

CourseDelivery

COSC2626 - Cloud Computing - Practical Problem-Solving Project

(By Aaron Chan - s3666603 & Jack Ryan s365590)

Contents

Project Link	3
Summary	3
Introduction	3
Related work	4
Software Design/Architecture	4
Pub/Sub	4
Dataflow	5
BigQuery	5
Datastore.....	5
Twitter API.....	5
Maps API.....	6
MySQL	6
AutoML Translation and Bucket.....	6
reCAPTHCA	6
Implementation	7
Flask	7
Twitter API.....	7
MySQL	8
BigQuery	9
PUB/SUB	10
DATAFLOW	11
AutoML Translation	13
DATASTORE	14
User manual	14
References	17
Google Cloud Services:	17
Pub/sub:	17
DataFlow:.....	17
autoML:.....	17
BigQuery:.....	17
Other References:	17
Guidance to setup twitter developer account:.....	17
Tweepy API:	17
Google Map API:.....	17
Signed Contribution Agreement	18

Project Link

Link: <https://cloudcoursedelivery.ts.r.appspot.com/>

Summary

Cloud course delivery introduces a website that allows users to view information about the selected universities such as locations, latest reviews, and most recent tweets. This is to provide convenience for users when selecting a potential university that the user might be interested in as well as for members of the public.

Introduction

The idea behind this app was to create a platform that allows users to navigate through different universities to retrieve helpful information for the specified university as opposed to having to scour the internet.

In a real-world scenario, students from any level may use this app to help decide their decision to pick which university to apply after overlooking the vast variety of universities.

The main functions of our app provide multiple services that enable users to retrieve location of the selected university as well as food outlets in the surrounding area. In addition, users are also greeted with the most recent tweets from a mix of all three universities with an optional language translated version, and lastly, a user is welcome to post a review about any improvements for the website or any content that is relatable to the universities.

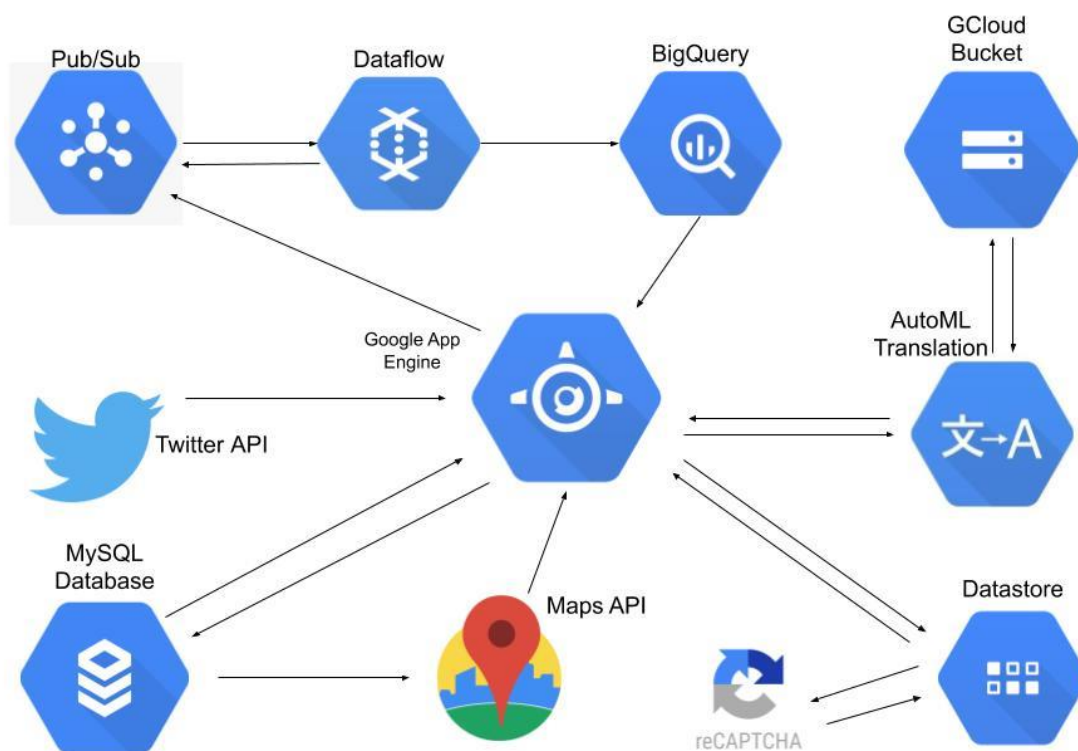
Adding on to the tweets. Recent university tweets are displayed providing users useful information such as upcoming open days, new courses offered as well as various updates.

