was used to uncheck the URL and DownloadHeaders columns in the data. This is important because these columns contain unneeded information that would just clutter the output and make it harder for the user to analyze the data. By unchecking these columns, the user was able to remove them from the output, leaving only the relevant information that they needed.

After using the Select Tool, the user then used the Filter Tool to present only the non-null values from the JSON\_ValueString column. This step was necessary because the API data often contains null values, which can skew the results of any analysis performed on the data. By filtering out the null values, the user was able to ensure that their analysis was based on accurate and relevant information.

The steps taken by the user to clean the data before analyzing it were critical to the success of their project. By removing unneeded information and filtering out null values, the user was able to produce accurate and meaningful insights into the data, helping them achieve their objective of finding the next three buses arrival for each bus service.

Graphical user interface, text, application

Description automatically generated Graphical user interface, text, application

Description automatically generated

Graphical user interface, table

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated Diagram

Description automatically generated with medium confidence

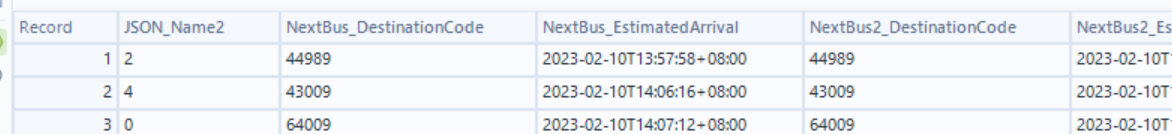
Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated Graphical user interface, table

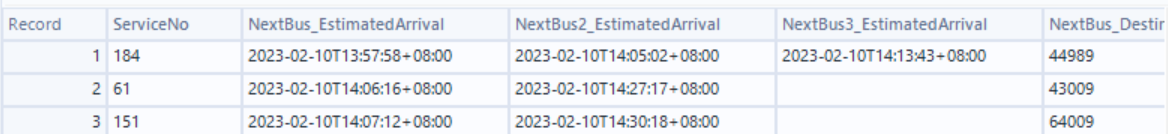
Description automatically generated



Table

Description automatically generated A picture containing table

Description automatically generated



By filtering the data in this way, the user was able to create a more manageable and readable table that would allow them to see the desired information about the next three buses for each bus service. This made it easier for the user to analyze the data and draw insights from it.

The Text to Column Tool was used to split the JSON\_Name column into three separate columns, which allowed the user to separate the data into different categories and make it easier to analyze. This step was important because without splitting the data into separate columns, it would have been more difficult to see patterns and trends in the data.

After splitting the JSON\_Name column, the user then used the Cross Tab Tool to concatenate the selected columns into a single table. This step was necessary because it allowed the user to bring together all of the relevant data into a single table, making it easier to analyze and compare the data.

Finally, the user used the Select Tool to select the columns they wanted to show in the final table. By doing this, they were able to choose only the most relevant information to display, which helped to simplify the output and make it easier to understand.

The user then used the Browse Tool to show the final table from the workflow. This allowed them to see the results of their analysis and ensure that the data was accurate and relevant.

Overall, the user's approach to presenting the final results of their analysis was thorough and effective, making it possible for them to draw meaningful insights from the data and achieve their objective of finding the next three buses arrival for each bus service.

Overall Final Table:

Graphical user interface, application

Description automatically generated A picture containing graphical user interface

Description automatically generated

## ****Enhancement (Optional)****

In the enchance section, the user manage to set up manual input so that users who wants to find out other BusStops bus services arrival time, they are able to see the next three arrival of the bus from that bus stop.

Diagram

Description automatically generated Chart, diagram

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated Graphical user interface

Description automatically generated with low confidence

Graphical user interface, text, application

Description automatically generated Diagram

Description automatically generated

A picture containing diagram

Description automatically generated

The user set up the Text Box Tool to provide a user-friendly interface for users to easily input the BusStopCode they want to retrieve the bus arrival information. The Action Tool then updates the BusStopCode value in the Text Input Tool which is then connected to the workflow, starting from the Select Tool, that was described earlier. This way, the user is able to retrieve the desired bus arrival information for the inputted BusStopCode, without the need to manually change the BusStopCode in the workflow every time. The output will then be presented in output log in the running as Analytic app as API Output. Below are the steps to do and how it will look like and where you can find the output.

Step 1: Run the workflow using Analytics App

Graphical user interface, text, application

Description automatically generated

Step 2: Input the Correct BusStopCode and click Finish

Graphical user interface, text, application

Description automatically generated

Step 3: Click on Show Output Log

Graphical user interface, text, application, email

Description automatically generated

Step 4: Output Shown:

A screenshot of a computer

Description automatically generated with medium confidence

These are the 4 steps when doing this enchancement. However, if the wrong input is made such as null values and wrong BusStopCode that does not exist, you will not be able to see the output but it will show open debug. Unfortunately this enhancement has its advantage of being able to input your desired Bus Stop Code, however, the disadvantage is it is unable to produce an error message particularly to show why the Bus Stop Code is wrong such as (Error: Bus Stop Code is null/ Error: Bus Stop Code does not exist). This steps will only be as followed only if the Bus Stop Code exist.

## ****Summary****

In summary, in the Alteryx workflow, the user aimed to retrieve the next three bus arrivals for each bus service. The Text Input tool was used to allow the user to input the desired BusStopCode. This value was then updated using the Action Tool and connected to the Select Tool, Formulae Tool, and Select Tool to change the BusStopCode into the required format for the API connection. The Download Tool was used to connect to the API using an API account key and retrieve the desired information. The data was cleaned by unchecking unnecessary columns in the Select Tool, filtering out null values, and selecting specific columns to show in the final table. The final table was presented in the Browse Tool. An enhancement was made by adding a Text Box Tool for users to manually input the BusStopCode, which updates the value in the workflow, making it easier for users to retrieve the desired information.

Text

Description automatically generated with medium confidence

Graphical user interface, text, application, Teams

Description automatically generated

## ****Reflection****

In addition to researching the Error Message Tool, the user can also improve the solution by considering other ways to handle invalid inputs. For example, the user can add a validation step to check if the input BusStopCode is valid and only proceed with the workflow if it is. This can be achieved by comparing the input BusStopCode with a pre-defined list of valid BusStopCodes, or by using a RegEx formula to check if the input matches the expected format.

Another improvement the user can consider is to add a dynamic drop-down list to the Text Box Tool, so that users only need to select the BusStopCode from a list of available options. This can be achieved by using the Get Data tool to retrieve a list of BusStopCodes and linking it to the Text Box Tool.

Additionally, the user can also consider adding a feature to display a detailed list of bus services and estimated arrival times for each bus stop. This can be achieved by adding a Filter Tool to the workflow, to select the desired bus stop, and a Summarize Tool to aggregate the information for each bus service.

By continuously refining and improving the solution, the user can make the Alteryx workflow more robust, user-friendly and efficient for end-users to obtain valuable insights from the API data if they had the time.

Reflecting on this Alteryx workflow, it highlights the importance of data preparation and manipulation in the data analysis process. The use of different tools in Alteryx, such as the Text Input, Select, Filter, Text to Column, and Cross Tab tools, showed how versatile Alteryx can be in transforming raw data into meaningful information. The enhancement of the Text Box Tool also showed how Alteryx can be made user-friendly, allowing for easy and efficient data retrieval. Overall, this Alteryx workflow demonstrated the capability of the platform in streamlining the data analysis process and presenting valuable insights to users. The user can learn more if the user had researched more on the different tools that may have helped in this Assignment rather than sticking to the few Tools that have been taught in class.