

**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 | : | Lim Wee Liang Kelven |
| Student Number | : | S10221788K |

**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

[Problem Statement Formulation 3](#_Toc126920543)

[Preparation 3](#_Toc126920544)

[Text Input Data tool 3](#_Toc126920545)

[Formula tool 4](#_Toc126920546)

[API Call 5](#_Toc126920547)

[Download tool 5](#_Toc126920548)

[Select tool 7](#_Toc126920549)

[Data Cleansing tool 8](#_Toc126920550)

[JSON Parse tool 9](#_Toc126920551)

[Filter tool 9](#_Toc126920552)

[Data Display 10](#_Toc126920553)

[Text to Columns 10](#_Toc126920554)

[Cross Tab 11](#_Toc126920555)

[Multi-Field Formula 12](#_Toc126920556)

[Select 13](#_Toc126920557)

[Sort 14](#_Toc126920558)

[Frontend Display 14](#_Toc126920559)

[Text Box 15](#_Toc126920560)

[Action 15](#_Toc126920561)

[Browse tool 16](#_Toc126920562)

[Summary 18](#_Toc126920563)

[Findings 18](#_Toc126920564)

[“Getting Started Learning Path” Completion 19](#_Toc126920565)

[Alteryx Foundation Micro-Credential Certificate 20](#_Toc126920566)

[Reflection 20](#_Toc126920567)

# Problem Statement Formulation

As Singapore strives towards being a car-lite society, more commuters will use public transport like buses to travel. And as more commuters use buses, the demand to know bus arrival times increases. A pipeline needs to be built to provide real-time bus arrival times for a given bus stop. By not knowing when buses arrive, commuters may have to wait under hot or wet weather for a long time, miss buses, or be unable to plan their travel time effectively. If buses are unattractive due to long wait times, commuters would not use them.

This problem can be seen across Singapore across all ages from students to workers. The problem occurs more often during peak hours of the morning and evening. This problem also affects the government as they want to incentivise commuting using public transport.

This report aims to explain the pipeline development process using Alteryx Designer. These processes include:

* Preparation
* API Call
* Data Display
* Frontend Display

# Preparation

Diagram, timeline

Description automatically generated

Before loading the data, a model must be prepared to provide the appropriate API call based on the bus stop code input. This is accomplished using the **Text Input** and **Formula** tools.

## Text Input tool



There are 2 types of input tools: **Input Data**, and **Text Input**. Input Data is used to connect to files (i.e., Excel, CSV, XML, etc.) or databases. Since the pipeline connects to a website, this tool is not applicable, and the Text Input Data tool is used instead.

1. From the **In/Out** tab, drag the **Text Input** tool from the toolbar onto the canvas.
2. In the configuration window, add 3 columns: **BusStopNumber**, **url**, and **User-Agent**. Graphical user interface, text, application, email

   Description automatically generatedGraphical user interface, text, application, email

   Description automatically generated
   1. **BusStopNumber**: Input the appropriate bus stop for the model to provide the arrival time (i.e. 12101).
   2. **url**: The website where the bus arrival times are taken from (i.e., <http://datamall2.mytransport.sg/ltaodataservice/BusArrivalv2>).
   3. **User-Agent**: Mozilla is used to access the website.



The output should look like this.

## Formula tool



1. From the **Preparation** tab, drag the **Formula** tool from the toolbar onto the canvas and connect it to the **Text Input** tool.

Graphical user interface, text, application

Description automatically generated

1. Click on the tool. In the configuration window, select **url** from the dropdown menu under **Output Column**.

Graphical user interface, text, application

Description automatically generated

1. Enter the formula **[url] + "?BusStopCode=" + ToString([BusStopNumber])**.



The output should look like this.

# API Call

Timeline

Description automatically generated

After preparing the website link, the data is downloaded and transformed into the appropriate format before displaying it. This is accomplished using the **Download**, **Select**, **Data Cleansing**, **JSON Parse**, and **Filter** tools.

## Download tool



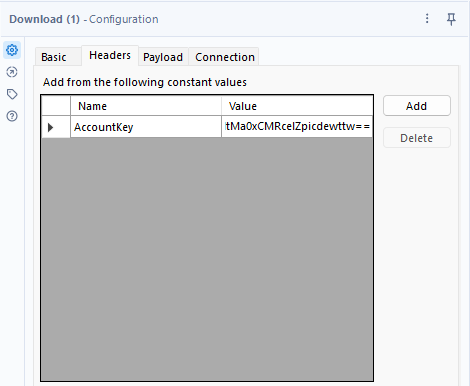
The **Download** tool is used to retrieve the information from the website.

