Melbourne Rental Market Dashboard

Supporting international students & new-comers to Melbourne

Asal Valisoltani
Dhiraj Kapoor
Dominique Dela Cruz
Eloise Moran

Problem statement

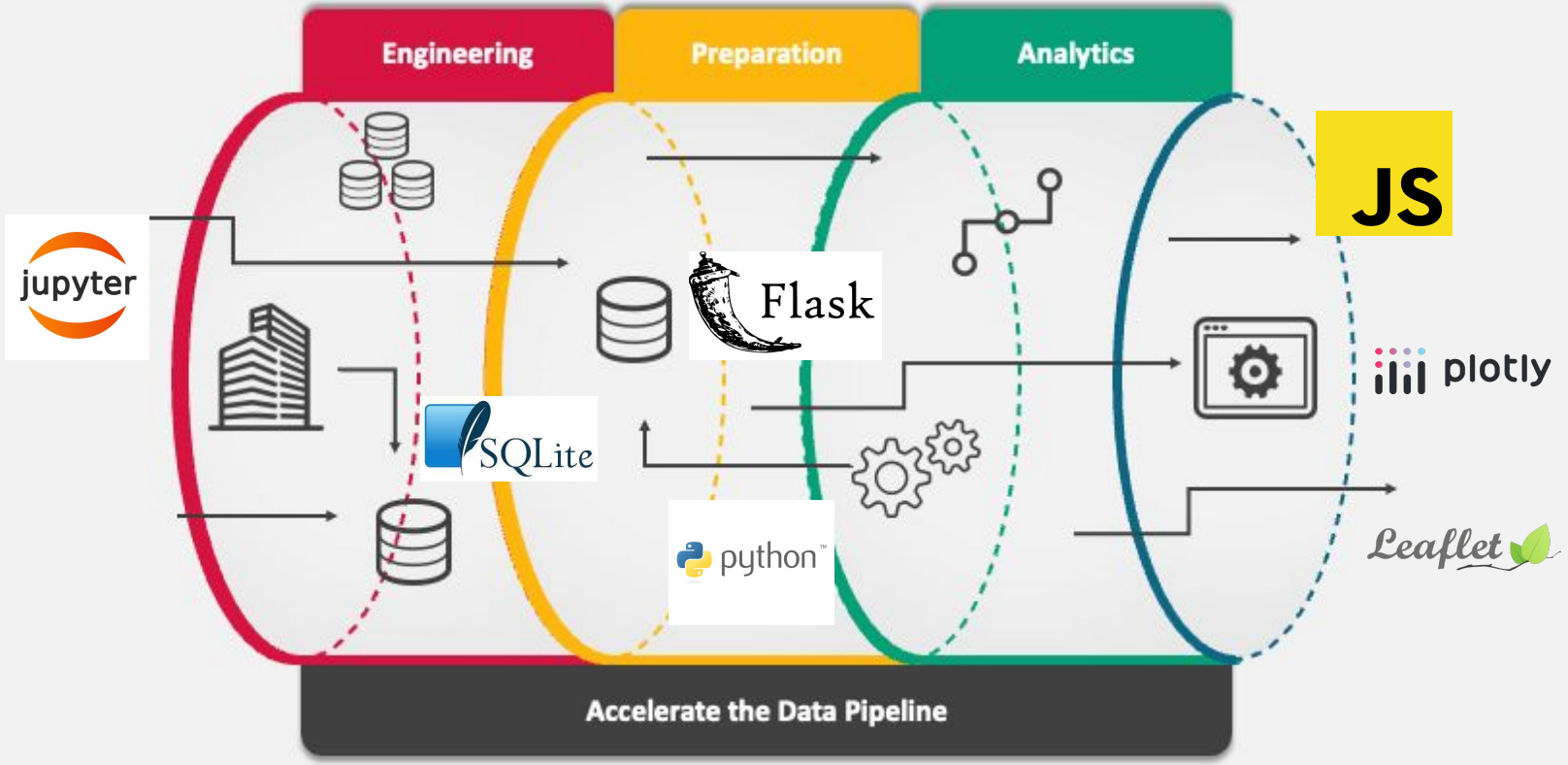
"I have just been accepted into the 2024 Masters in Data Analytics program at RMIT, Melbourne, Australia. I'm preparing my Visa documents and the Immigration Office requires me to find housing before I move into Australia. I would like to find a place that suits the below needs:

