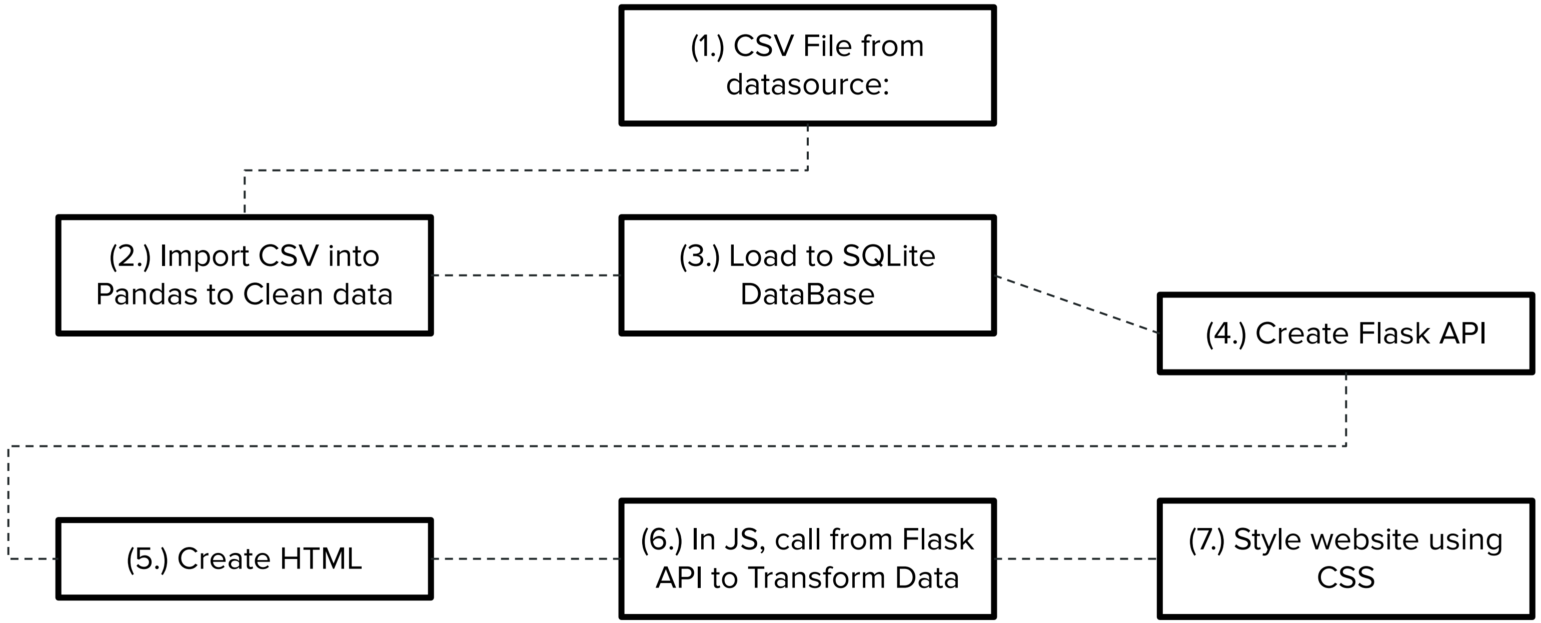
Twin Cities: AirBnB Dataset

Project 3_Group_5: DJ Thapa, Caleb Steeves, Tanner Victorian , John Cuevas Gonzalez

Process Overview:

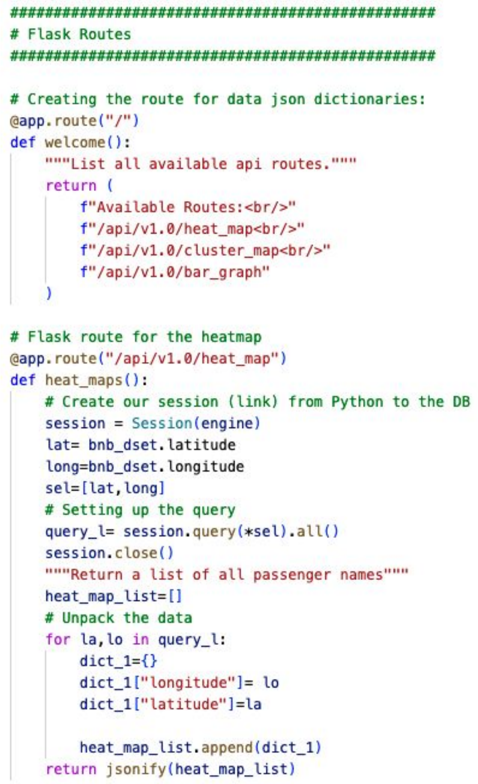

(1.) CSV File from datasource:
(2.) Import CSV into Pandas to Clean data
(3.) Load to SQLite DataBase
(4.) Create Flask API
(5.) Create HTML
(6.) In JS, call from Flask API to Transform Data
(7.) Style website using CSS