1. From the **Preparation** tab, drag the **Download** tool from the toolbar onto the canvas and connect it to the **Formula** tool.

Graphical user interface, text, application, email

Description automatically generated

1. Click on the tool. In the configuration window 🡪 **Basic** tab 🡪 URL Box 🡪 ensure “**url**” is selected.



1. Under the **Headers** tab, add an API key.
   1. Click on the **Add** button.
   2. For **Name**, enter **AccountKey**.
   3. For **Value**, enter **HtMa0xCMRcelZpicdewttw==**



1. Still in the **Headers** tab, select **BusStopNumber**, **User-Agent**, and **Dynamic or Unknown Fields**.



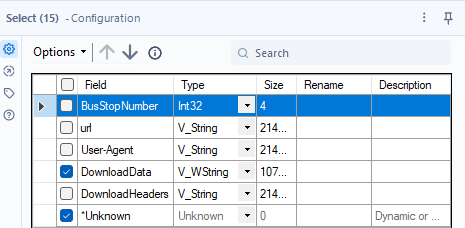
The output should look like this.

## Select tool



The **Select** tool is used to filter out redundant columns.

1. From the **Preparation** tab, drag the **Select** tool from the toolbar onto the canvas and connect it to the **Download** tool.



1. Click on the tool. In the configuration window, select **DownloadData** and **\*Unknown** from the list of columns.



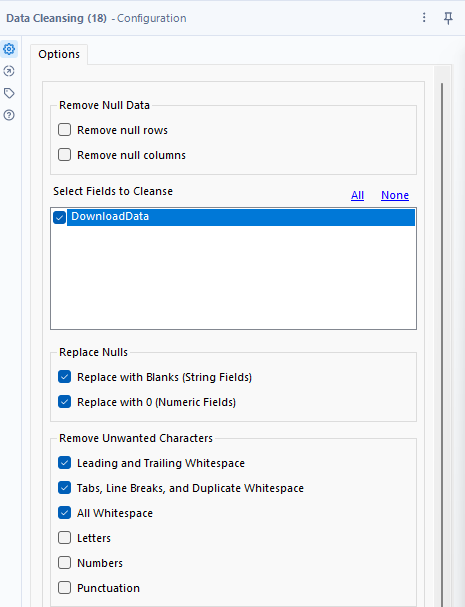
The output should look like this.

## Data Cleansing tool



The **Data Cleansing** tool is used to remove any present null values or whitespaces.

1. From the **Preparation** tab, drag the **Data Cleansing** tool from the toolbar onto the canvas and connect it to the **Select** tool.



1. Click on the tool. In the configuration window,
   1. Under **Select Fields to Cleanse**, select **DownloadData**.
   2. Under **Replace Nulls**, select **Replace with Blanks (String Fields)** and **Replace with 0 (Numeric Fields)**.
   3. Under **Remove Unwanted Characters**, select **Leading and Trailing Whitespace**, **Tabs, Line Breaks, and Duplicate Whitespace**, and **All Whitespace**.



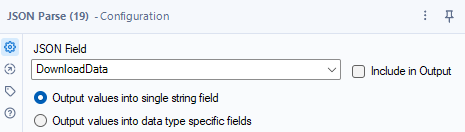
The output should look like this.

## JSON Parse tool



The **JSON Parse** tool is used to remove all curly brackets, splits the data into rows by colon, and separates the key and value of each row.

1. From the **Preparation** tab, drag the **JSON Parse** tool from the toolbar onto the canvas and connect it to the **Data Cleansing** tool.



1. Click on the tool. In the configuration window, select **DownloadData** from the **JSON Field** dropdown menu and select **Output values into single string field**.

Table

Description automatically generated

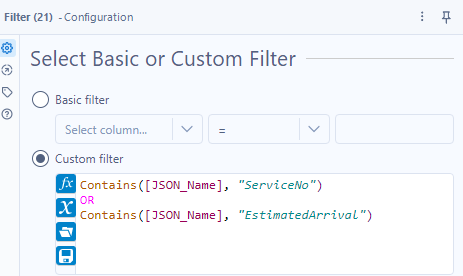
The output should look like this.

## Filter tool



The **Filter** tool is used to filter out redundant columns.

1. From the **Preparation** tab, drag the **Filter** tool from the toolbar onto the canvas and connect it to the **JSON Parse** tool.



