Vehicle Population Search Engine

5400 Team project proposal-Group 8

Yiqing Lin (yl5238)
Ting Yi Ho (th3019)
Tiannuo Yu (ty2474)
Shan Jiang (sj3231)
Yuxuan Zhu (yz4353)

Background and the Business Use Case

- Background
  - With the development of cars, more and more people prefer electric vehicles because of consciousness of environment protection
  - With the rise of Tesla and other brands, we wonder the current trends of choosing cars
- Business Problem
  - Car seller and car dealer have lack information about which vehicle to promote to the consumers in different regions
  - Consumers have lack information about the popularity of each vehicles when making purchase decision
- Project Purpose
  - Develop an engine to search for the most popular car brands and prototypes for each city in Washington state

Data source specification and procurement details

- We are looking to obtain data from DATA.GOV, which is an official website of the United States government, and we are going to use the electric vehicle population dataset. DATA.GOV (https://catalog.data.gov/dataset/electric-vehicle-population-data)

- This dataset shows the Battery Electric Vehicles (BEVs) and Plug-in Hybrid Electric Vehicles (PHEVs) that are currently registered through Washington State Department of Licensing (DOL)

- In this FREE dataset, we can obtain key information about the location(country, City, state) of vehicles , vehicle brand and electric vehicles types.

Data Governance Policy

- **Data source:** Since the dataset is provided by the Washington State Department of Licensing (DOL) and is publicly available, there will not be extra fee and data privacy concern.
- **Data quality:** We will perform data validation and cleaning after retrieving the data to ensure the accuracy and reliability of the data. Also , we will regularly update the database to ensure the data is current and accurate.

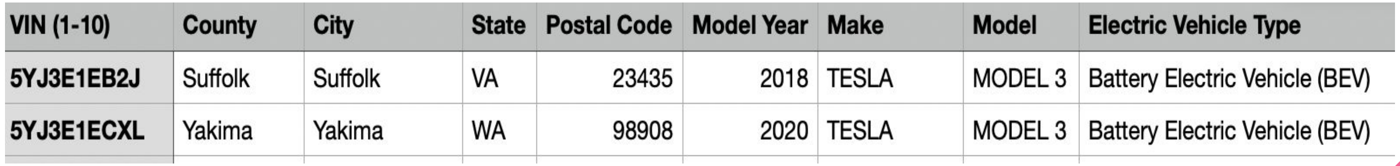
Projected Cost

- **Data source:** As the data source is provide by DOL, there not ba a cost for data source. Unless we want to expand to other paid data sources.
- **API service:** The developer version of the API services is free. However, if we were to scale it up, our monthly licensing cost would be $100 and can goes up to $300 or higher if we needed.
- **Cloud environment:** For a moderate database storage and operation, the cost will typically be about $150 per month

Proposed Design Choices

- Database Technology Choice
  - Dataset is relational which is one table with keys, so we decided to build relational database
  - We build SQL database in PostgreSQL using PGadmin
  - SQL Database is appropriate for the fundamental of retrieving system
- Dataset Sample - csv files
  - primary key : VIN number
  - Other used variables : County, City, State, Postal Code, Model Year, Make, Model, Vehicle Type

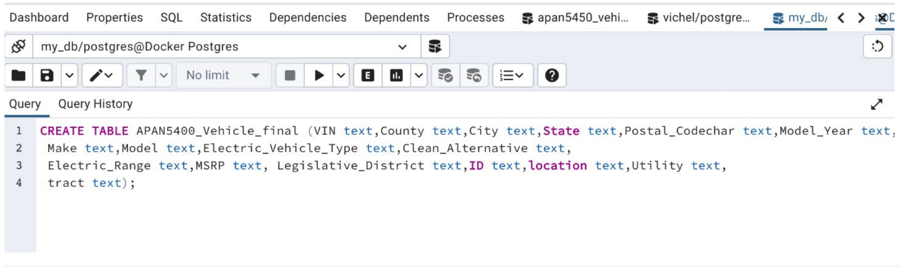
Data example :



