📘 Project Title: Serverless Data Pipeline using AWS Lambda and AWS Glue for CSV Transformation

🧩 Project Scenario:

A user uploads raw CSV files into an S3 bucket. These files may contain duplicate records or inconsistent formatting. The goal is to automate a transformation pipeline such that:

- ✅ A Lambda function is triggered when a new CSV file is uploaded.
- ✅ The Lambda function reads the CSV, removes duplicates, and stores the cleaned CSV in another S3 folder.
- ✅ An AWS Glue job automatically reads the cleaned CSV, performs further transformations (e.g., uppercasing name fields), and stores the final output in a third S3 folder.

🏗️ Solution Architecture:

📁 1. S3 Bucket Structure:

- `lambda_folder/input_folder/` – Raw CSV files uploaded here
- `lambda_folder/output_folder/` – Cleaned CSV output from Lambda
- `glue/final_output/` – Final transformed CSV output from Glue

🧠 2. Lambda Configuration:

- **Trigger:** S3 event notification on `lambda_folder/input_folder/`
- **Functionality:**
  - Read CSV file from S3
  - Remove duplicate rows
  - Save cleaned CSV to `lambda_folder/output_folder/`
- **Key Fixes:**
  - Used `.decode('utf-8-sig')` to remove BOM
  - Used `csv.reader` and `csv.writer` with deduplication logic

🔍 3. Lambda Code Summary:

- Reads file from S3 input folder
- Deduplicates rows
- Writes clean data as CSV back to S3

🔧 4. Glue Configuration:

- **Trigger:** Manual run or triggered via EventBridge
- **Input Path:** `lambda_folder/output_folder/`
- **Output Path:** `glue/final_output/`
- **Transformations:**
  - Reads CSV with `.option("header", "true")`
  - Applies transformation: uppercasing “name” column (if it exists)
  - Writes final CSV with header to output folder

💻 5. Glue Code Summary:

```
spark.read.option("header", "true").option("inferSchema",
"true").csv(input_path)
```

- Ensures header recognition
- Applies transformation
- Uses `.write.mode("overwrite").option("header", "true").csv(output_path)`

🚧 Common Issues & Fixes:

Issue	Cause	Fix
`AnalysisException` / File Not Found	BOM issues, missing files, malformed rows	Cleaned with Lambda; verified file presence
IAM Permission Errors	IAM role lacked `s3:GetObject`, `s3:PutObject`	Added correct IAM policies to Lambda and Glue roles
Format Mismatch	Initially tried JSON	Switched to CSV for simplicity

---

🧪 Test Dataset Used:

Book1.csv:

```
id,name,email,age
1,Alice,alice@example.com,28
2,Bob,bob@example.com,34
3,Charlie,charlie@example.com,30
2,Bob,bob@example.com,34
```

☑️ Final Result:

- Fully automated, serverless CSV processing pipeline
- Cleaned duplicate rows using Lambda

- Uppercased `name` field using Glue
- Final transformed CSVs available in S3 output folder

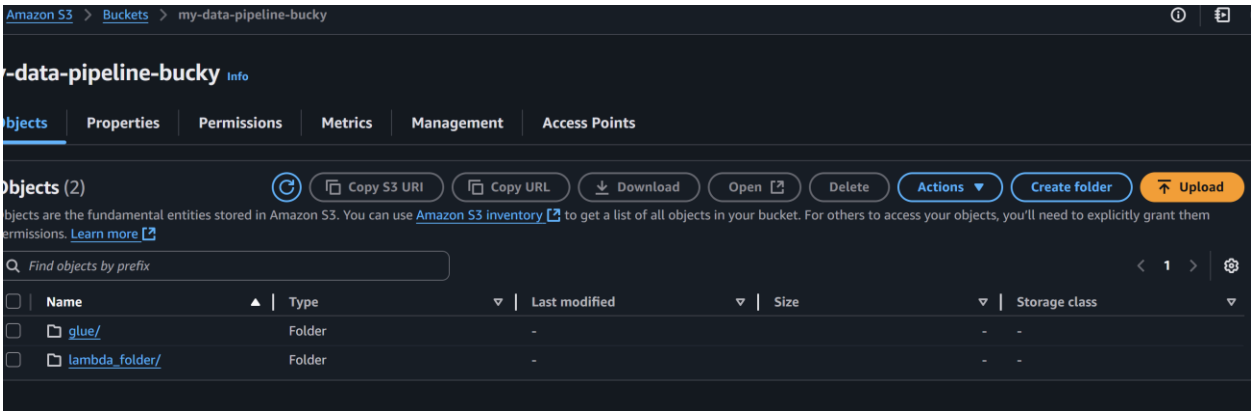
Next Steps (Optional Enhancements):

- Add timestamp or UUID to file names for uniqueness
- Trigger Glue via EventBridge automatically
- Integrate Athena or QuickSight for reporting and querying