1. Click on the tool. In the configuration window, select **Custom Filter** and enter the formula **Contains([JSON\_Name], "ServiceNo") OR Contains([JSON\_Name], "EstimatedArrival")**.

Graphical user interface, application

Description automatically generated

The output should look like this.

# Data Display

Graphical user interface

Description automatically generated with medium confidence

After downloading the data, more filtering and cleansing are done to display the bus service number and estimate arrival time. This is accomplished using the **Text to Columns**, **Cross Tab**, **Multi-Field Formula**, **Select**, and **Sort** tools.

## Text to Columns



The **Text to Columns** tool is used to remove and split the JSON\_Name column by “.”

1. From the **Parse** tab, drag the **Text to Columns** tool from the toolbar onto the canvas and connect it to the **Filter** tool.

Graphical user interface, text, application, email

Description automatically generated

1. Click on the tool. In the configuration window,
   1. Under **Columns to split**, select **JSON\_Name** from the dropdown menu.
   2. In the **Delimiters** box, enter “**.**” (period/full stop)
   3. Select **Split to columns**.
   4. Under **Number of columns**, select **3**.
   5. Under **Extra characters**, select **Leave extra in last column** from the dropdown menu.

Graphical user interface, text, application

Description automatically generated

The output should look like this.

## Cross Tab



The **Cross Tab** tool is used to pivot the column values into column names and select new values for the columns.

1. From the **Transform** tab, drag the **Cross Tab** tool from the toolbar onto the canvas and connect it to the **Text to Columns** tool.

Graphical user interface, application, email

Description automatically generated

1. Click on the tool. In the configuration window,
   1. Under **Group data by these values**, select **2**.
   2. Under **Change Column Headers**, select **3** from the dropdown menu.
   3. Under **Values for New Columns**, select **JSON\_ValueString** from the dropdown menu.
   4. Under **Method for Aggregating Values**, select **Concatenate**.
   5. In the **Separator** box, select “**,**”.

Text

Description automatically generated

The output should look like this, but the EstimatedArrival columns may be different.

## Multi-Field Formula

Graphical user interface, text

Description automatically generated

Since the three columns, NextBus\_EstimatedArrival, NextBus2\_EstimatedArrival, and NextBus3\_EstimatedArrival, must be formatted and aggregated, the **Multi-Field Formula** tool is used as it performs transformations on all three columns as compared to the standard Formula tool where it only transforms one column.

1. From the **Preparation** tab, drag the **Multi-Field Formula** tool from the toolbar onto the canvas and connect it to the **Cross Tab** tool.

Graphical user interface, text, application

Description automatically generated

1. Click on the tool. In the configuration window,
   1. Under **Select Fields**, select **Text** from the dropdown menu.
   2. Select **NextBus\_EstimatedArrival**, **NextBus2\_EstimatedArrival**, and **NextBus3\_EstimatedArrival** from the list of columns.
   3. In the **Expressions** box, enter the formula **Replace(LEFT([\_CurrentField\_], 19), 'T', ' ')**.

Text

Description automatically generated

The output should look like this, but the EstimatedArrival columns may be different.

1. Drag a new **Multi-Field Formula** tool from the toolbar onto the canvas and connect it to the previous **Multi-Field Formula** tool.

Graphical user interface, text, application

Description automatically generated

1. Click on the tool. In the configuration window,
   1. Under **Select Fields**, select **Text** from the dropdown menu.
   2. Select **NextBus\_EstimatedArrival**, **NextBus2\_EstimatedArrival**, and **NextBus3\_EstimatedArrival** from the list of columns.
   3. In the **Expressions** box, enter the formula **DateTimeDiff([\_CurrentField\_], datetimenow(), "Minutes")**.

Application

Description automatically generated with low confidence

The output should look like this, but the EstimatedArrival columns may be different.

## Select



The **Select** tool is used to filter out redundant columns.

1. From the **Preparation** tab, drag the **Select** tool from the toolbar onto the canvas and connect it to the **Multi-Field Formula** tool.

Graphical user interface, table

Description automatically generated

1. Click on the tool. In the configuration window, select all but from the list of columns.
2. Rename the columns:
   1. **NextBus\_EstimatedArrival** 🡪 First Bus Arrival Time
   2. **NextBus2\_EstimatedArrival** 🡪 Second Bus Arrival Time
   3. **NextBus3\_EstimatedArrival** 🡪 Third Bus Arrival Time
3. Move **ServiceNo** to the top.

Table

Description automatically generated

The output should look like this.