Related work

Similar websites/applications are university search courses, where users can search for university courses. An example would be [The Good Universities Guide](#), where they have the courses of universities and how they are rated.

[Campus Review](#) is also a similar website in foundation, they have news posts created about Australian Universities. We wanted this to be a foundation of our website, where users can get relevant information about Australian universities and keep up to date with current university reviews.

Software Design/Architecture



Pub/Sub

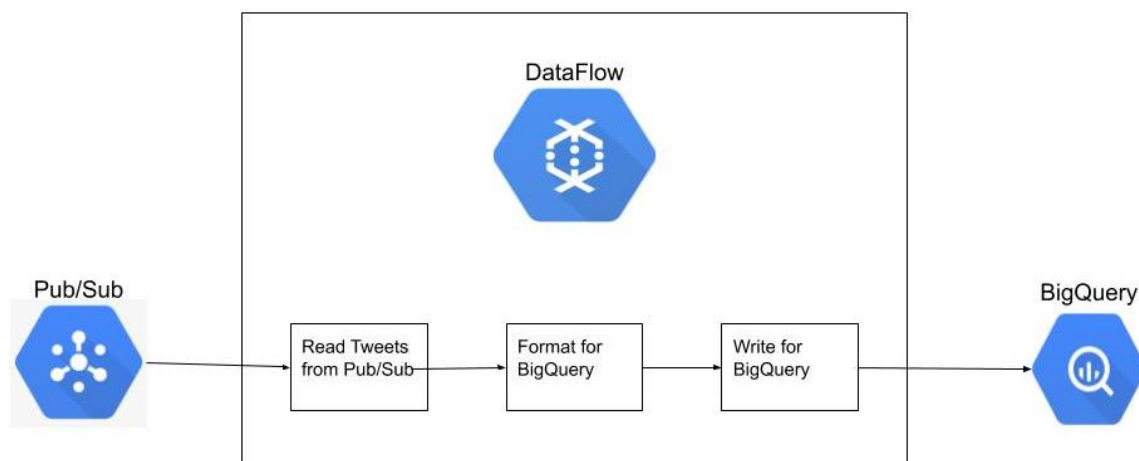
After getting tweets from the Twitter API, they are published as messages to Pub/Sub which is then pulled from Dataflow to query in BigQuery.

Dataflow

Dataflow was implemented as a core part of our cloud architecture, Dataflow connects Pub/Sub to BigQuery to display tweets on the web application. Once tweets are read in from the Twitter API, it publishes the message to Pub/Sub. Pub/Sub has a subscriber that will pull this message, Dataflow will convert the tweets from a json format into a BigQuery table through a pipeline. Dataflow is what connects Pub/Sub to BigQuery and plays a major role in our cloud system architecture.

BigQuery

As discussed before BigQuery is used to query the table of tweets that is retrieved from the twitter API through Pub/Sub and Dataflow. We also used BigQuery to query a dataset of food locations in Melbourne from a csv file, BigQuery grabs this data and shows it on the Maps API alongside the University locations.



Datastore

Interacts with GAE, GAE can post new reviews for Google Datastore to store, as well as showing the most recent reviews on the main page of the GAE. New posts are confirmed by the reCAPTCHA to stop bots from spamming the Datastore submission.

Twitter API

We used the Twitter API to get tweets relevant to universities that we wanted to display on our page, this is then published to Pub/Sub as a message for further use.

Maps API

Displays a Google Map on the website with markers of the whereabouts of the selected University, this is linked with MySQL which holds the locations of each university. As well as food locations around the universities, which are grabbed using BigQuery. This is integrated using a JavaScript API, to be displayed on html.

MySQL

Stores both latitude and longitude for each university, which is connected to the Maps API to display the location of a selected university.

AutoML Translation and Bucket

Exported training data (tsv and csv files) to google cloud bucket to provide easy accessibility when creating and building a machine learning model with autoML. Once the model is trained, we use it to predict and convert each word from a text file to the specified language. The text files contain the recent twitter posts.

reCAPTHCA

Google cloud provides a human authenticator for submission forms to help prevent websites from spam, it is a simple test that allows to tell the difference between human and bots. In our case, we wanted real reviews to be submitted.

Implementation

To implement this application the first step to take is to create a google cloud project, this project can then be set in a local directory on your machine, using the project ID. Once this has been done it makes implementation easier, because it automatically links some of the services when implementing them.

