**Project Overview & Setup Guidelines**

**Objective**

The capstone project is designed to assess foundational skills in Python programming, SQL & Apache Spark, focusing on the practical application of Spark's APIs. This project serves as an opportunity for team members to demonstrate their understanding of key Spark concepts and their ability to implement them effectively in a data engineering context.

**Dataset**

The project will utilize a publicly available dataset located at s3://hls-eng-data-public/data/synthea/fhir/fhir/. This dataset serves as the source data for the project. Participants are required to ingest, process, and manipulate this data using Spark to meet the objectives outlined in the project instructions.

**Prerequisites**

Participants are expected to have completed their initial training and to possess familiarity with the following components and concepts in Apache Spark:

* SparkContext: The core entry point for Spark functionalities.
* SparkSession: The unified entry point for Spark's functionality, introduced in Spark 2.0.
* DataFrame: A distributed collection of data organized into named columns, analogous to a table in a relational database.
* DataFrameReader: Used to load data from various sources into a DataFrame.
* DataFrameWriter: Used to save a DataFrame to a specified data source.
* Functions from the pyspark.sql.functions module: Includes built-in functions for data manipulation and transformations.
* Python programming
* Docker/Kubernetes
* DBT

**Guidelines**

Throughout this project, participants will be provided with a series of instructions to complete specific tasks using the s3://hls-eng-data-public/data/synthea/fhir/fhir/ dataset. These tasks are designed to test both theoretical understanding and practical proficiency with skills. It is expected that participants will:

* Leverage their existing knowledge acquired during the training phase.
* Utilize additional resources such as the [Spark API Documentation](https://spark.apache.org/docs/latest/api/python/) to resolve challenges and extend their understanding.
* Interact with the source data effectively, employing appropriate Spark methods for ingestion and transformation.

**Scope**

**Data Lake Creation**: Ingest and organize the healthcare FHIR data into an AWS S3-based data lake.

Implement appropriate partitioning and structuring of data to optimize query performance and scalability. Process and transform the raw FHIR data into curated datasets suitable for analytics.

**Data Processing and Transformation**:

Utilize Apache Spark and AWS services (e.g., AWS Glue (Optional), Athena) for processing the FHIR data bundles. Extract meaningful insights from structured and semi-structured healthcare data, ensuring data quality and consistency.

**Operational Analytics Dashboard**:

Develop an AWS QuickSight dashboard to visualize key operational metrics and trends derived from the FHIR data. Design interactive charts and graphs that provide actionable insights, such as patient demographics, healthcare utilization, and other relevant metrics.

**Key Deliverables:**

* A fully functional AWS S3-based data lake housing healthcare FHIR data.
* Curated datasets optimized for analytical queries.
* An AWS QuickSight dashboard with multiple operational charts providing insights, such as:
* Patient distributions by age, gender, and location.
* Healthcare service utilization patterns.
* Trends in specific healthcare metrics derived from FHIR bundles.

**Technologies and Tools:**

* AWS Services: S3, Glue, Athena, QuickSight
* Data Processing: Apache Spark, PySpark
* Data Format: FHIR bundles (JSON/Parquet)

**Outcome**:  
The project will result in a robust and scalable data lake architecture in AWS S3 and a user-friendly QuickSight dashboard, enabling stakeholders to derive meaningful insights and make data-driven decisions from healthcare FHIR data.

**Exercise Details**

#### **1. Setting Up AWS Environment**

* **Objective**: Configure AWS resources for the project.
* **Steps**:
  1. Raise IT Ticket to get access to AWS Account (**679994934532**)
  2. Set up an **S3 bucket** for storing raw, processed, and curated datasets.
     + Example structure:

s3://your-bucket-name/raw-data/ (Raw)

s3://your-bucket-name/processed-data/ (Refined)

s3://your-bucket-name/curated-data/

* 1. Configure AWS CLI for local development.
  2. Set up an **IAM role** with permissions for S3, Glue, Athena, and QuickSight.

#### **2. Data Ingestion**

* **Objective**: Load FHIR data from the public dataset into the raw zone of the S3 data lake.
* **Steps**:
  1. Write a Python script using **Boto3** to copy data from the source S3 bucket (s3://hls-eng-data-public/data/synthea/fhir/fhir/) to your bucket's raw-data folder.
  2. Verify the data is stored correctly in your S3 bucket.
  3. While creating raw layer, add audit columns for source file name, insert timestamp,update\_timestamp(This value should be updated when you modify record in refined layer)

#### **3. Data Processing and Transformation**

* **Objective**: Use Apache Spark to process raw FHIR data into structured data suitable for analysis.
* **Steps:**
  1. Load FHIR data from raw-data folder and parse it to create output tables based on all resource\_types.
  2. **Acceptance criteria**: - This step should create below output tables
     + allergyintolerance
     + careplan
     + careteam
     + claim
     + condition
     + device
     + diagnosticreport
     + documentreference
     + encounter
     + explanationofbenefit
     + imagingstudy
     + immunization
     + location
     + medication
     + medicationadministration
     + medicationrequest
     + observation
     + organization
     + patient
     + practitioner
     + practitionerrole
     + procedure
     + provenance
     + supplydelivery

#### **4.Cataloging Data**

* **Objective**: Register datasets with AWS Glue Catalog for querying with Athena. You can also use Dbeaver as alternative option.
* **Steps**:
  1. Use **AWS Glue Crawler** to scan the processed-data folder and create a table in the Glue Data Catalog.
  2. Verify the table structure in the Glue Console.