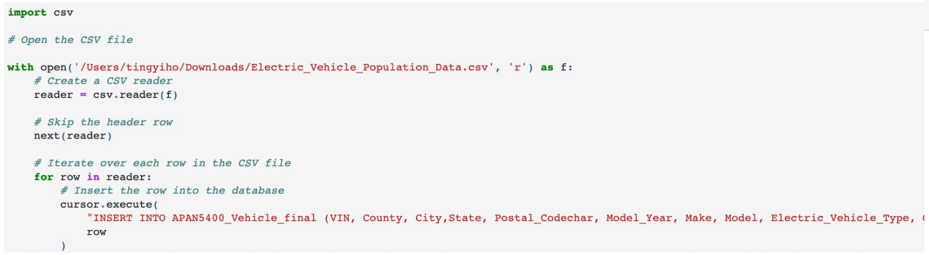
VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type
5YJ3E1EB2J	Suffolk	Suffolk	VA	23435	2018	TESLA	MODEL 3	Battery Electric Vehicle (BEV)
5YJ3E1ECXL	Yakima	Yakima	WA	98908	2020	TESLA	MODEL 3	Battery Electric Vehicle (BEV)

Selected Technologies- Design of the System

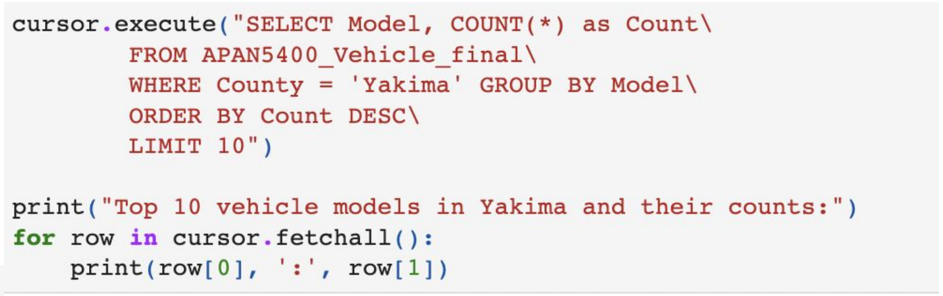
- Data retrieval system : SQL
    - Read CSV file and import it to PostgreSQL
    - Create table and build the database
    - Connect database to Jupyter Notebook
    - Using commands to retrieve data, ex SELECT, WHERE, GROUP BY



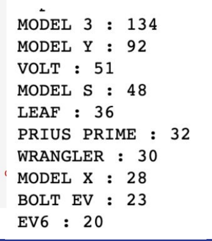
```sql
CREATE TABLE APAN5400_Vehicle_final (VIN text,County text,City text,State text,Postal_Codechar text,Model_Year text,
 Make text,Model text,Electric_Vehicle_Type text,Clean_Alternative text,
 Electric_Range text,MSRP text, Legislative_District text,ID text,location text,Utility text,
 tract text);
```



```python
import csv

# Open the CSV file

with open('/Users/tingyiho/Downloads/Electric_Vehicle_Population_Data.csv', 'r') as f:
    # Create a CSV reader
    reader = csv.reader(f)

    # Skip the header row
    next(reader)

    # Iterate over each row in the CSV file
    for row in reader:
        # Insert the row into the database
        cursor.execute(
            "INSERT INTO APAN5400_Vehicle_final (VIN, County, City,State, Postal_Codechar, Model_Year, Make, Model, Electric_Vehicle_Type,
            row
        )
```



```python
cursor.execute("SELECT Model, COUNT(*) as Count\
        FROM APAN5400_Vehicle_final\
        WHERE County = 'Yakima' GROUP BY Model\
        ORDER BY Count DESC\
        LIMIT 10")

print("Top 10 vehicle models in Yakima and their counts:")
for row in cursor.fetchall():
    print(row[0], ':', row[1])
```



