Project: Lakehouse Architecture for E-Commerce Transactions

Project Brief

You are tasked with designing and implementing a production-grade **Lakehouse architecture** for an e-commerce platform on **AWS**. The system will ingest raw transactional data stored in Amazon S3, clean and deduplicate it using **Delta Lake**, and expose it for downstream analytics through **Amazon Athena**.
You will orchestrate the entire process using **AWS Step Functions** and automate deployment with **CI/CD on GitHub Actions**.

The business expects high data reliability, schema enforcement, and consistent freshness of data for analytical workloads.

Core Services You Must Use

Service	Purpose
Amazon S3	Raw zone and processed zone storage
AWS Glue + Spark	Distributed ETL jobs, Delta Lake integration
Delta Lake	ACID-compliant table format on top of S3
AWS Step Functions	Orchestrate the ETL lifecycle
AWS Glue Data Catalog	Metadata layer for Athena + Glue interoperability
Amazon Athena	Query engine for downstream analytics
GitHub Actions	CI/CD pipeline for ETL scripts

Your Mission

Design and implement a system that:

1. Detects new data dropped into S3's raw zone.
2. Cleans and transforms the data using Glue + Delta Lake.
3. Writes the clean data into optimized Delta tables in the processed zone.
4. Updates Glue Data Catalog for querying through Athena.
5. Archives the original files after successful ingestion.
6. Automates orchestration with AWS Step Functions.
7. Implements CI/CD using GitHub Actions on the main branch.

You are expected to:

- Make architectural and schema design decisions.

- Simulate a real-world production scenario.
- Write modular, reusable Spark code.
- Justify your structure, validations, and partitioning logic.
- Enable reproducibility via GitHub Actions.

Data Sources

Your project works with 3 core datasets, all ingested from CSVs into an S3 **raw zone**:

Product Data

Fields:

- product_id
- department_id
- department
- product_name

Orders

Fields:

- order_num
- order_id
- user_id
- order_timestamp
- total_amount
- date

Order Items

Fields:

- id
- order_id
- user_id
- days_since_prior_order
- product_id
- add_to_cart_order
- reordered
- order_timestamp

- date

Expected Deliverables

Delta Lake Tables

Each dataset must be stored in Delta Lake format in a **lakehouse-dwh** zone in S3.
Expect:

- Deduplicated data
- Schema enforcement
- Partitioning for performance
- Merge/upsert logic

Validation Rules

Define and enforce rules such as:

- No null primary identifiers
- Valid timestamps
- Referential integrity
- Deduplication across files

Log rejected records.

Orchestration Requirements

Use **AWS Step Functions** to orchestrate the ETL pipeline. Your state machine should:

1. Detect a new file arrival in S3 (simulate trigger).
2. Run a Glue Job (with Delta Lake) for each dataset.
3. On success, archive files to /archived/.
4. On failure, log the error and send an alert (optional).
5. Optionally, run a Glue Crawler to update the Data Catalog.
6. Optionally, run an Athena query to validate data presence.

Incorporate failure handling, timeouts, and branching logic.

CI/CD Expectations

Use **GitHub Actions** to automate:

- CI of Spark job
- Unit / integration tests
- Optional: Deploy Step Function definition as JSON/YAML

Triggers must be scoped to the main branch.