ChatGPT

RightNow NYC Project Plan

Team Roles & Responsibilities

Ammaar — Data Engineer / ML & DevOps

Ammaar handles data processing, clustering, and deployment. In simple terms, he collects NYC location data, cleans it, runs a clustering algorithm to find groups of points, and sets up how the app is hosted. His work turns raw geospatial data into organized clusters and ensures the system runs in production.

- **Data Pipeline:** Ingest and preprocess NYC data (e.g. latitude/longitude lists) using Python scripts.
- **Clustering Logic:** Implement geo-spatial clustering (e.g. scikit-learn's KMeans) to group location coordinates [1] [2]. For example, using `KMeans(n_clusters=5)` on a list of (lat, lon) pairs.
- **Deployment:** Write Dockerfiles or Kubernetes configs to containerize the app and set up CI/CD pipelines.

```
from sklearn.cluster import KMeans

coords = [(40.7128, -74.0060), (40.7138, -74.0050), ...]  # example [lat, lon]
pairs
kmeans = KMeans(n_clusters=5, init='k-means++', random_state=0)
kmeans.fit(coords)
labels = kmeans.labels_
print(labels)  # cluster label for each point
```

Citing scikit-learn clustering example [2].

David — Backend Developer (Flask API)

David builds the server-side API using Flask. Simply put, he writes endpoints to handle requests and serve data. He sets up routes that the frontend can call (e.g. to fetch filtered datasets or submit queries).

- **Flask Routes:** Define API endpoints using Flask's `@app.route` decorator [3]. For example, `@app.route('/api/data')` binds a URL path to a Python function.
- **JSON Responses:** Return data as JSON by returning Python dicts or lists (Flask auto-serializes them) [4].
- **Data Integration:** Query and manipulate data (e.g. from a database or file) inside each route. Handle query parameters and input validation.
- **Error Handling:** Implement error catching (e.g. return 400/404 codes) and logging for robustness.

```python
from flask import Flask
app = Flask(__name__)

@app.route('/api/hello')
def hello():
    data = {"message": "Hello, World!"}
    return data  # Flask converts this dict to JSON [4]

# Example of binding a URL to a function (Flask docs example) [3]:
@app.route('/hello')
def greet():
    return "Greetings from Flask!"
```

Nikhil — Frontend Developer (Map)

Nikhil implements the interactive map component in React. In plain terms, he creates the map view of NYC and plots data points on it. He uses a mapping library to render a map, add markers for locations, and handle user interactions (like clicking on markers).

- **Map Component:** Use React Leaflet (or similar) to render a map. For example, a `<MapContainer>` with a `<TileLayer>` for map tiles [5].
- **Markers & Popups:** Add `<Marker>` (and optional `<Popup>`) elements at data coordinates to display info when clicked.
- **State Management:** Control the map center, zoom level, and re-render when data or filters change.

```jsx
import { MapContainer, TileLayer, Marker, Popup } from 'react-leaflet';

const position = [40.7128, -74.0060];  // Center at NYC
<MapContainer center={position} zoom={13} scrollWheelZoom={false}>
  <TileLayer
    attribution='&copy; OpenStreetMap'
    url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png"
  />
  <Marker position={position}>
    <Popup>NYC City Center</Popup>
  </Marker>
</MapContainer>
```

Example React Leaflet code from docs [5].

Ethan — Frontend Developer (UI/UX)

Ethan designs and implements the user interface using React and Tailwind CSS. In simple terms, he builds the page layouts and styles them. He ensures the app looks good and works well on all devices.

- **UI Components:** Create React components for the layout (navbars, dashboards, forms, cards).
- **Tailwind Styling:** Apply Tailwind utility classes for styling. For example, set background, text color, spacing, and hover effects using class names [6].
- **Responsive Design:** Use Tailwind's responsive classes to make the UI adapt to different screen sizes (mobile, tablet, desktop).

```
<!-- Example Tailwind-styled button from docs: -->
<button class="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4
rounded">
  Click Me
</button>
```

Tailwind CSS example using utility classes [6].

---

[1] [2] Clustering Geospatial Data
https://engo645-assignments.readthedocs.io/en/latest/notebooks/L5/Clustering_Geospatial_Data.html

[3] [4] Quickstart — Flask Documentation (3.1.x)
https://flask.palletsprojects.com/en/stable/quickstart/

[5] React components for Leaflet maps | React Leaflet
https://react-leaflet.js.org/

[6] Buttons - Tailwind CSS
https://v1.tailwindcss.com/components/buttons