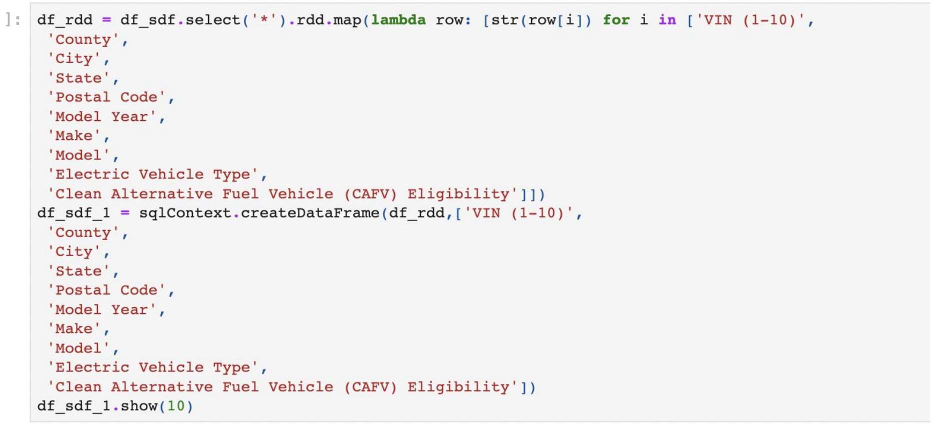
```
MODEL 3 : 134
MODEL Y : 92
VOLT : 51
MODEL S : 48
LEAF : 36
PRIUS PRIME : 32
WRANGLER : 30
MODEL X : 28
BOLT EV : 23
EV6 : 20
```

Selected Technologies- Design of the System

- Advanced data retrieval system : Pyspark
  - Read retrieved written CSV file to Pyspark dataframe
  - Select columns into RDD
  - Format them into str, and back to dataframe
  - Retrieve data with certain condition and input query

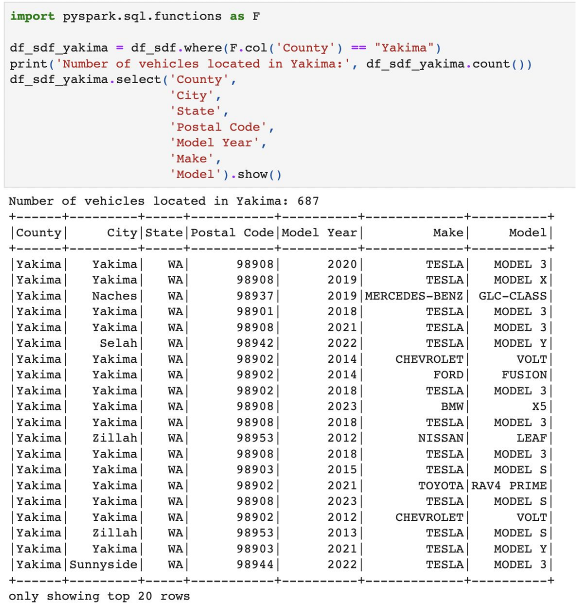
(4)Construct RDD

- Select columns into RDD, format them into str, and back to dataframe



```
]: df_rdd = df_sdf.select('*').rdd.map(lambda row: [str(row[i]) for i in ['VIN (1-10)',
    'County',
    'City',
    'State',
    'Postal Code',
    'Model Year',
    'Make',
    'Model',
    'Electric Vehicle Type',
    'Clean Alternative Fuel Vehicle (CAFV) Eligibility']])
   df_sdf_1 = sqlContext.createDataFrame(df_rdd,['VIN (1-10)',
    'County',
    'City',
    'State',
    'Postal Code',
    'Model Year',
    'Make',
    'Model',
    'Electric Vehicle Type',
    'Clean Alternative Fuel Vehicle (CAFV) Eligibility'])
   df_sdf_1.show(10)
```