#### **5. Analytical Querying with Athena**

* **Objective**: Query the processed data to prepare insights for the dashboard.
* **Steps:**
  + Setup DBT – Clone DBT Repo from [git@git:databuildtool/dbt.git](mailto:git@git:databuildtool/dbt.git)
  + Run your DBT Models on Thrift server
  + Create Terget Tables using above resource\_type tables as source, These target tables should be used for creating below Dashboards in QuickSight.
    - patient\_distribution (Columns: patient\_id, age, gender, count)
    - service\_utilization(Columns: patient\_id, encounter\_id, service\_type, date, count) –

*(Data sourced from encounter, careplan, procedure, immunization, and medicationrequest)*

* + - metric\_trends(Columns: metric\_id, date, metric\_value, trend\_indicator)-

*(Data sourced from observation, diagnosticreport, and imagingstudy)*

* + - patient\_geography(Columns: patient\_id, location\_id, city, state, zip, count)-

*(Data sourced from patient, location, and organization)*

* + - outcomes\_and\_risk (Columns: patient\_id, condition\_id, outcome\_status, risk\_score, at\_risk\_flag)-

*Data sourced from condition, allergyintolerance, explanationofbenefit, and provenance)*

#### **6. Creating the QuickSight Dashboard**

* **Objective**: Visualize operational insights from the curated data.
* **Steps**:
  1. Set up **AWS QuickSight** and connect it to your Glue Catalog or Athena.
  2. Import the curated datasets from S3 or Athena into QuickSight.
  3. Create the following visualizations:
     + **Patient Distribution by Age and Gender**: Bar or pie chart.

**Key Metrics:**

* Total number of patients segmented by age groups (e.g., 0–18, 19–35, 36–50, 51–65, 66+).
* Gender breakdown (e.g., Male, Female, Other) within each age group.

**Purpose**: Provide an overview of the patient population, helping stakeholders understand demographic patterns.

**Visual Elements**:

* A bar chart showing age groups on the x-axis and patient count on the y-axis, with bars color-coded by gender.
* Alternatively, a pie chart showing the percentage distribution of genders across the entire dataset.
  + - **Healthcare Service Utilization Patterns**: Line chart showing service usage over time.

**Key Metrics:**

* Number of healthcare services utilized over a time period (e.g., months, quarters, years).
* Categories of services (e.g., outpatient visits, inpatient admissions, lab tests, medications).

**Purpose**: Track how healthcare services are being utilized over time, identifying trends and peak periods.

**Visual Elements:**

* A line chart with time on the x-axis and service utilization count on the y-axis.
* Use separate lines or markers for each service type.
* Add filters for stakeholders to focus on specific time periods or service categories.
  + - **Trends in Specific Metrics**: Line or area chart showing trends (e.g., hospital visits, medication usage).

**Key Metrics:**

* Number of hospital visits per month/quarter.
* Number of medications prescribed over time.

**Purpose**: Highlight patterns in patient engagement with healthcare systems and medication consumption.

**Visual Elements:**

* An area chart showing hospital visits and medication usage on the y-axis, with time on the x-axis.
* Add trendlines to forecast future patterns.
* Use distinct color gradients to differentiate between hospital visits and medication usage.
  + - **Patient Segmentation and Insights (Geographic Distribution of Patients)**

**Key Metrics**:

* Patient count segmented by geographic regions (e.g., states, cities, or ZIP codes).

**Purpose**:Understand patient distributions geographically to aid in resource allocation and regional planning.

**Visual Elements**:

* A **heatmap** or **geo-map** highlighting patient density across different regions.
* Use tooltips to display exact patient counts when hovering over specific areas.
  + - **Healthcare Outcomes and At-Risk Insights : At-Risk Patient Metrics**

**Key Metrics**:

* Number of patients flagged as "At Risk" (e.g., based on specific medical conditions).
* Trends in healthcare costs or resource utilization for at-risk patients.

**Purpose**: Provide actionable insights into high-priority patient groups and their healthcare needs.

**Visual Elements**:

* A **bar chart** showing at-risk patient counts by age group or gender.
* Include a **cost trendline** overlay to show healthcare expenses associated with these patients.
  1. Customize the dashboard with filters, drilldowns, and interactivity. Each Dashboard should have ability to show trend in hour, date, weeks timeframe.

**Submission Requirements**

1. **S3 Bucket Structure**: Demonstrate the organization of raw, processed, and curated datasets.
2. **PySpark Script**: Provide the Python script for data ingestion and transformation.
3. **Glue Table**: Verify the table schema in Glue Catalog or in DBeaver.
4. **DBT Models**: Run DBT Job for data analysis.
5. **QuickSight Dashboard**: Share screenshots or a link to the operational dashboard showcasing insights.