Flask

Flask was used for our google app engine backend, by using Flask routes the google app engine displays the website @ <https://cloudcoursedelivery.ts.r.appspot.com/>.

To install flask, you would need to have either python 2 or 3 installed, create a virtual environment and do a “**pip install flask**” which installs all the required start-up dependencies.

To launch development server, you would need to enter the following:

- **Set FLASK_APP=[your application name]**
- **Set FLASK_ENV=development**
- **Flask run**

By using the pip commands, you can install any dependencies, for more information on which package to install visit:

- <https://pypi.org/>

Twitter API

The twitter API uses the python package tweepy which is a twitter web scraper. Firstly, to gain authentication you are required to have a twitter developer account, from there you can retrieve the credentials to connect from your application to twitter. Store credentials within your application code and start retrieving JSON data from the tweets.

Use the following command inside your virtual environment to install tweepy:

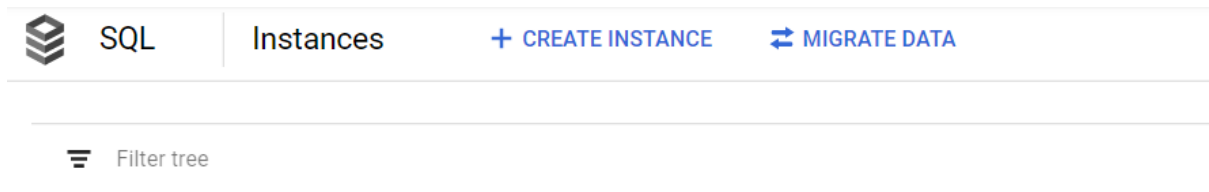
- **Pip install tweepy**

And visit the following website to apply for a twitter developer account:

- <https://developer.twitter.com/en>

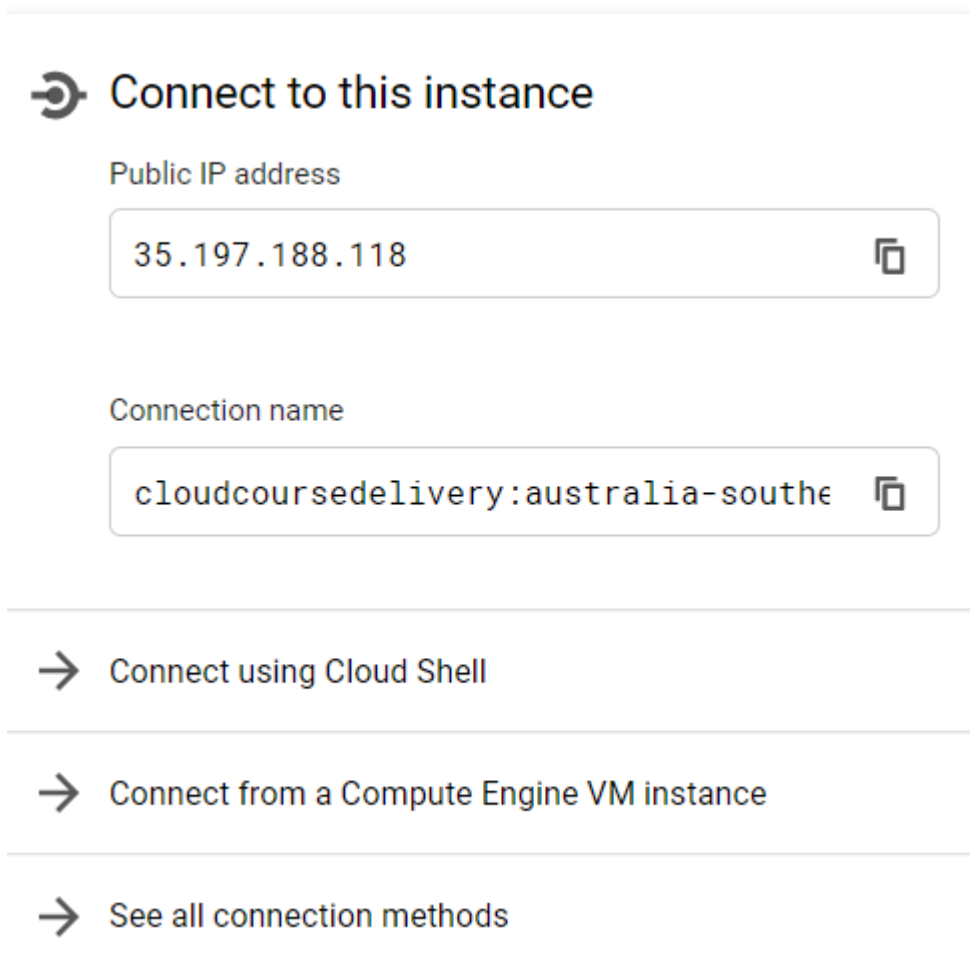
MySQL

To set up MySQL you first need to create an instance in the Cloud SQL instances page, Click Create Instance and enter the prompted fields.

