### **Notes on the UI Implementation**

The UI for the **DDS\_Project Database Comparison** was designed and implemented using **Flask**, providing an interactive way to compare the performance metrics of three databases (**MongoDB**, **DynamoDB**, and **Elasticsearch**) for regional and global content recommendations. Here is an overview:

### **Purpose**

The purpose of the UI is to:

1. Allow users to select a **region** (e.g., Europe, Asia, North America, South America).
2. Display and compare the performance metrics for **local and global recommendation queries** executed on the three databases.
3. Present actionable insights, such as **recommendations for improvement**, and display the best-performing query results interactively.

### **Key Features**

1. **Region Selection**:
   * A dropdown menu allows users to choose a specific region (e.g., Asia, Europe).
   * Dynamically fetches the performance metrics for the selected region and database.
2. **Performance Metrics Display**:
   * Metrics for each database are displayed in an interactive bar chart using **Plotly**.
   * Metrics include:
     + Execution Time (ms)
     + Throughput (queries/sec)
     + Average Response Time (ms)
     + CPU Utilization (%)
     + Memory Utilization (%)
3. **Recommendation Results**:
   * Based on the performance comparison, the UI provides actionable suggestions to improve database performance for the selected region.
   * Displays the actual **top 10 regional recommendations** or **top 5 global recommendations** for the best-performing database.
4. **Backend Integration**:
   * The application uses Flask to serve the data and generate interactive charts dynamically using **Plotly** and **Pandas**.

### **How It Works**

1. **Route /**:
   * Serves the main landing page (index.html) for user interaction.
2. **Route /metrics/<database>/<region>**:
   * Fetches performance metrics and recommendations for a specific database and region.
   * Processes the metrics into a **Plotly bar chart** for visualization.
   * Returns the chart data, raw metrics, and recommendations in JSON format for rendering in the frontend.

### **Technology Stack**

1. **Frontend**:
   * **HTML** and **CSS**: Basic structure and styling.
   * **Plotly.js**: Interactive bar charts for metric visualization.
2. **Backend**:
   * **Flask**: Serves routes, processes data, and generates metrics.
   * **Pandas**: Processes metrics for visualization.
3. **Data Format**:
   * Performance metrics and recommendations are stored in a nested dictionary structure for easy retrieval.