- **Group 159**
- **Nathan Kaul, Antonio Vega, Julio Figueredo-Alvarez**
- **Ride-Sharing Demand Zones: K-Means vs DBSCAN Clustering**

1. **Problem:** Ride-sharing companies need to identify areas of high demand within a city to optimize service allocation and reduce wait times. Current approaches often rely on aggregated statistics rather than spatial clustering of actual pickup locations and times. Our project aims to uncover these demand hotspots using clustering algorithms.
2. **Motivation:**Understanding where and when rides are most frequently requested can help drivers, companies, and city planners make better decisions. Detecting both common zones (commutes, airports) and rare or dense zones (nightlife, events) provides actionable insights for improving transportation efficiency.
3. Features: Successfully implement K-Means and DBSCAN from scratch on NYC ride data. Identify meaningful clusters of pickup locations (major hotspots vs smaller dense zones). Provide clear visualizations (static scatter plots or density maps) showing cluster locations. Optionally leave the structure open to future interactive maps with sliders for hourly data.
4. **Data**: TLC Trip Record Data - TLC
5. **Tools:** Python, NumPy, Pandas, Matplotlib, Seaborn, scikit-learn (for metrics, not algorithms)
6. **Visuals**: Menu driven interface, plots, charts, graphs, streamlit
7. **Strategy**:**Algorithms:**

- **K-Means:** Initialize centroids, assign points to nearest centroid, update centroids iteratively.
- **DBSCAN:** Use radius (ε) and minimum points (minPts) to find dense clusters and label noise.
- **Data Representation:** Each ride represented as a 2D or 3D vector [latitude, longitude, hour(optional)]
  Stored as a NumPy array for efficient distance computations.

8. **Distribution of Responsibility and Roles:**

- Member 1 responsible for implementation of kmeans algorithm
- Member 2 responsible for implementation of DBSCAN
- Member 3 responsible for interface and visuals