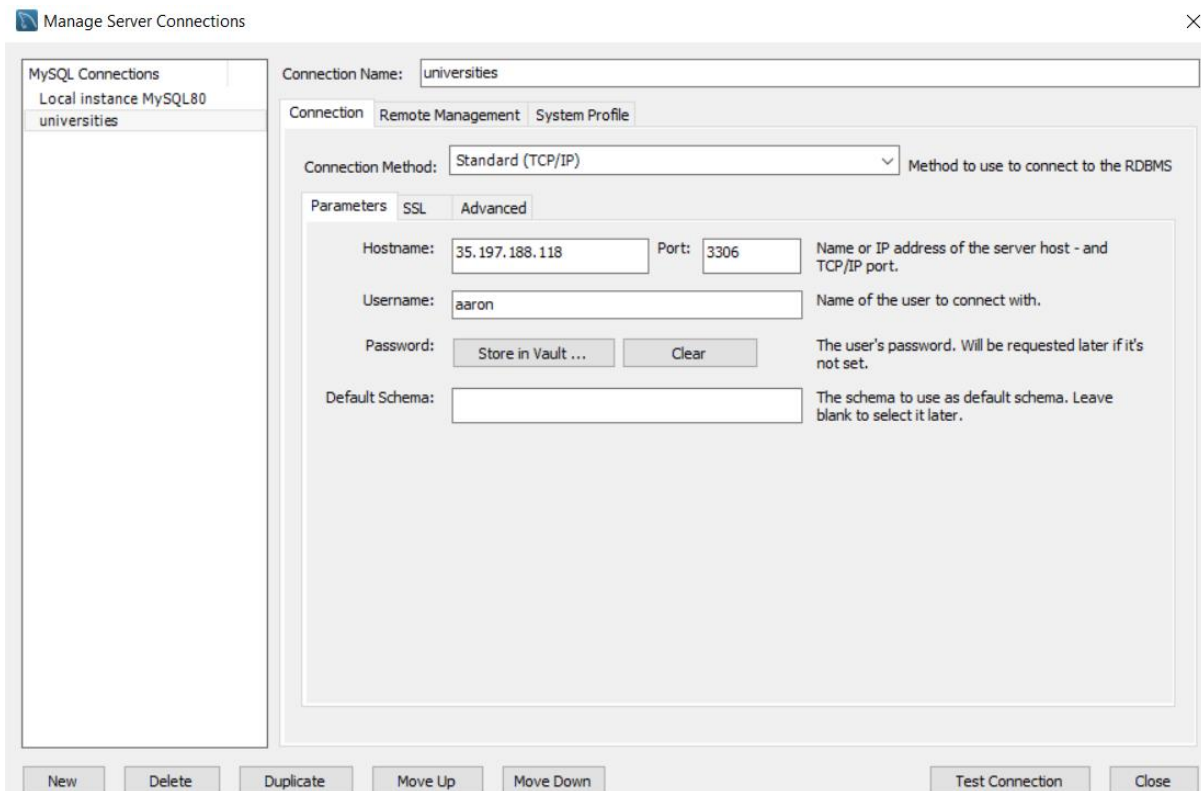


Once the SQL instance has initialized it can be connected to using the public IP address provided, a VPC can also be set up to allow the GAE to connect to it. However, if you want to connect it to a local server you need to install MySQL workbench and connect it via IP address.

Google Cloud IP address



MySQL Workbench



The application executes query commands to retrieve data from the table by importing the mysql.connector package.

Type the following in the virtual environment to install mysql.connector:

- **Pip install mysql-connector-python**

BigQuery

After enabling the BigQuery API on GCloud, a dataset needs to be uploaded to be queried by BigQuery. Install BigQuery packages by using:

- **Pip install google-cloud-bigquery**

To enable accessibility to google cloud service, a service account is required.

Create service account private key

Create service account

Service account name [?] Role [?]

Service account ID

☒ **Furnish a new private key**
Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

☒ **JSON**
Recommended

☐ **P12**
For backward compatibility with code using the P12 format

☐ **Enable G Suite Domain-wide Delegation**
Allows this service account to be authorized to access all users' data on a G Suite domain without manual authorization on their part. [Learn more](#)

After successfully obtaining the private key, add the path to your PATH ENVIRONMENT VARIABLES.