Data Source:

- Where did the data come from? -
  - We obtained the data from Insideairbnb.com , which is a website that collects and publishes data on AirBnb listings in many cities around the world.
- What format was it in?
  - The Twin Cities dataset was in CSV format.
- What are metrics or considerations we cared about?
  - We wanted to find data on metrics customers would typically care about when booking an airbnb. We also wanted to find a decent size dataset for meaningful analysis. Our data included a little under 4000 listings. A few key metrics include:
    - Price:
    - Review_Score
    - Cleanliness_Score
    - Counties
    - Coordinates

Data Transform & Load: (DJ)

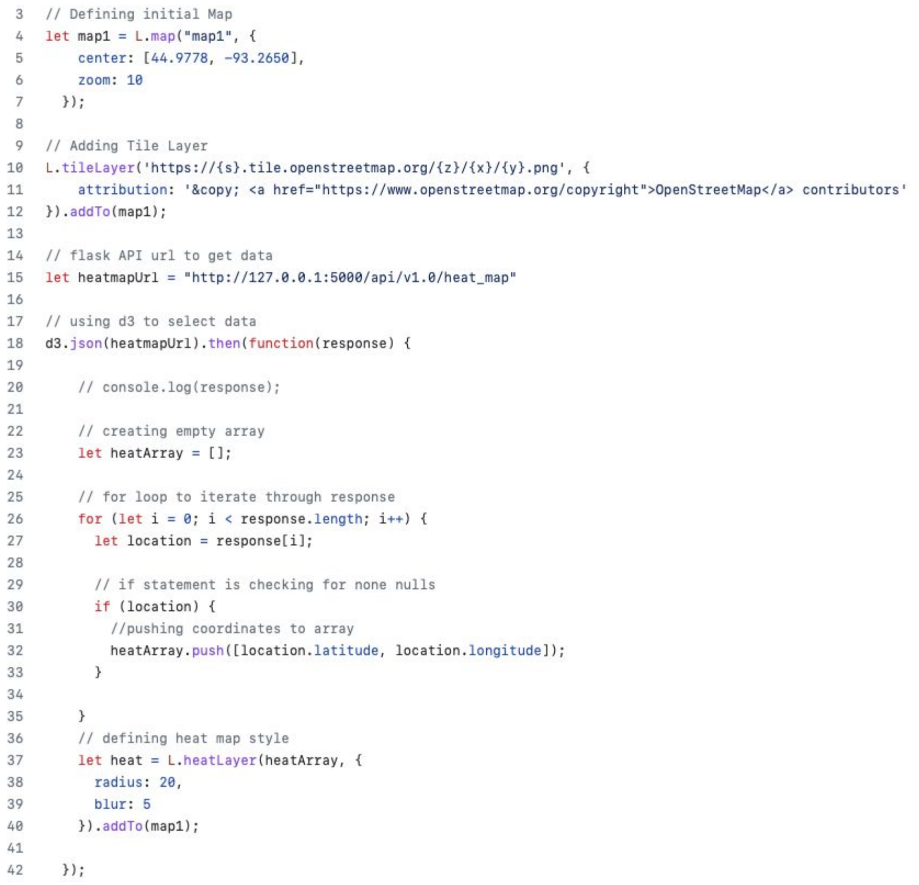
- Loaded information from CSV from our resource
- Cleaned data ➔ Dataframe for the metrics we wanted to look at
- Utilized Pandas to format our dataframe to Sqlite database
- Used Sqlalchemy to query ➔ specific visuals that jsonified data it was feeding into the Flask API:
    - HeatMap
    - Cluster Map
    - Price Gauge
    - Bar Graph



```python
#################################################
# Flask Routes
#################################################

# Creating the route for data json dictionaries:
@app.route("/")
def welcome():
    """List all available api routes."""
    return (
        f"Available Routes:<br/>"
        f"/api/v1.0/heat_map<br/>"
        f"/api/v1.0/cluster_map<br/>"
        f"/api/v1.0/bar_graph"
    )

# Flask route for the heatmap
@app.route("/api/v1.0/heat_map")
def heat_maps():
    # Create our session (link) from Python to the DB
    session = Session(engine)
    lat= bnb_dset.latitude
    long=bnb_dset.longitude
    sel=[lat,long]
    # Setting up the query
    query_l= session.query(*sel).all()
    session.close()
    """Return a list of all passenger names"""
    heat_map_list=[]
    # Unpack the data
    for la,lo in query_l:
        dict_1={}
        dict_1["longitude"]= lo
        dict_1["latitude"]=la

        heat_map_list.append(dict_1)
    return jsonify(heat_map_list)
```

JavaScript Map Creation:

