**Due Date**: 9th November 2024

**Max Marks:** 100

**Instructions:**

• All code should be well-commented and easy to read.

• You are allowed to use external libraries such as pandas, numpy, matplotlib,

sqlalchemy, pymongo, Flask, Streamlit, and kafka-python.

• Submit your solution by providing a link to a GitHub repository that

contains:

**–** Python scripts (Jupyter notebooks or .py files) for the tasks.

**–** SQL and MongoDB queries in a .txt file.

**–** CSV files, graphs, and any other outputs required by the tasks.

**–** A README.md file explaining how to run the project locally.

**–** The hosted link to your Streamlit web application.

**Question: Full-Stack Data Science Application**

**using Global Weather Dataset**

You are required to create a full-stack data science project that involves data

processing, data analytics, Kafka-based data streaming, database operations,

and hosting a web application using Streamlit. For this assessment, you will

use the **World Weather Dataset** from Kaggle, which can be downloaded

from the following link:

URL:

https://www.kaggle.com/datasets/nelgiriyewithana/global-weather-repository

**Step-by-step Tasks:**

1. **Dataset Exploration and Justification (5 Marks):**

• Download the dataset from Kaggle.

• Provide a brief justification (3-4 lines) for why this dataset is suitable for weather-related analysis and predictions on a global scale.

2. **Data Processing (20 Marks):**

• Load the dataset into a pandas DataFrame.

• Clean the dataset (e.g., handle missing values, convert columns to

appropriate data types).

• Store the cleaned data in a CSV file with appropriate column

names and formats.

• Display a summary of key statistics from the dataset (e.g., mean,

max, and min temperatures, humidity levels, etc.).

3. **Data Analytics and Visualization (20 Marks):**

• Using pandas and numpy, perform the following analyses:

**–** Generate a summary based on the global weather dataset

(e.g., top 5 hottest and coldest locations globally).

**–** Group the data by a relevant field (e.g., region, year) and

compute the average, maximum, or minimum temperature,

humidity, or precipitation.

**–** Plot at least two different visualizations (e.g., histogram of

temperatures, line graph showing changes in temperature or

precipitation over time for a specific region).

4. **Kafka Producer and Consumer for Streaming Data (20 Marks):**

• Set up a Kafka topic named global weather.

• Write a Kafka Producer script that simulates real-time weather

updates for global locations based on your dataset. Each second, send an updated data point (e.g., temperature, humidity, precipitation) to the Kafka topic.

• Write a Kafka Consumer script that listens to the global weather topic and logs the updated data to a CSV file.

• Display the summary of the updates consumed after running the Producer for 60 seconds.

5. **Database Operations (20 Marks):**

• Create a MySQL database named GlobalWeatherDB and import

the cleaned dataset.

• Write SQL queries to:

**–** Retrieve the top 5 locations with the highest temperatures or lowest precipitation.

**–** Retrieve all records for a specific date or condition (e.g., temperature ¿ 35°C, precipitation ¿ 100 mm).

**–** Perform a group by operation (e.g., average temperature by

country, total precipitation by region).

• Create a MongoDB database and import the same dataset into a collection named GlobalWeather data.

• Using MongoDB queries, retrieve:

**–** All records that match a specific condition (e.g., data from a

specific month or continent).

**–** The top 3 records with the highest or lowest values for a

specific metric (e.g., hottest locations, days with the highest

precipitation).

6. **Streamlit Web Application (20 Marks):**

• Use Flask to build a REST API that serves the cleaned data and streams updated global weather data via Kafka.

• Build a web application using Streamlit that interacts with this Flask API.

• The web application should have two sections:

**– Data Dashboard:** Display the cleaned global weather dataset with filters (e.g., by date, region, or other relevant metrics).

**– Live Data Updates:** Show live weather updates being streamed via Kafka, with a real-time plot of the data points.

• Deploy the Streamlit app on the free Streamlit cloud platform and

include the link to the hosted app in your GitHub README.md

file.

**Note:** Submit the link to your GitHub repository containing the code, queries, CSV outputs, and a README.md file. The README should include instructions on how to run the project locally and the hosted Streamlit app link