Once this is done you can import the BigQuery package which automatically links to your BigQuery service if you have set the project ID in your working directory. You can then query the dataset from the backend codebase and handle the results.

PUB/SUB

After enabling Pub/Sub API you would need to create a topic to handle incoming data to send to its subscribers.

Create new topic

Google Cloud Platform CloudCourseDelivery

Search products and resources

Pub/Sub

Topics [+ CREATE TOPIC](#) [DELETE](#)

Create a subscriber for that topic.

SUBSCRIPTIONS

SNAPSHOTS

A subscription captures the stream of messages published to a given topic. You can also stream messages to BigQuery or Cloud Storage by creating a subscription from a Cloud Dataflow job. [Learn more](#)

CREATE SUBSCRIPTION ▾

Filter table

Subscription ID ↑	Subscription name	Project
MySub	projects/cloudcoursedelivery/subscriptions/MySub	cloudcoursedelivery

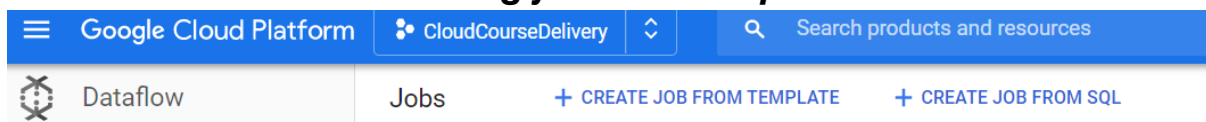
After setting Pub/Sub up, run the following command to install the packages:

- **Pip install google-cloud-pubsub**

DATAFLOW

To create a pipeline that orchestrates a conversion from json into a BigQuery table. We would need to first “Create a job from template”

Creating job from template



Fill in the following fields

← Create job from template

Job name *
temp
Must be unique among running jobs. Use lowercase letters, numbers, and hyphens (-).

Regional endpoint *
asia-east1
Choose a Dataflow regional endpoint to deploy worker instances and store job metadata. You can optionally deploy worker instances to any available Google Cloud region or zone by using the worker region or worker zone parameters. Job metadata is always stored in the Dataflow regional endpoint. [Learn more](#)

Dataflow template *
Pub/Sub Topic to BigQuery
Streaming pipeline. Ingests JSON-encoded messages from a Pub/Sub topic, transforms them using a JavaScript user-defined function (UDF), and writes them to a pre-existing BigQuery table as BigQuery elements.