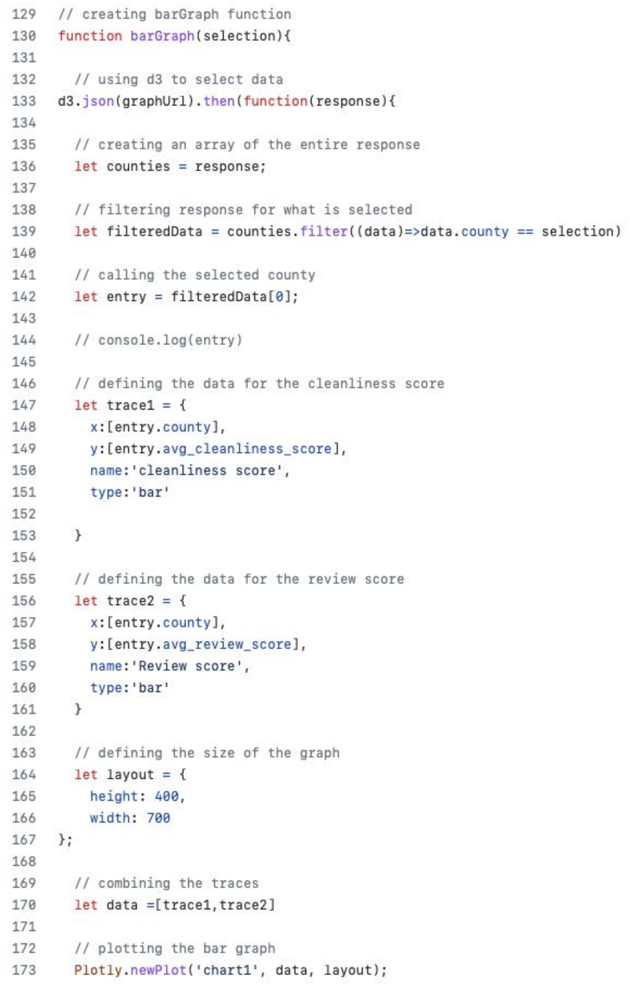
- Define map
- Create tile layer
- Queried from our Flask API
- Used D3 to select data from our query
- Create empty array
- For loop to populate array
- Add to map



```javascript
// Defining initial Map
let map1 = L.map("map1", {
    center: [44.9778, -93.2650],
    zoom: 10
  });

// Adding Tile Layer
L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'
}).addTo(map1);

// flask API url to get data
let heatmapUrl = "http://127.0.0.1:5000/api/v1.0/heat_map"

// using d3 to select data
d3.json(heatmapUrl).then(function(response) {

    // console.log(response);

    // creating empty array
    let heatArray = [];

    // for loop to iterate through response
    for (let i = 0; i < response.length; i++) {
      let location = response[i];

      // if statement is checking for none nulls
      if (location) {
        //pushing coordinates to array
        heatArray.push([location.latitude, location.longitude]);
      }

    }
    // defining heat map style
    let heat = L.heatLayer(heatArray, {
      radius: 20,
      blur: 5
    }).addTo(map1);

  });
```

JavaScript Graph Creation:

- Queried from our Flask API
- Used D3 library to fetch data from our JSON file.
- *Filtered the data to select the county that user selects*
- Define 2 sets of data- one for cleanliness score and another for review score
- Define the layout of the graph
- plot using plotly.



```javascript
// creating barGraph function
function barGraph(selection){

  // using d3 to select data
d3.json(graphUrl).then(function(response){

  // creating an array of the entire response
  let counties = response;

  // filtering response for what is selected
  let filteredData = counties.filter((data)=>data.county == selection)

  // calling the selected county
  let entry = filteredData[0];

  // console.log(entry)

  // defining the data for the cleanliness score
  let trace1 = {
    x:[entry.county],
    y:[entry.avg_cleanliness_score],
    name:'cleanliness score',
    type:'bar'

  }

  // defining the data for the review score
  let trace2 = {
    x:[entry.county],
    y:[entry.avg_review_score],
    name:'Review score',
    type:'bar'
  }

  // defining the size of the graph
  let layout = {
    height: 400,
    width: 700
};

  // combining the traces
  let data =[trace1,trace2]

  // plotting the bar graph
  Plotly.newPlot('chart1', data, layout);
```

JavaScript - Libraries/Plugins:

- Leaflet
- Leaflet-Heat
- Leaflet-MarkerCluster
- Plotly
- D3
- Unique JS Library: VidBG

Conclusion

- Sibley | Pierce has the most bang for your buck listing (least priced and highest review score)
    - Sibley ($133 & 4.98) | Pierce ($255 & 4.91)
- Hennepin has the most listing with 1961 listings ($158/night)
- Ramsey is the most affordable and closet to Metro Areas ($139/ night)
- Le Sueur has the highest cleanliness score 4.92 but also one of the highest ($290/night)