DataEng S24: Project Assignment 3

Data Integration

Due date: May 26, 2024 at 10pm

Congratulations! By now you have a working, end-to-end data pipeline. Unfortunately, it does not have enough data to properly implement our Data Scientist's visualization. To fill out information such as "route ID" you need to access another source of data and build a new pipeline to integrate it with your initial pipeline. Here are your steps:

A. access the stop event data
B. build a new pipeline for the stop event data
C. integrate the stop event data with the breadcrumb data
D. testing

A. Stop Events Data

Access TriMet "Stop Events" data at this URL:

```
https://busdata.cs.pdx.edu/api/getStopEvents?vehicle_num=<vehicle_num>
```

As with the previous data source, this data set gives all TriMet vehicle stop events for a single day of operation. Again, make sure to replace the vehicle_num with the vehicle id's assigned to you.

B. New Pipeline

Your job is to build a new pipeline that operates just like the previous one, including use of Cloud Pub/Sub, automation, validation and loading.

C. Integrate Stop Events with Bread Crumbs

The two pipelines (BreadCrumb pipeline and StopEvent pipeline) must update the values in the Trip table such that all of the columns of both tables are filled correctly.

Alternatively, it would be OK to load the StopEvent data into a separate table and then use SQL views to integrate the two datasets.

D. Visualization

MapboxGL is a data visualization tool that allows you to view your breadcrumb data and display it on a map. Your job is to integrate this tool with your database tables so that you can query the breadcrumb and trip data in your database server, transform to geoJSON format and display the resulting map visualization. To get started, see this guide.

Alternatively, you may use an alternative visualization tool (such as folium) to create the required visualizations. We do not provide any guides for doing it, but you are free to do so if you prefer. The submitted visualizations must be equivalent or superior to the visualizations produced by the provided MapboxGL based visualization tool.

Submission

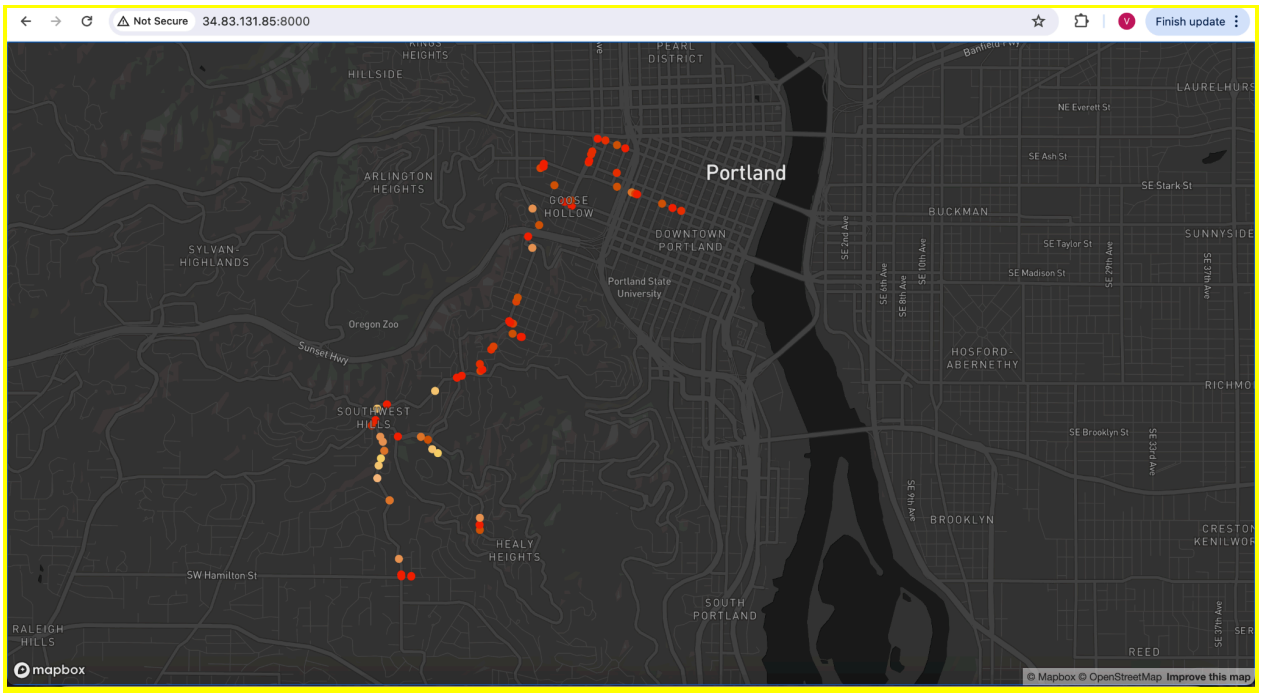
Make a copy of this document and update it to include the following visualizations. For each visualization extract from your database a list of (latitude, longitude, speed) tuples and then use the provided visualization code (see Section D above) to display bus speeds at all of the corresponding geographic coordinates. So, for example, if you are asked to visualize a “trip”, then you must query your database to find all of the (latitude, longitude, speed) tuples for that trip, and then display a map showing the recorded/calculated bus speed at each (latitude,longitude) location.