Required parameters

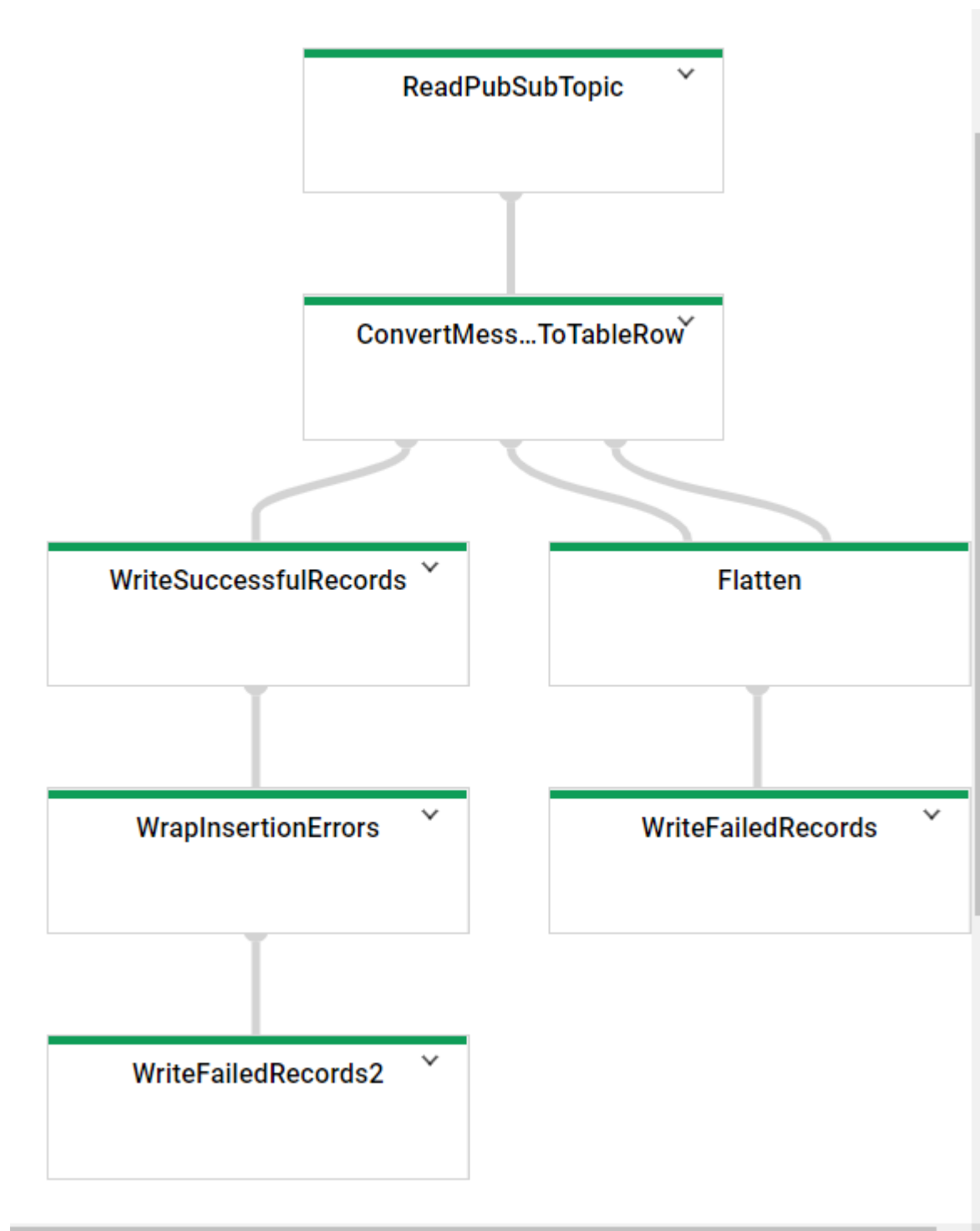
Input Pub/Sub topic *
The Pub/Sub topic to read the input from. Ex: projects/<project-id>/topics/<topic-name>

BigQuery output table *
The location of the BigQuery table to write the output to. If you reuse an existing table, it will be overwritten. The table's schema must match the input JSON objects. Ex: <your-project>:<your-dataset>:<your-table>

Temporary location *

Once the form is filled out, let the pipeline run its course and would take approximately 2-5 minutes.

DataFlow in action



AutoML Translation

Machine learning handles data to predict unseen data in a structural way, Google Cloud has provided a service which eases the construction of creating, building and evaluating machine learning models making it easier for us to predict and translate words.

Firstly, you would need to enable the API and then create a dataset from uploaded csv and tsv files from google cloud bucket.

Snippet of all the uploaded training data inside google cloud bucket

[← Bucket details](#) [EDIT BUCKET](#) [REFRESH BUCKET](#)

cloudcoursedelivery-vc

[Objects](#) [Overview](#) [Permissions](#) [Bucket Lock](#)

[Upload files](#) [Upload folder](#) [Create folder](#) [Manage holds](#) [Delete](#)

Buckets / cloudcoursedelivery-vc

<input type="checkbox"/>	Name	Size	Type	Storage class	Last modified	Public access	Encryption	Retention expiration date	Holds	
<input type="checkbox"/>	en-de.tsv	857 KB	application/octet-stream	Regional	5/21/20, 1:17:09 PM UTC+10	Not public	Google-managed key	–	None	⋮
<input type="checkbox"/>	en-es.csv	51 B	application/octet-stream	Regional	5/21/20, 1:17:11 PM UTC+10	Not public	Google-managed key	–	None	⋮
<input type="checkbox"/>	en-es.tsv	878.35 KB	application/octet-stream	Regional	5/21/20, 1:17:12 PM UTC+10	Not public	Google-managed key	–	None	⋮
<input type="checkbox"/>	en-fr.tsv	907.62 KB	application/octet-stream	Regional	5/21/20, 1:17:13 PM UTC+10	Not public	Google-managed key	–	None	⋮
<input type="checkbox"/>	en-ru.tsv	1.13 MB	application/octet-stream	Regional	5/21/20, 1:17:09 PM UTC+10	Not public	Google-managed key	–	None	⋮
<input type="checkbox"/>	en-zh.tsv	772.29 KB	application/octet-stream	Regional	5/21/20, 1:17:08 PM UTC+10	Not public	Google-managed key	–	None	⋮

Create dataset

Google Cloud Platform CloudCourseDelivery

Translation

Dashboard

Datasets

Models

Datasets

Filter datasets

	Name	Source	Target	Total pairs	Last updated	Status	
✓	en_es_dataset1 TRL1243680691921616896	English (EN)	Spanish (ES)	8,720	5/21/20, 2:13 PM	Success: Training model	⋮

Once the dataset is uploaded, it is self explanatory from there, as it has tabs to guide you through

IMPORT	SENTENCES	TRAIN	EVALUATE	PREDICT
--------	-----------	-------	----------	---------

NOTE

- Training a dataset would take time depending on the amount of inputted data
- Large datasets may take up to an hour to train.