(5)trail 1 test for selecting Yakima County



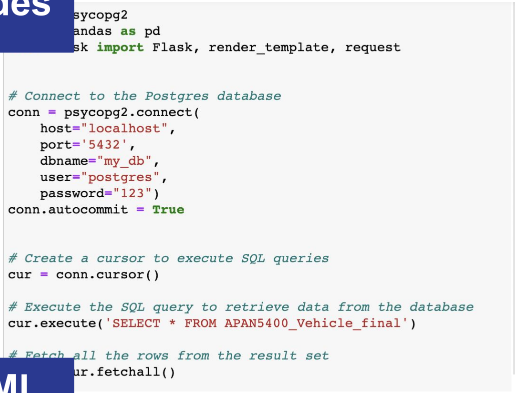
```
import pyspark.sql.functions as F

df_sdf_yakima = df_sdf.where(F.col('County') == "Yakima")
print('Number of vehicles located in Yakima:', df_sdf_yakima.count())
df_sdf_yakima.select('County',
                     'City',
                     'State',
                     'Postal Code',
                     'Model Year',
                     'Make',
                     'Model').show()
```

```
Number of vehicles located in Yakima: 687
+------+---------+-----+-----------+----------+-------------+----------+
|County|     City|State|Postal Code|Model Year|         Make|     Model|
+------+---------+-----+-----------+----------+-------------+----------+
|Yakima|   Yakima|   WA|      98908|      2020|        TESLA|   MODEL 3|
|Yakima|   Yakima|   WA|      98908|      2019|        TESLA|   MODEL X|
|Yakima|   Naches|   WA|      98937|      2019|MERCEDES-BENZ| GLC-CLASS|
|Yakima|   Yakima|   WA|      98901|      2018|        TESLA|   MODEL 3|
|Yakima|   Yakima|   WA|      98908|      2021|        TESLA|   MODEL 3|
|Yakima|    Selah|   WA|      98942|      2022|        TESLA|   MODEL Y|
|Yakima|   Yakima|   WA|      98902|      2014|    CHEVROLET|      VOLT|
|Yakima|   Yakima|   WA|      98902|      2014|         FORD|    FUSION|
|Yakima|   Yakima|   WA|      98902|      2018|        TESLA|   MODEL 3|
|Yakima|   Yakima|   WA|      98908|      2023|          BMW|        X5|
|Yakima|   Yakima|   WA|      98908|      2018|        TESLA|   MODEL 3|
|Yakima|   Zillah|   WA|      98953|      2012|       NISSAN|      LEAF|
|Yakima|   Yakima|   WA|      98908|      2018|        TESLA|   MODEL 3|
|Yakima|   Yakima|   WA|      98903|      2015|        TESLA|   MODEL S|
|Yakima|   Yakima|   WA|      98902|      2021|       TOYOTA|RAV4 PRIME|
|Yakima|   Yakima|   WA|      98908|      2023|        TESLA|   MODEL S|
|Yakima|   Yakima|   WA|      98902|      2012|    CHEVROLET|      VOLT|
|Yakima|   Zillah|   WA|      98953|      2013|        TESLA|   MODEL S|
|Yakima|   Yakima|   WA|      98903|      2021|        TESLA|   MODEL Y|
|Yakima|Sunnyside|   WA|      98944|      2022|        TESLA|   MODEL 3|
+------+---------+-----+-----------+----------+-------------+----------+
only showing top 20 rows
```

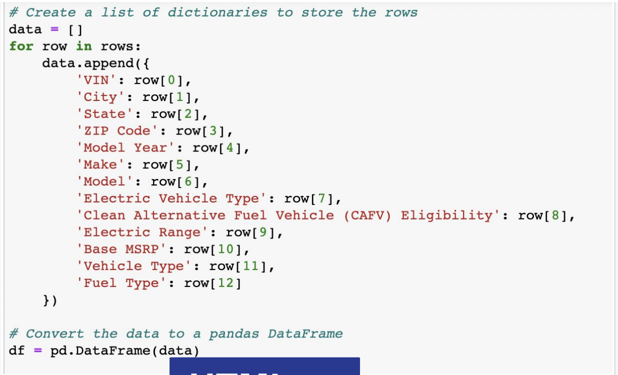
Selected Technologies- Design of the System

- Website Demo : Flask
  - Building search page and result page using Flask and connecting to the Postgres Database
  - There is one flask page can search by city, and the another page can show the result.