No need to produce software that neatly displays trips, routes, dates, times, etc. onto the visualization itself. Instead, just paste a screen capture of the map-based speed visualization into your submission document and then include a text description of the contents of the visualization. For example, text like this: “Bus Speeds for all outbound trips of route 72 between 9am and 11am on Wednesday, February 15, 2023.”

Visualization 1. A visualization of speeds for a single trip for any bus route that crosses the US-26 tunnel. You choose the day, time and route for your selected trip. To find a trip that traverses this tunnel, consider finding a trip that includes breadcrumb sensor points within this bounding box: [(45.506022, -122.711662), (45.516636, -122.700316)]. Any bus trip that includes breadcrumb points within that box either drove across the tunnel or teleported across!

SQL: SELECT longitude, latitude, speed from breadcrumb WHERE trip_id = (SELECT b.trip_id from breadcrumb b INNER JOIN trip t on t.trip_id=b.trip_id where b.latitude BETWEEN 45.506022 AND 45.516636 AND b.longitude BETWEEN -122.711662 AND -122.700316 AND t.route_id > 0 ORDER BY DATE(b.tstamp) desc limit 1);

Bus speeds for the outbound trip 247253202 of route 51 were recorded within the provided breadcrumb points. This trip occurred between 8 AM and 9 AM on Thursday, January 26, 2023.

