**ETL Pipeline Project README**

**Introduction**

Welcome to the ETL Pipeline Project README. This document provides an overview of setting up, running, and understanding the ETL pipeline designed for transforming E-commerce data.

**Setup Instructions**

**SQL Database Setup**

Install PostgreSQL: If PostgreSQL is not installed, download and install it from PostgreSQL official website.

**Create Database and User:**

Open a terminal and execute the following commands:

CREATE DATABASE ecommerce\_db;

CREATE USER ecommerce\_user WITH PASSWORD 'your\_password';

GRANT ALL PRIVILEGES ON DATABASE ecommerce\_db TO ecommerce\_user;

**Database Connection Details:**

Host: localhost

Port: 5432

Username: ecommerce\_user

Password: your\_password

NoSQL Database Setup (MongoDB)

Install MongoDB: Install MongoDB by following the instructions on the MongoDB installation guide.

Start MongoDB Service: Start MongoDB service on your system.

Default configurations should work for local development.

**Running the Pipeline**

Using Airflow for Orchestration

Install Python Dependencies:

Navigate to your project directory and install required Python libraries:

pip install -r requirements.txt

Start Airflow Scheduler and Webserver:

Open two terminals and start Airflow components:

bash

Copy code

airflow scheduler &

airflow webserver &

**Access Airflow UI:**

Open a web browser and go to <http://localhost:8080.>

Navigate to the 'ecommerce\_etl\_pipeline' DAG and trigger it to start the pipeline.

Data Structures and Time Complexities

**Data Models Used**

**SQL Database Tables:**

**customers**: Stores customer information.

**orders**: Contains order details including customer references.

**order\_items**: Lists items in each order with product references.

**products**: Describes products available with category references.

**categories**: Categorizes products.

**reviews**: Holds product reviews by customers.

**NoSQL Database Collection:**

**aggregated\_data** : Stores aggregated customer metrics and insights.

**Time Complexities (Big O Notations)**

SQL Operations:

Joins: O(m \* n) where m and n are the number of rows in joined tables.

Aggregations: O(n log n) for group by operations.

Retrievals: O(n log n) for fetching top N records.

NoSQL Operations:

Insertions: O(1) for inserting documents.

Reads: O(1) for retrieving documents by indexed fields.

Challenges Faced and Solutions

Challenges

**Data Consistency Issues:**

Discrepancies between SQL and NoSQL databases due to sync delays.

Inconsistencies in product data across systems.

**Performance Bottlenecks:**

High query times during peak load periods.

Heavy resource consumption during data transformations.

Solutions Implemented

**Data Reconciliation Scripts:**

Automated scripts to reconcile data between SQL and NoSQL databases.

Scheduled jobs to ensure regular updates and consistency checks.

**Performance Optimization:**

Database indexing for faster query execution.