Once the model is trained, we install the autoML, and google translate package:

- **Pip install google-cloud-automl**
- **Pip install google-cloud-translate**

DATASTORE

Because the project ID has previously been set when the directory was being set up, Datastore will automatically connect to the Datastore database. To implement GCloud Datastore you need to enable the Google Datastore API, once it is accessible you can use it to store data. (Note: that once you upload data you can no longer use it in Firestore mode). No extra implementation is required other than declaring the entity in the code, which is used to update, edit, and retrieve entities from Datastore. No extra implementation is needed on the GCloud Client GUI side.

User manual

User will be greeted with recent twitter posts from all universities on the left side of the web and is able to scroll to tweets that are posted later than the current date.

Most Recent Twitter Posts

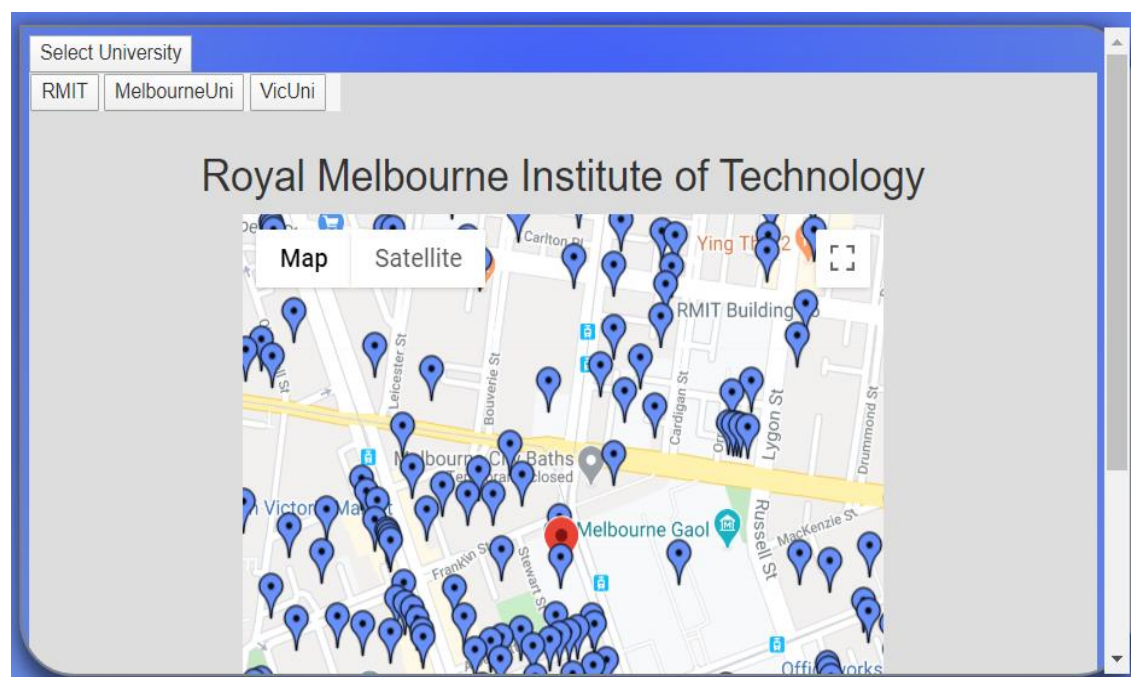
```
-----  
datetime.datetime(202  
5, 20, 6, 44),  
'unimelb',  
'Exciting news!  
@SciMelb &  
@qldmuseum  
researchers have  
revealed the  
discovery of new  
#megafauna that  
lived in  
northe???'  
https://t.co/i74lBWR5
```

```
-----  
datetime.datetime(202  
5, 20, 5, 42, 58),  
'unimelb', 'RT
```

Also, given an optional button to translate the tweet posts into another language, in this example we use Spanish.