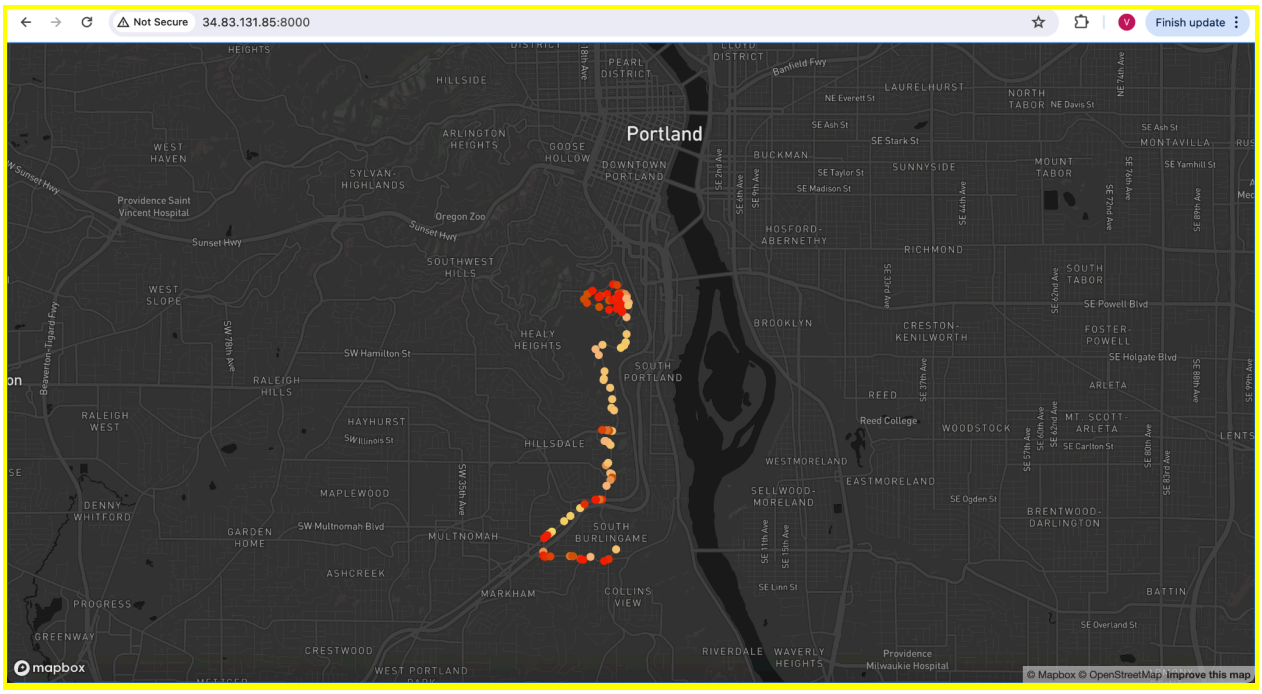


Visualization 2. All outbound trips that occurred on route 65 on any Friday (you choose which Friday) between the hours of 4pm and 6pm.

As I didn't have any proper data from Friday, as Stop events were not loaded, I chose Monday and also the trips were inbound for monday.

SQL: select longitude, latitude, speed from breadcrumb where EXTRACT(HOUR FROM tstamp) BETWEEN 16 AND 18 and EXTRACT(DOW FROM tstamp) = 1 and trip_id IN (Select DISTINCT trip_id from trip where route_id=65 and direction="Back");

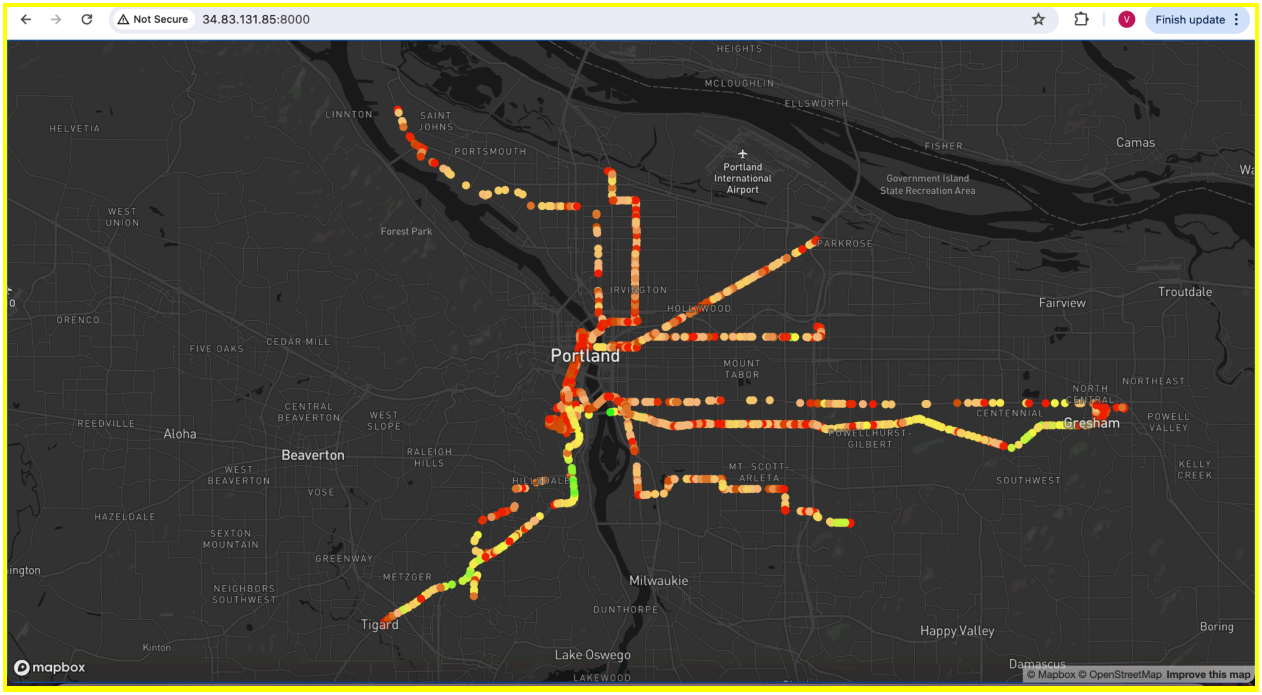
Bus speeds for all inbound trips, specifically trips 243320956 and 243727852 with route_id 65, which occurred on Monday, January 23, 2023, and ran between 4 PM and 6 PM.



