Design Specification and Project Plan

Components

Component 1:

Name: Date Entry

What is does: Date Entry component takes a specific date between April and September as input based on user's selection. There are two drop down menus for users to make selections. One for month selection and one for day selection. This can be done by bokeh package (bokeh.plotting). Then the input will be passed into a SQL statement and the database is filtered by the input date.

Input: Month and day selections in the drop down menus

Output: Records returned from Azure SQL database filtered by Date Entry

Component 2:

Name: Hour Entry

What it does: Hour Entry component takes a specific hour which between 0 and 23 as input based on user's selection. There is a slider with range from 0 to 23 for users to select hour. This can be done by bokeh package (bokeh.plotting). Then the input will be passed into a SQL statement and the database is filtered by the input date.

Input: Hour selection from a slider

Output: Records returned from Azure SQL database filtered by Hour Entry

Component 3:

Name: Uber/Taxi Selection

What it does: The Uber/Taxi Selection component takes an user input of Uber, Taxi, or Both. This selection would be used to determine the presentation of Uber data, taxi data, or both Uber data and taxi data on the map.

Input: Uber/Taxi Selection

Output: Categorical data (0,1,2). 0 represents Uber data; 1 represents taxi data; 2 represents both Uber and taxi data

Component 4:

Name: Address Entry

What it does: The Address Entry component takes an user input address. We will retrieve the coordinates of the entered address by using a python package called Geocoder associated with Google Map API.

Geocoder documentation:

http://geocoder.readthedocs.io/index.html

Then, we will input all the latitudes and longitudes with deviations of -0.1 and +0.1 ofp into a SQL statement.

Input: Address Entry

Output: Records returned from Azure SQL database filtered by Address Entry

Component 5:

Name: Azure SQL Database

What it does: Azure SQL database will store both Uber and yellow cab data sets. We will use use a python package called pyodbc to connect to the database and query the data. When we query the data, we first take the date, hour, and address as inputs and prepare the SQL statements. Then the database will execute the query and return a collection of desired records. Based on this collection, we will load the coordinates into our map interface. Each coordinate will be displayed as a dot on the map. We could also do some counts and aggregations through queries and use the results to generate a heatmap and some trend plots or scatter plots.

https://docs.microsoft.com/en-us/azure/sql-database/sql-database-connect-query-python

Input: Date, Hour, Address, Uber/Taxi Indicator

Output: Records returned from Azure SQL database filtered by given entry

Component 6:

Name: Map of Uber and yellow cab pickup locations (Using Bokeh)

What it does: The Map component is produced by using a python package called Bokeh. It takes a collection of coordinates and display them as dots on the map. The map will be interactive, so users can play around with it It also can associate with date and hour filters to provide users more controls of tWe will use the python Bokeh package to implement this component. Moreover, we will import GMapPlot, GMapOptions, ColumnDataSource, Circle, DataRange1d, PanTool, WheelZoomTool, BoxSelectTool from bokeh.model.

Input: Date, Hour, Uber/Taxi, Latitude and Longitude, number_of_traffic

Output: Display an area of all Uber and Taxi pickups coordinates surrounding that latitude/longitude pair on the map.

Interactions

Use Case 1: Comparison of Uber and Taxi

System will receive inputs from component of Uber/Taxi Selection. System will talk to Azure SQL Database using python package pyodbc. Outputs returned from the database are data including variables of Date, Hour, Uber/Taxi, Latitude and Longitude, and number_of_traffic. System will talk to the Map component to represent the data results.

In this use case, user can select the presentation of Uber data, taxi data, or both data. The use case enables user to compare the traffic of Uber and taxi. Once the user inputs of Uber/Taxi Selection. change, system will go through the process above the presents the new results.

Use Case 2: Analysis of traffic and location data

System will receive inputs from component of Address Entry. System will talk to Azure SQL Database using python package pyodbc. Outputs returned from the database are data including variables of Date, Hour, Uber/Taxi, Latitude and Longitude, and number_of_traffic. System will talk to the Map component to represent the data results.

In this use case, user can specific address or area of results. The use case enables user to compare the traffic of different boroughs or the traffic near landmark. Zoom in/zoom out feature is also enabled in this use case. Scope of visualization variable is set to default at the beginning. Once the user click zoom in or zoom out button, the map component will update the visualization accordingly.

Use Case 3: Analysis of time series data - the rise of Uber

System will receive inputs from components of Date Entry and Hour Entry. Data System will talk to Azure SQL Database using python package pyodbc. Outputs returned from the database are data including variables of Date, Hour, Uber/Taxi, Latitude and Longitude, and number_of_traffic. System will talk to the Map component to represent the data results.

In this use case, user can change Date Entry and Hour Entry. The use case enables user to compare the traffic of different hours in a day; the use case can also represent the traffic change over dates through the year. Once the user inputs of Date Entry and Hour Entry changed, system will go through the process above the presents the new results.

Project Plan

Plan For Week 8:

- Finalize data cleaning and selecting for the Uber pickup dataset and Yellow Taxi Trip dataset. (from April 2014 to September 2014)
- Develop our Database for these two datasets. Such as create tables, insert data into the SQL Server.
- Test our Database by trying some simple queries.
- Create the user interface and overall layout for the system.

Plan For Week 9:

- Implement all the components in Python as functions.
- Test each function individually make sure all the inputs and outputs are we expected (Use Unittest).
- Develop the interactions between components.

Plan For Week 10:

- Test the interactions between components. Make sure all visualizations are properly display.
- Polish our project, such as comments and coding style. (Run the pylint)

Plan For Week 11:

- Finish the project poster.
- Prepare some demos for the project.
- Present and demo our project on 6th June.