📊 Project Documentation: Data Ingestion from S3 to RDS with Fallback to AWS Glue Using Dockerized Python App

🎯 Objective:

Develop a Dockerized Python application to:
- Read data from an Amazon S3 bucket
- Insert data into an RDS (MySQL-compatible) database
- If RDS is unavailable or the upload fails, fallback to AWS Glue (catalog table creation + schema registration)

🧩 PHASE 1: LOCAL/ENVIRONMENT SETUP

☑ Step 1: Launch an EC2 Instance

- Go to AWS Console > EC2 > Launch Instance
- Choose Ubuntu 20.04 LTS (Free Tier eligible)
- Instance type: t2.micro
- Configure key pair
- Allow ports: SSH (22), MySQL (3306)

☑ Step 2: Login to Instance

ssh -i your-key.pem ubuntu@your-ec2-ip

☑ Step 3: Create Project Directory

mkdir s3-to-rds-glue-fallback
cd s3-to-rds-glue-fallback

☑ Step 4: Install All Tools & Services

-
  - Python 3.9+

- sudo apt update
- sudo apt install python3.9 python3.9-venv python3.9-dev -y

- PIP & Virtualenv

- python3.9 -m ensurepip --upgrade
- pip install virtualenv

- Docker

- sudo apt update
- sudo apt install docker.io -y
- sudo systemctl enable docker
- sudo systemctl start docker
- sudo usermod -aG docker $USER
- Logout and re-login to apply Docker group permissions

- AWS CLI

- curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
- unzip awscliv2.zip
- sudo ./aws/install
- aws configure

- MySQL Client

- sudo apt install mysql-client -y

- Optional Tools (Curl/Git)

- sudo apt install curl git -y

- Install Python Libraries (locally)

- pip install boto3 pandas sqlalchemy pymysql

PHASE 2: AWS SERVICE SETUP

Step 5: Create an S3 Bucket

- Go to S3 > Create Bucket
- Name: my-data-ingestion-bucket
- Region: us-east-1

- (Optional) Uncheck "Block all public access" if needed

Upload people.csv file:

```
id,name,email
1,Shubham,shubham@example.com
2,Alice,alice@example.com
```

☑ Step 6: Create an RDS MySQL Instance

- Go to RDS > Create Database
- Engine: MySQL
- Username: admin, Password: *******
- Public Access: Enabled (for testing only)
- Security group: Open port 3306 to your IP

```
mysql -h <rds-endpoint> -u admin -p
```

```
CREATE DATABASE data_ingestion_db;
USE data_ingestion_db;
CREATE TABLE people (
  id INT PRIMARY KEY,
  name VARCHAR(50),
  email VARCHAR(100)
);
```

☑ Step 7: Create AWS Glue Database

- Go to Glue > Databases > Create Database
- Name: my_glue_db

🧠 Step 8: main.py (Python Script)

Handles S3 download, RDS insert, and Glue fallback.

📄 Step 9: requirements.txt

```
boto3
pandas
sqlalchemy
pymysql
```

🔐 Step 10: .env File Example

```
AWS_ACCESS_KEY_ID=your-access-key
AWS_SECRET_ACCESS_KEY=your-secret-key
AWS_DEFAULT_REGION=us-east-1

S3_BUCKET=my-data-ingestion-bucket
S3_KEY=people.csv

RDS_HOST=mydb-instance.rds.amazonaws.com
RDS_PORT=3306
RDS_DB=data_ingestion_db
RDS_USER=admin
RDS_PASSWORD=shubham123456
RDS_TABLE=people

GLUE_DB=my_glue_db
GLUE_TABLE=people_glue
GLUE_S3_LOCATION=s3://my-data-ingestion-bucket/
```

🏗 Step 11: Build Docker Image

```
docker build -t s3-to-rds-glue-app .
```

🚀 Step 12: Run Docker Container

```
docker run --env-file .env s3-to-rds-glue-app

sudo docker run --env-file .env s3-to-rds-glue-app

docker run --rm -it --env-file .env -v $(pwd)/app:/app s3-to-rds-glue-app
```

☑ Validation

✅ Check MySQL RDS table: SELECT * FROM people;

✅ If RDS fails, check AWS Glue > Tables > people

✅ Logs from container should indicate fallback or success

☑ Final Notes

```
git init
git remote add origin https://github.com/youruser/s3-rds-glue-pipeline.git
git add .
git commit -m "Initial commit"
git push -u origin main
```

- Document your .env file setup securely
- Rotate AWS keys after test usage
- Clean up AWS resources to avoid unnecessary billing

📌 GitHub Repository Link

https://github.com/ShubhamSarkar516/Data_Ingestion_S3_RDS_GLUE