Visualization 3. All trips that travel to and from PSU campus on any Sunday morning (you choose which Sunday) between 9am and 11am.

SQL: SELECT longitude, latitude, speed FROM breadcrumb WHERE trip_id IN (SELECT DISTINCT b.trip_id FROM breadcrumb b INNER JOIN trip t ON t.trip_id = b.trip_id WHERE EXTRACT(DOW FROM b.tstamp) = 0 AND b.latitude BETWEEN 45.5090203 AND 45.5097928 AND b.longitude BETWEEN -122.6856295 AND -122.6816254 AND EXTRACT(HOUR FROM b.tstamp) BETWEEN 9 AND 11 AND t.route_id > 0 AND DATE(b.tstamp) = '2023-01-22');

Bus speeds of all inbound and outbound trips that occurred on Sunday 22 January 2023.



Visualization 4. The longest (as measured by time) trip in your entire data set. Indicate the date, route #, and the trip ID of the trip along with a visualization showing the entire trip.

Trip_id : 243309359 is the longest but this does not have recorded route_id. So I chose the second longest trip.

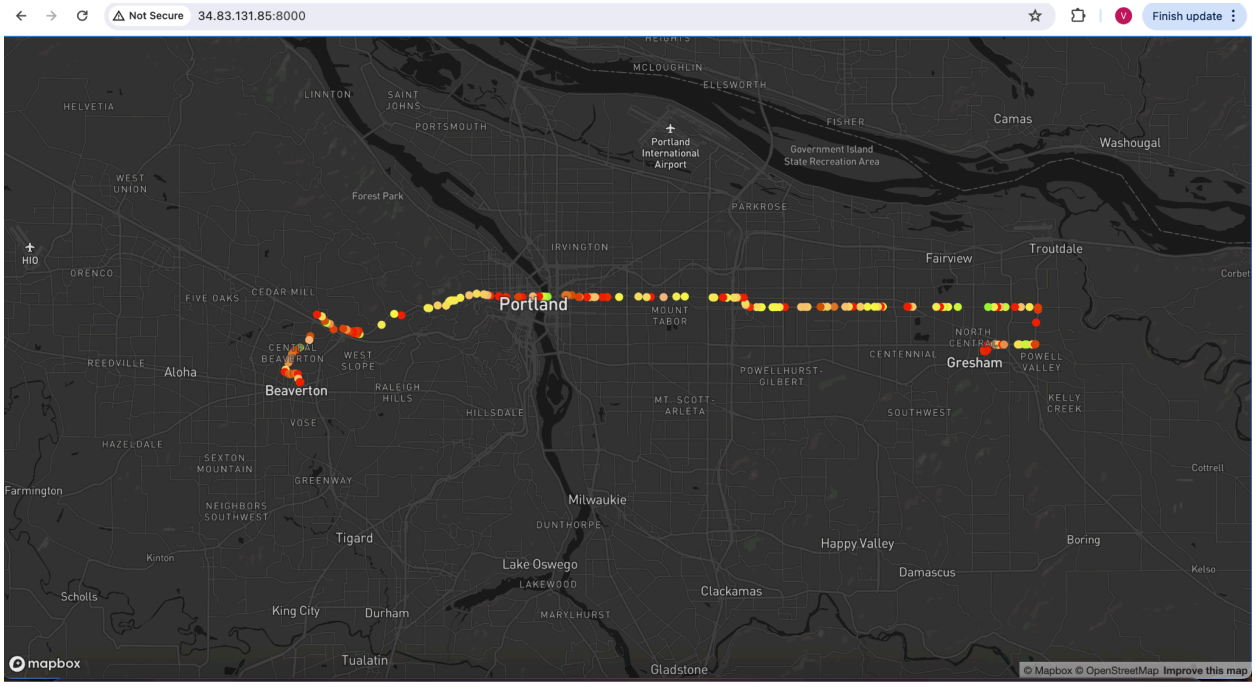
First longest trip - 12425.00 seconds
Date - 2023-01-23
Route ID - Not Recorded
Trip ID -243309359