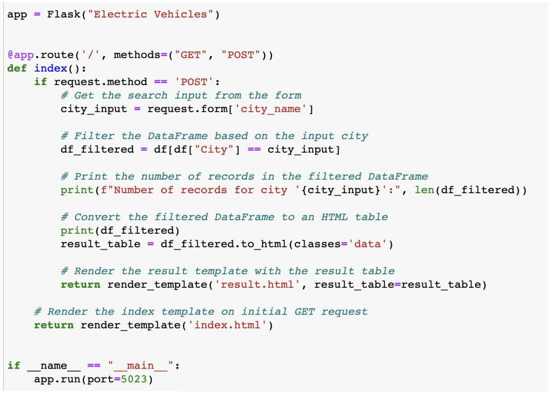
Codes



```
sycopg2
andas as pd
sk import Flask, render_template, request


# Connect to the Postgres database
conn = psycopg2.connect(
    host="localhost",
    port='5432',
    dbname="my_db",
    user="postgres",
    password="123")
conn.autocommit = True


# Create a cursor to execute SQL queries
cur = conn.cursor()

# Execute the SQL query to retrieve data from the database
cur.execute('SELECT * FROM APAN5400_Vehicle_final')

# Fetch all the rows from the result set
ur.fetchall()
```

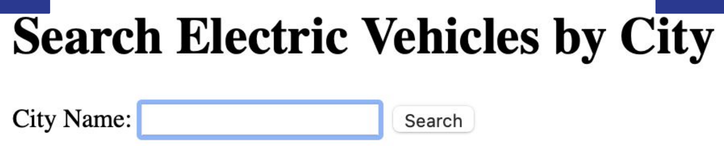


```
# Create a list of dictionaries to store the rows
data = []
for row in rows:
    data.append({
        'VIN': row[0],
        'City': row[1],
        'State': row[2],
        'ZIP Code': row[3],
        'Model Year': row[4],
        'Make': row[5],
        'Model': row[6],
        'Electric Vehicle Type': row[7],
        'Clean Alternative Fuel Vehicle (CAFV) Eligibility': row[8],
        'Electric Range': row[9],
        'Base MSRP': row[10],
        'Vehicle Type': row[11],
        'Fuel Type': row[12]
    })

# Convert the data to a pandas DataFrame
df = pd.DataFrame(data)
```



```
app = Flask("Electric Vehicles")


@app.route('/', methods=("GET", "POST"))
def index():
    if request.method == 'POST':
        # Get the search input from the form
        city_input = request.form['city_name']

        # Filter the DataFrame based on the input city
        df_filtered = df[df["City"] == city_input]

        # Print the number of records in the filtered DataFrame
        print(f"Number of records for city '{city_input}':", len(df_filtered))

        # Convert the filtered DataFrame to an HTML table
        print(df_filtered)
        result_table = df_filtered.to_html(classes='data')

        # Render the result template with the result table
        return render_template('result.html', result_table=result_table)

    # Render the index template on initial GET request
    return render_template('index.html')


if __name__ == "__main__":
    app.run(port=5023)
```

