1. **How will you read messages from the queue?**
   * I will read messages from the AWS SQS queue using the boto3 library. The consume\_messages function fetches messages in batches of up to 10 and processes them. It continues to fetch and process messages until the queue is empty.
2. **What type of data structures should be used?**
   * For this project, I used dictionaries to represent JSON objects and lists to collect processed records before inserting them into the PostgreSQL database.
3. **How will you mask the PII data so that duplicate values can be identified?**
   * PII data such as device\_id and ip are masked using a SHA-256 hash function. This ensures that the same input value always produces the same hashed output, allowing duplicate values to be identified without revealing the original data.
4. **What will be your strategy for connecting and writing to Postgres?**
   * The connection to the PostgreSQL database is established using psycopg2. The write\_to\_postgres function inserts processed records into the user\_logins table. Each record is inserted using a parameterized query to prevent SQL injection attacks.
5. **Where and how will your application run?**
   * The application is designed to run locally using Docker containers. The docker-compose.yml file sets up the necessary services, including LocalStack for simulating AWS services and a PostgreSQL database. The main script can be executed from the host machine, which will interact with the services running in the containers.

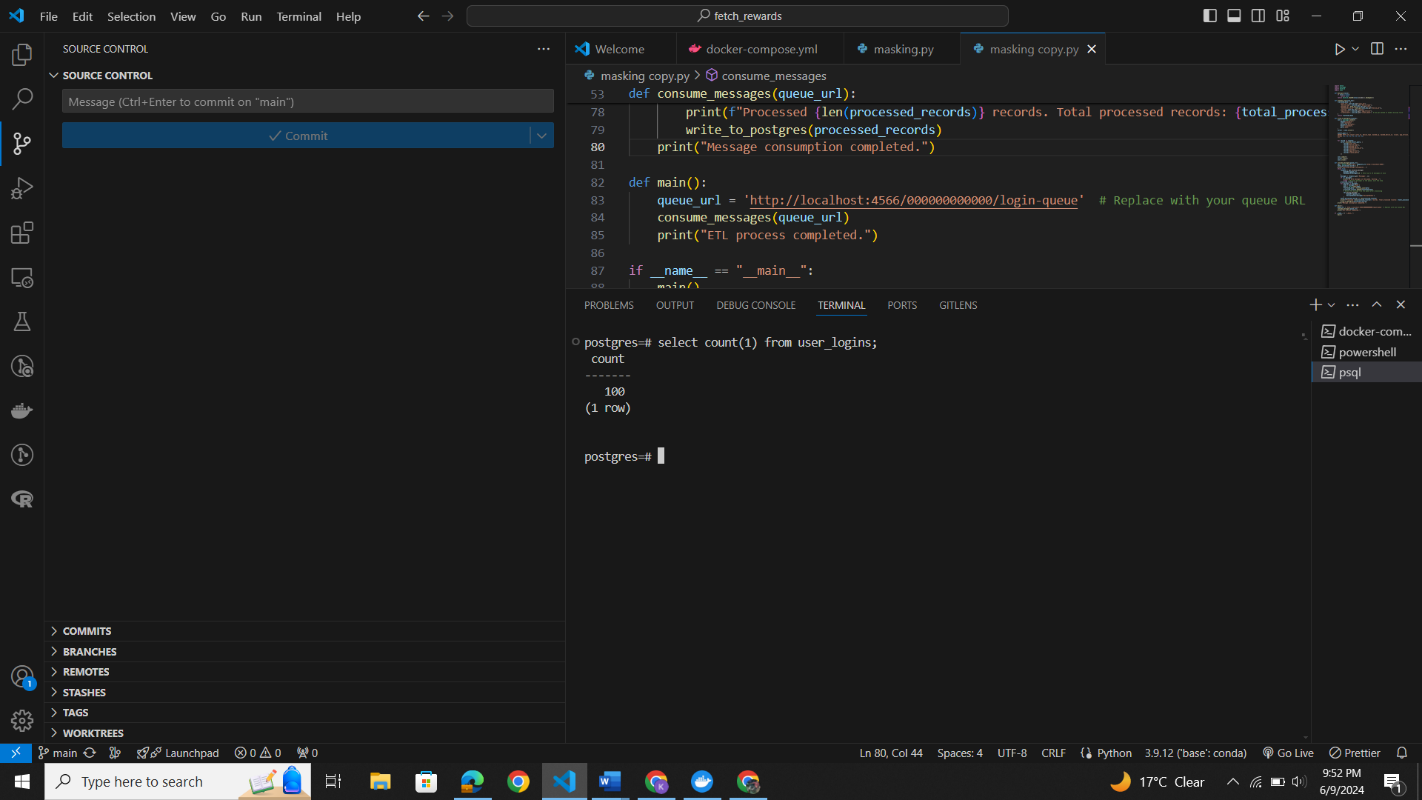
**Output Validation:-**

**Code Execution:-**

**A computer screen shot of a computer screen

Description automatically generated**

**\* All records(100) loaded into Database**

****

**A screenshot of a computer

Description automatically generatedsample data after masking in Database:-**

**A screenshot of a computer

Description automatically generated**