Second longest trip - 9388.00 seconds
Date - 2023-01-26
Route ID - 20
Trip ID - 245571696

SQL:
```
SELECT  longitude, latitude, speed
FROM breadcrumb
WHERE trip_id IN (
   SELECT b.trip_id
   FROM breadcrumb b
   INNER JOIN trip t ON t.trip_id = b.trip_id
   WHERE t.route_id > 0
   GROUP BY b.trip_id
```

```
    ORDER BY EXTRACT(EPOCH FROM (MAX(b.tstamp) - MIN(b.tstamp))) DESC
    LIMIT 1
);
```

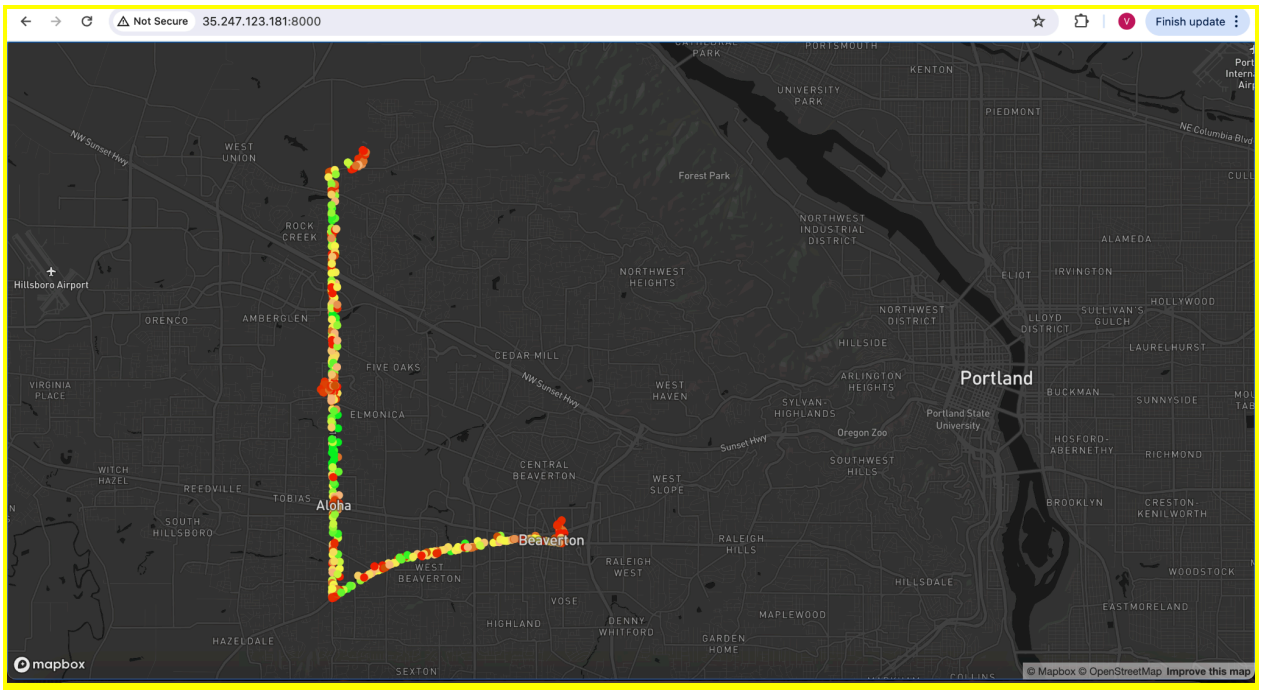
Bus speed of longest trip 245571696(Second largest) occurred on Thursday 26 January 2023



Visualization 5a, 5b, 5c, …. Three or more additional visualizations of your choice. Indicate why you chose each particular visualization.