## Sort



The **Sort** tool is used to sort the results by the **ServiceNo**.

1. From the **Preparation** tab, drag the **Sort** tool from the toolbar onto the canvas and connect it to the **Select** tool.

Graphical user interface, text

Description automatically generated

1. Click on the tool. In the configuration window,
   1. Under Name, select ServiceNo from the dropdown menu.
   2. Under Order, select Ascending from the dropdown menu.

Table

Description automatically generated

The output should look like this.

# Frontend Display

To create a frontend display, the workflow utilises **Interface** tools such as **Text Boxes** and **Actions** and the **Browse** tool. In using Interface tools, the workflow file type changes from Alteryx Workflow to Alteryx Apps.

Diagram

Description automatically generated

## Text Box



The **Text Box** tool is used to get the user to input the bus stop code.

1. From the **Interface** tab, drag the **Text Box** tool from the toolbar onto the canvas.

Graphical user interface, text, application, email

Description automatically generated

1. Click on the tool. In the configuration window, under **Enter the text or question to be displayed**, enter **Enter Bus Stop Number**.

## Action



The **Action** tool is used to update the **BusStopCode** column in the **Text Input** tool.

1. From the **Interface** tab, drag the **Action** tool from the toolbar onto the canvas and connect it to the **Text Box** and **Text Input** tool from the **Data Preparation** container.

Graphical user interface, text, application, email

Description automatically generated

1. Click on the tool. In the configuration window,
   1. Under **Select an action type**, select **Update Value** from the dropdown menu.
   2. Under **Value or Attribute to Update** 🡪 **Text Input** 🡪 **Data** 🡪 **r**, select the column with the bus stop code. In this example, it is the first column. When selecting, make sure it is highlighted.



To run the app, click on the wand beside the Run button.

Graphical user interface, text, application

Description automatically generated

If the interface is set up correctly, a new window should appear like this.

## Browse tool

Graphical user interface, text, application, email

Description automatically generated

Currently, there is no way to see the output when using the app. To resolve this, the **Browse** tool is used.

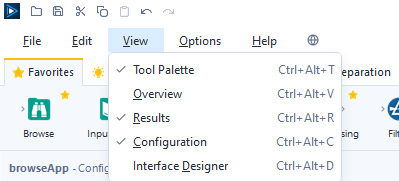
Timeline

Description automatically generated



The **Browse** tool is used to display the output table when using the app.

1. From the **In/Out** tab, drag the **Browse** tool from the toolbar onto the canvas and connect it to the **Sort** tool from the **Display Data container**.



