Zypp Data Scientist Assignment

Problem Statement:

Design a dashboard to visualize trip details across the city, focusing on trips with durations of x minutes. The dashboard should display the start and end points of each trip on a map and use color-coding to differentiate trips based on their duration.

With the following requirements:

1. Display the start and end points of each trip on a map.
1. Connect the start and end points with colored lines to represent the trips.
2. Apply the following color-coding scheme to the lines and points based on trip duration:
3. Color 'A' for trips lasting less than 5 minutes.
4. Color 'B' for trips lasting between 6 to 15 minutes.
5. Color 'C' for trips lasting between 16 to 20 minutes.
6. Color 'D' for trips lasting between 21 to 30 minutes.
7. Color 'E' for trips lasting more than 30 minutes.

Expected Technical Constraints:

1. Volume Handling: The trip data received every minute can be more than a GB,(as zypp plans to reach the goal of adding ~200000 vehicles in the fleet and has been in operation since 2017) and hence complete data cannot be consumed at once.
2. Frontend Capacity: Frontend of the dashboard cannot consume more than 5000 points at once.
3. Display Limit: The dashboard should limit the display to a maximum of 500 points (say *g*) on the overview to ensure clarity and readability.

Meeting the technical constraints:

1. Volume Handling: Using batching we can divide the data into fixed size, and easily add and then remove from the backend of the dashboard.
2. Frontend Capacity: reduce/combine points at the backend rather than after receiving on the frontend.
3. Display Limit: Use a clustering approach to combine similar points.

Solution:

1. Send the batched data of completed trips to the backend of the dashboard at fixed intervals.
2. Deploy an unsupervised clustering algorithm like Kmeans++ to combine the trips with the following similar characteristics.
    a. Duration of the trip.
    b. Start of the trip.
    c. End of the trip.
3. Create *g* clusters, and calculate the centroid points of start and end location and average of the trip duration and return to the frontend.
4. For each of the cluster, mark the centroid lat,long of start point and end point on the map, and connect it with a straight line.
5. Use the average trip duration of the cluster to color connected line.
6. Use a tooltip to display the number of points that lies in the cluster.

Limitations:

1. The points on the dashboard are aggregated and hence does not represent the actual data but a similar distribution of the data.
2. The trip durations are averaged, and the standard deviation is ignored, and hence the underlying trip duration of a line coded as D can deviate by +- x mins.

Note: A numerical accuracy of the above limitations should be calculated and reported.

Estimated Time Required:

1. While this also depends on the techstack and current data base schema, I believe most of the time will be taken by deciding the parameters of the clustering algorithm and batch size which needs an understanding of the underlying data distributions.

Success Metric:

With the HEART Goal:

1. Happiness: Displaying the dashboard in a readable format. (Feedback)
2. Engagement: Usefulness of the dashboard. (number of features added / number of intended features)
3. Adoption: Adding the new points. (memory available)
4. Retention: caching the previous points.
5. Task: Loading the dashboard with low latency. (time taken to render dashboard)

Closing Notes:

1. The dynamic loading and regeneration of the dashboard is not considered.
2. An alternative approach to using clustering method, is to calculate the an h3 hash of every lat,long point with a fixed resolution of 8. (say, ~0.7km^2) to combine similar start and end point.