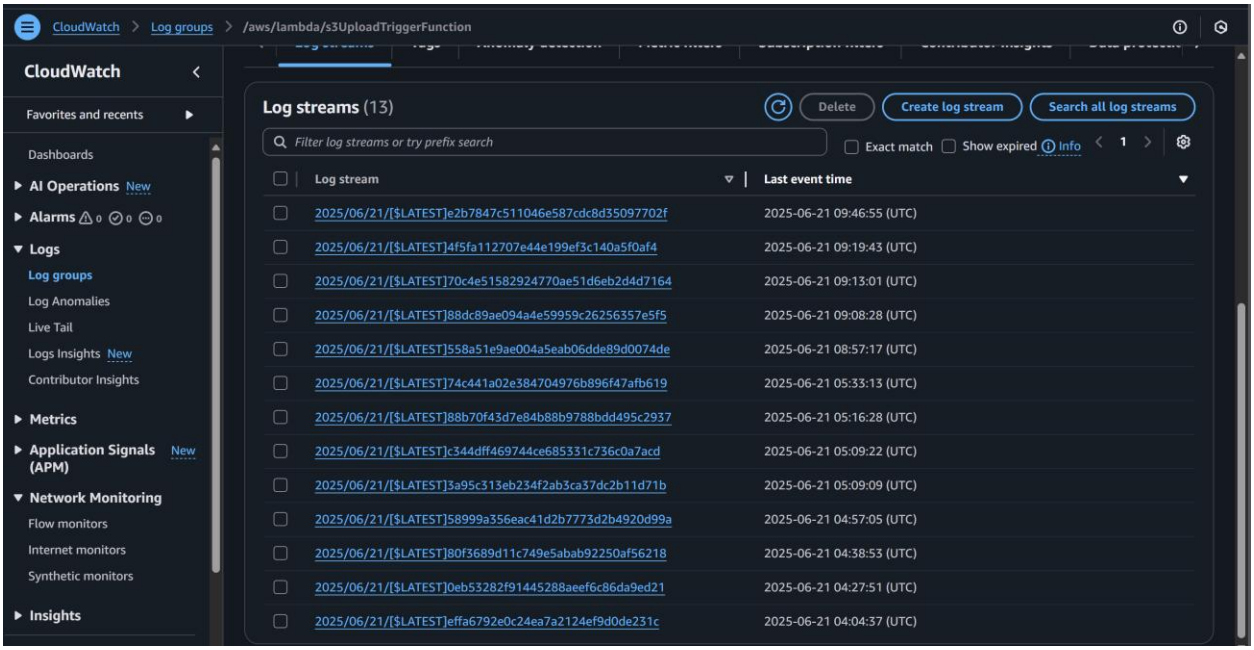
Workspace Output (Paste Your Results Below)

📝 Paste screenshots, logs, and outputs:

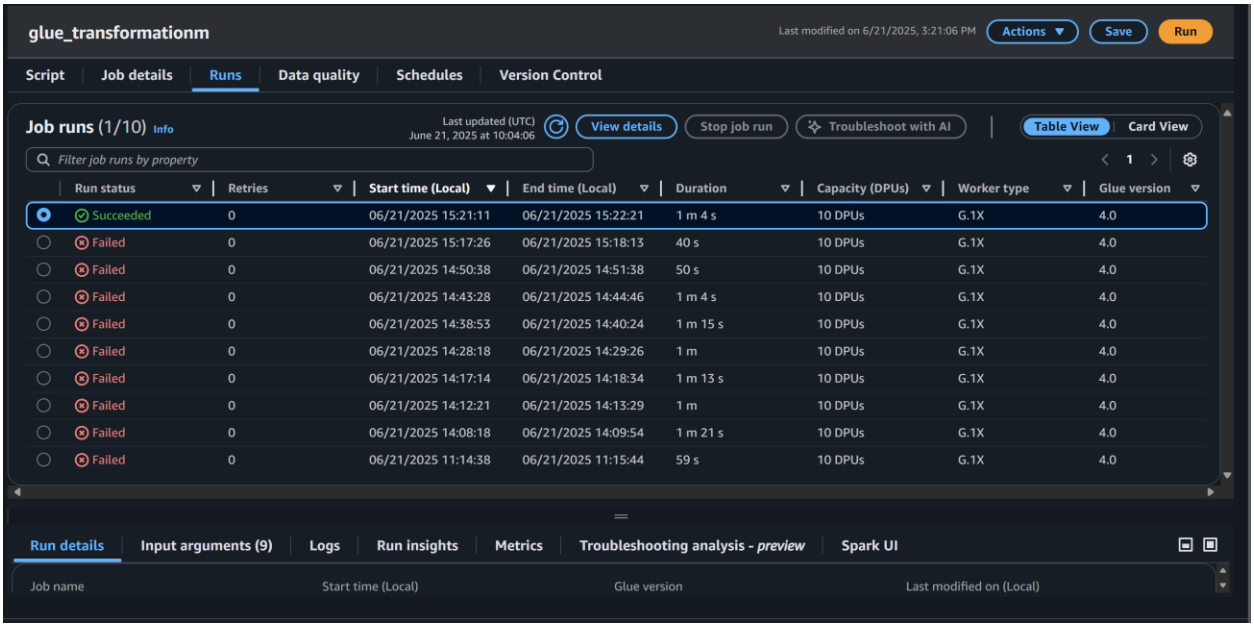
`[S3 structure screenshots]`



`[Lambda execution logs]`



[Glue job logs]



[Before & After CSV samples]

🧢 **Before (Raw CSV - uploaded to input_folder/)**

```
id,name,email,age
1,Alice,alice@example.com,28
2,Bob,bob@example.com,34
3,Charlie,charlie@example.com,30
2,Bob,bob@example.com,34
```

⚙️ **After Lambda (Cleaned CSV - stored in output_folder/)**

```
id,name,email,age
1,Alice,alice@example.com,28
2,Bob,bob@example.com,34
3,Charlie,charlie@example.com,30
```

👨‍💻 Author & Date:

Name: Atmakuru Siva Sandeep
Date: June 21, 2025