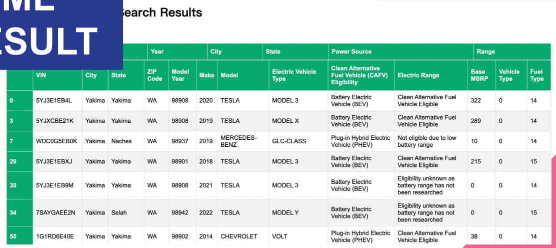
HTML INDEX



Search Electric Vehicles by City

City Name: Search

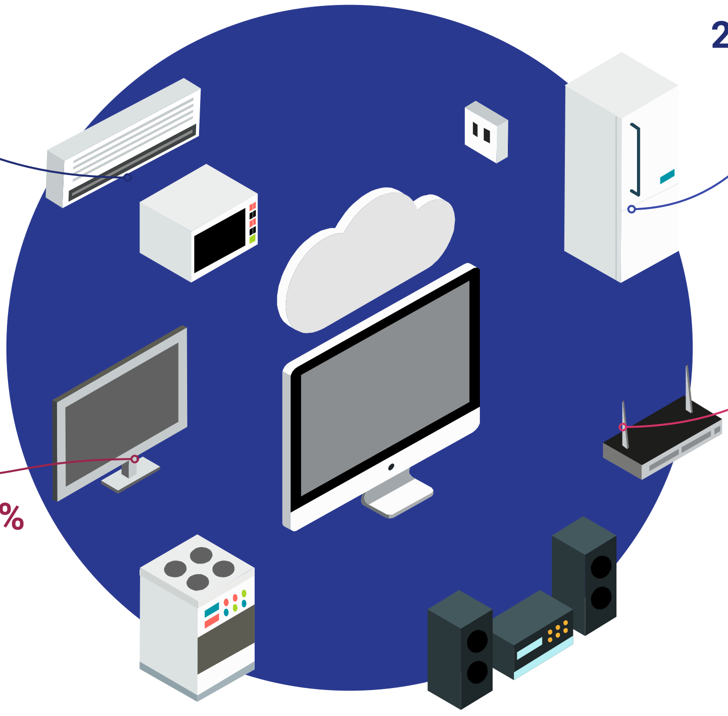
HTML RESULT



earch Results

				Year		City	State		Power Source		Range		
	VIN	City	State	ZIP Code	Model Year	Make	Model	Electric Vehicle Type	Clean Alternative Fuel Vehicle (CAFV) Eligibility	Electric Range	Base MSRP	Vehicle Type	Fuel Type
0	5YJ3E1EB4L	Yakima	Yakima	WA	98908	2020	TESLA	MODEL 3	Battery Electric Vehicle (BEV)	Clean Alternative Fuel Vehicle Eligible	322	0	14
3	5YJXCBE21K	Yakima	Yakima	WA	98908	2019	TESLA	MODEL X	Battery Electric Vehicle (BEV)	Clean Alternative Fuel Vehicle Eligible	289	0	14
7	WDC0G5EB0K	Yakima	Naches	WA	98937	2019	MERCEDES-BENZ	GLC-CLASS	Plug-in Hybrid Electric Vehicle (PHEV)	Not eligible due to low battery range	10	0	14
29	5YJ3E1EBXJ	Yakima	Yakima	WA	98901	2018	TESLA	MODEL 3	Battery Electric Vehicle (BEV)	Clean Alternative Fuel Vehicle Eligible	215	0	15
30	5YJ3E1EB9M	Yakima	Yakima	WA	98908	2021	TESLA	MODEL 3	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not been researched	0	0	14
34	7SAYGAEE2N	Yakima	Selah	WA	98942	2022	TESLA	MODEL Y	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not been researched	0	0	15
55	1G1RD6E40E	Yakima	Yakima	WA	98902	2014	CHEVROLET	VOLT	Plug-in Hybrid Electric Vehicle (PHEV)	Clean Alternative Fuel Vehicle Eligible	38	0	14

Evaluation criteria



User engagement & satisfaction

25%

Users reaction towards our prototype will help us to further improve our prototype. We will consider:

1. Number of active users
2. Average session duration
3. Ratings and reviews from users

Usability/Flexibility

25%

The user interface should be easy to operation and the response time should not be long. We will consider:

1. API response time
2. Operation difficulty
3. Data process speed

Accuracy & Volume

25%

The prototype should reach 95% accuracy. We will also consider:
1. Frequency of updating dataset
2. More data add into the dataset
3. Reliability of our sources

Compatibility

25%

The prototype should work with a variety of browsers and different investment styles

Conclusion and Recommendations

Conclusion

Our engine provides insight into the electric vehicle market in various cities in Washington, D.C. By selling the engine to car sales companies(such as CarMax, Autotrader,etc), they can better use the engine to understand the popularity and sales trends of different models, so they can adjust their product mix and sales strategies to meet consumer demand better.

Future Recommendations

- Since our engine can only provide electric car sales in Washington, D.C., in the future, we can provide more customized services, including cars, electric cars, motorcycles, etc.
- By upgrading the engine to perform more data analysis and reporting services to meet the needs of different auto sales companies.
- As we collect and develop the data, we can build an all-purpose search engine at the future.

Thank you!

Vehicle Population Search Engine

5400 Team project proposal-Group 8

Roles and contributions:

<Team Leader>Yiqing Lin (yl5238) 20% Integrated code contributor
Ting Yi Ho (th3019) 20% SQL Database & Spark code contributor
Tiannuo Yu (ty2474) 20% Flask code contributor
Shan Jiang (sj3231) 20% SQL Database code contributor
Yuxuan Zhu (yz4353) 20% SQL Database code contributor