1. ***At least within 10kms of my University***
2. ***Average Weekly Rent is below $400 AUD***
3. ***Single bedroom unit is highly preferred***

Currently there is no 'one stop shop' view of Melbourne's suburbs to provide new international students this view of our great city. Students must search across multiple websites (e.g. RealEstate.com, Visit Melbourne, Youtube etc).

The Melbourne Rental Market Dashboard aims to solve this gap in the market.

DATA PIPELINE



Development Workflow

Frontend:

Make API requests to the backend to retrieve data for visualization .
Website Development(HTML, CSS, ...)

Data Source:

Merged: Melbourne Housing Market from Kaggle + Census Dataset

Cleaning and sorting the Main dataset

Visualization:

Leaflet, Plotly, Video JS Library

Backend:

Transferred to SQlite Database and connected to Flask API

Data Transformation

In [46]:
```
Mel_Rental_Market['id'] = range(len(Mel_Rental_Market))
Mel_Rental_Market.head()
```

Out[46]:

	Suburb	Type	Method	Real_Estate_Agent	Distance	Postcode	Number_of_Bedroom	Number_of_Bathroom	Number_of_Carpark	Latitude	Longitude	Mec
0	Abbotsford	h	S	Biggin	2.5	3067	2	1	0	-37.8079	144.9934	
1	Abbotsford	h	SP	Biggin	2.5	3067	3	2	0	-37.8093	144.9944	

In [50]:
```
# Connect to the database
conn = sqlite3.connect('Melbourne.db')
cursor = conn.cursor()

# Load the data into a DataFrame
df = pd.read_sql_query("SELECT * FROM Melbourne", conn)

# Create a new table with a primary key constraint
cursor.execute("CREATE TABLE Melbourne_2 (id INTEGER PRIMARY KEY, Suburb TEXT, Type TEXT, Method TEXT, Real_Estate_Age

# Commit the changes and close the connection
conn.commit()
conn.close()
```

Flask Routes/ Backend


```python
# Flask routes
@app.route('/api/all-data-json', methods= ['GET'])
def get_data():
    results = session.query(Mel_Rental).all()
    data = []
    for row in results:
        data.append({'id': row.id, 'Suburb': row.Suburb , 'Type' : row.Type, 'Method': row.Method, 'Real_Estate_Agent': row.Real_Estate_Agent, 'Distance': row.Distar
        'Postcode': row.Postcode, 'Number_of_Bedroom' : row.Number_of_Bedroom , 'Number_of_Bathroom': row.Number_of_Bathroom , 'Number_of_Carpark': row.Number_of_Ca
       'Latitude': row.Latitude, 'Longitude':row.Longitude, 'Median_rent_weekly': row.Median_rent_weekly})
    session.close()
    return jsonify(data)

@app.route('/api/geojson',methods=['GET','POST'])
def geojson():
    json_url = os.path.join("data","Melb_data.json")
    data_json = json.load(open(json_url))

    return jsonify(data_json)


if __name__ == "__main__":
    app.run(debug=True)
```

Flask Routes/Frontend


```
################################
######## FRONT END #############
################################

@app.route("/")
def home():
    return render_template("index.html", title = "Project 3- Week 15")

@app.route("/overview/")
def proj_overview():
    return render_template("overview.html")

@app.route('/rental-map')
def rental_map():
    return render_template("map.html")


###Data Visualisation Routes
@app.route('/data-viz/pie')
def pie_chart():
    return render_template("plot_pie.html")

@app.route('/data-viz/bar')
def bar_chart():
    return render_template("plot_bar.html")

@app.route('/data-viz/scatter')
def scatter_plot():
    return render_template("plot_scatter.html")

################################
```