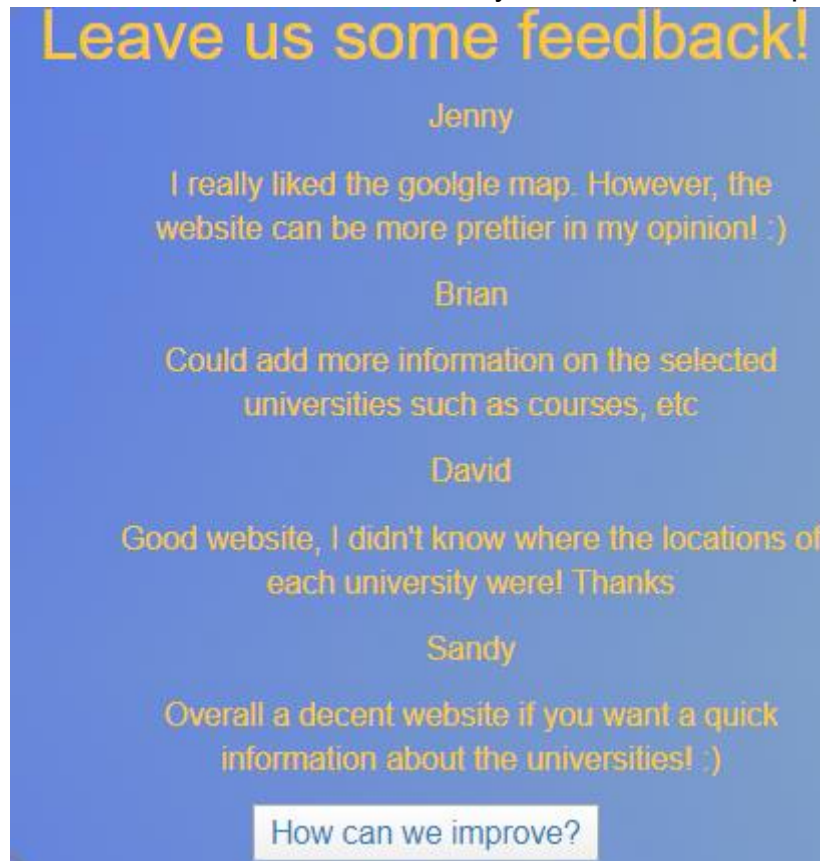


When you click on select universities, it drops down a list of universities which can be clicked on to show the location of the university and the surrounding food outlets. If you want to select a new



university map, deselect your current university and click on the new university that you want to display. The red marker is the location of the university that you selected, and the blue markers are the different food outlets around the university.

As you can see there are University reviews posts, to create a new review post click the “Create New Post” button and it will redirect you to the “/reviews” page.



It will redirect you to a page with this form, enter the name and review comment, then click the Submit button. This will then redirect you to the main page, where it will display your new review post as the top. (These posts are ordered by time created, most recent showing first).

A screenshot of a feedback form titled "Enter Name:" and "Enter Review:". It includes a text input field for the name, a text input field for the review, a checkbox for "I'm not a robot", a reCAPTCHA logo, and a "Submit" button.


Enter Name:

Enter name here...

Enter Review:

Enter review here...

☐ I'm not a robot

 reCAPTCHA
Privacy - Terms

References

Google Cloud Services:

Pub/sub:

- <https://cloud.google.com/appengine/docs/flexible/python/writing-and-responding-to-pub-sub-messages>

DataFlow:

- <https://cloud.google.com/dataflow/docs/quickstarts/quickstart-python>

autoML:

- <https://cloud.google.com/translate/automl/docs/before-you-begin>

BigQuery:

- <https://cloud.google.com/bigquery/docs/quickstarts/quickstart-client-libraries>

Other References:

Guidance to setup twitter developer account:

- <https://developer.twitter.com/en/docs/basics/getting-started>

Tweepy API:

- <http://docs.tweepy.org/en/v3.5.0/api.html>

Google Map API:

- <https://developers.google.com/maps/documentation/javascript/tutorial>

Signed Contribution Agreement

Student Name: Jack Ryan	Student Name: Aaron Chan
Student ID: s3655960	Student ID: s3666603
Contributions: Flask Google App Engine Pub/Sub Datastore BigQuery Twitter API reCAPTCHA	Contributions: Flask Google App Engine Pub/Sub Dataflow BigQuery Maps API Twitter API MySQL reCAPTCHA AutoML Translation
Contribution Percentage: 50%	Contribution Percentage: 50%
By signing below, I certify all information is true and correct to the best of my knowledge Signature: Jack Ryan Date: 22/05/2020	By signing below, I certify all information is true and correct to the best of my knowledge Signature: Aaron Chan Date: 22/05/2020