1. Above the toolbar, select **View** 🡪 **Interface Designer**.

Graphical user interface, application

Description automatically generated

1. A new window should appear. Next, click on the last icon (Properties).

Graphical user interface, text, application

Description automatically generated

1. Select the **Browse** tool.

Table

Description automatically generated

1. If the **Browse** tool was configured correctly, the app output should be visible.

# Summary

## Findings

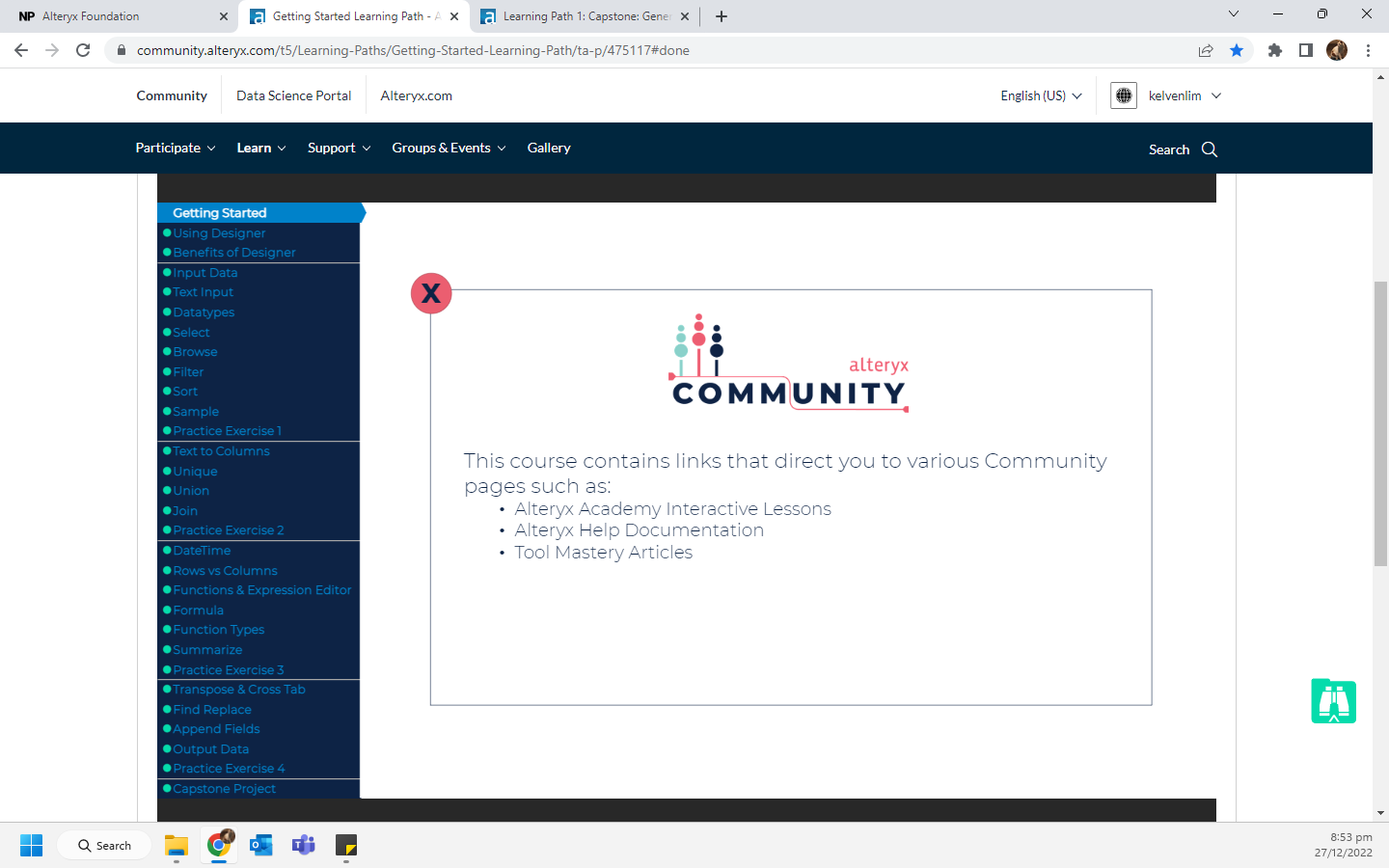
Most of this assignment can be referred to using week 13’s practical about the weather as the steps are uncanny.

When parsing the data using the **JSON Parse** tool, it was observed that there were many redundant rows like Operator, Latitude, Longitude, VisitNumber, etc. To filter the important rows (ServiceNo, EstimatedArrival), the **Filter** tool was used. The formula used was **Contains([JSON\_Name], "ServiceNo") OR Contains([JSON\_Name], "EstimatedArrival")**.