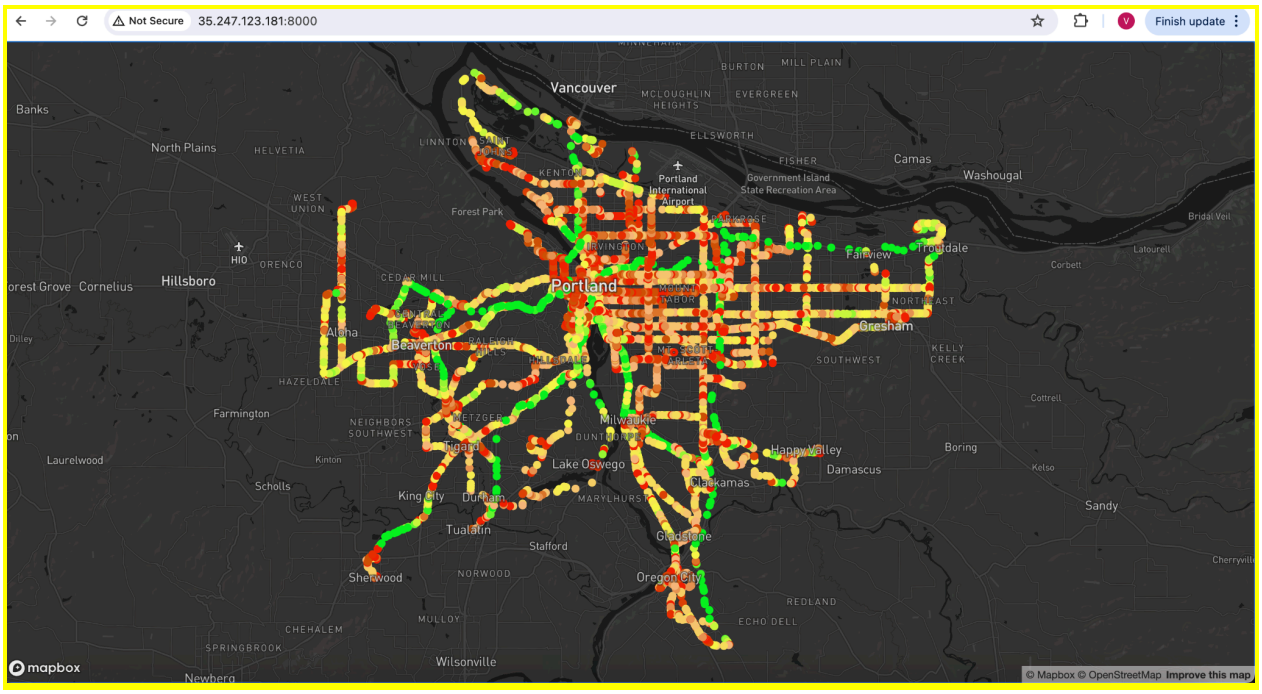
Visualization 5a. All the trips that ran on specific route for date 23 January 2023. I chose route 52 (52-Farmington/185th runs between Beaverton Transit Center, Aloha, Willow Creek Transit Center, Tanasbourne and PCC Rock Creek, along Farmington, 185th and Springville.)

SQL: SELECT b.longitude,b.latitude, b.speed FROM Breadcrumb b JOIN Trip T ON b.trip_id = t.trip_id WHERE t.route_id =52 AND DATE(b.tstamp) = ‘2023-01-23’



Visualization 5b: Find speed of Buses on each route during peak hours during all the fridays.This query can help analyze the efficiency of bus routes during peak traffic hours. I have taken peak hours as 6pm - 8pm for fridays.

SQL:SELECT b.longitude,b.latitude, b.speed FROM Breadcrumb b JOIN Trip t ON b.trip_id = t.trip_id WHERE EXTRACT(HOUR FROM b.tstamp) BETWEEN 18 AND 20 AND DATE(b.tstamp) = ‘2023-01-2’ AND EXTRACT(ISODOW FROM b.tstamp) = 5 ;



Your Code

Provide a reference to the repository where you store your python code. If you are keeping it private then share it with the Professor (rbi@pdx.edu or mina8@pdx.edu) and TA (vysali@pdx.edu).