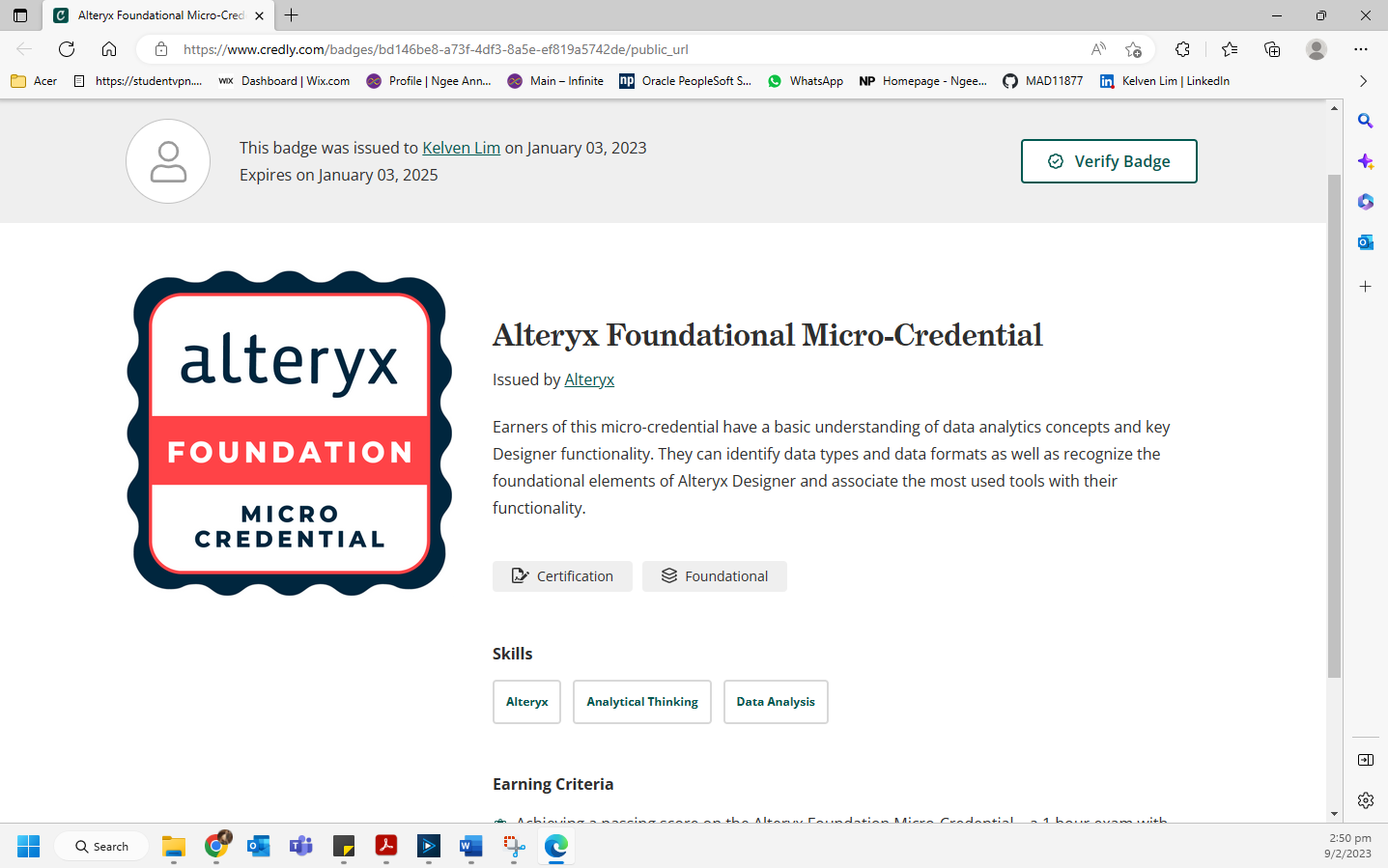
When transforming the data, it was found that the **Multi-Field Formula** tool can be more effective than the standard **Formula** tool. The **Formula** tool can only transform one column at a time, whereas the **Multi-Field Formula** tool behaves like a for-loop in programming; the tool goes through each selected column and transforms all of them. The EstimatedArrival columns were not in the proper DateTime format, there was an unknown “T” and +08:00 due to the time zone. To fix this, the **LEFT([\_CurrentField\_], 19)** function was used to remove +08:00. Next, the **REPLACE(“T”, “ “)** was used to replace the “T” with a space. Now, the EstimatedArrival columns had a proper DateTime format that can be used in calculations.

After some research, it was found that **Interface** tools were needed in creating the app. The **Text Box** tool was used to get the user input, and the **Action** tool was used to update the data before downloading from the website. Issues were encountered when attempting to display the results. After posting a question on Alteryx’s Community page, I found out that a **Browse** tool was needed, and changed some configurations.

## “Getting Started Learning Path” Completion



## Alteryx Foundation Micro-Credential Certificate



<https://www.credly.com/badges/bd146be8-a73f-4df3-8a5e-ef819a5742de/public_url>

# Reflection

If the website contained the bus stop names along with their number, it would be possible to show the users what the bus stop name is as compared to just showing them numbers. The website also contains information about where the bus service started (OriginCode) and ended (DestinationCode). The starting and ending bus stops are usually in bus interchanges next to the MRT. Hence, being able to show the OriginCode, DestinationCode, and their respective names would make the app more useful to the user.

Additionally, if the website contained information about the number of available seats, conditional formatting could be used using the Table tool. Buses with many available seats can be green, those with some available seats would be yellow, and those with little to no available seats would be red.

Through this assignment, I learnt how to use tools such as the Text Box, Action, and Multi-Field Formula which were neither taught in class nor the Getting Started Learning Path. I learnt how to create an app and perform calculations on multiple columns at once.

A skill I have learnt through this assignment was building a data pipeline using Alteryx. In another module, I learnt to create pipelines using Airbyte. Thus, learning a new pipelining method was educational and beneficial. My understanding of Alteryx Designer tools has deepened. Through Alteryx’s community, I have asked questions and reviewed questions by others to better understand my mistakes. Not only did it clear up misunderstandings, but I had also learnt to use new tools that were not taught in lessons.