**TECHNICAL REPORT ON THE ETL PIPELINE FOR ELECTRONIC PRODUCTS REVIEWS ON ALIEXPRESS**

**INTRODUCTION**

This ETL (Extract, Transform, Load) pipeline is designed to scrape reviews from Trustpilot for the electronic products from AliExpress website, transform the scraped data, and load it into a Kafka topic and an AWS S3 bucket. The pipeline is implemented in Python and uses several libraries that, including Selenium, BeautifulSoup, Kafka, and Boto3.

**COMPONENTS AND TECHNOLOGIES**

1. **Web Scraping**:
   * **Selenium**: Automates browser actions to scrape data.
   * **BeautifulSoup**: Parses HTML content to extract relevant data.
2. **Data Transformation**:
   * **Pandas**: Processes and cleans data.
3. **Data Loading**:
   * **Kafka**: Publishes data to a Kafka topic.
   * **AWS S3**: Stores the processed data in a cloud storage bucket.
   * **AWS Redshift**: To published database to the data scientist and others to have access for analysis
4. **Configuration Management**:
   * **ConfigParser**: Manages configuration keys and other confidential details.
5. **Logging**:
   * **Logging**: Provides logging capabilities for monitoring and debugging.

**WORKFLOW**

**Step 1: Data Extraction**

* **Initialization**:
  + Configure logging and initialize the Chrome WebDriver in headless mode for scraping.
* **Scraping Logic**:
  + Loop through more than 2300 pages of reviews on Trustpilot.
  + For each page, extract user details, location, number of reviews, date, rating, and review content using Selenium and BeautifulSoup.
  + Store the extracted data in lists.

**Step 2: Data Transformation**

* **Processing**:
  + Convert lists into a pandas DataFrame.
  + Clean and format the 'Date\_of\_Experience' column.
  + Save the DataFrame to a CSV file.

**Step 3: Data Loading**

* **Kafka Producer**:
  + Initialize a Kafka producer to publish the data.
  + Convert each row of the DataFrame to a comma-separated string and send it to a Kafka topic.
* **AWS S3 Bucket**:
  + Use Boto3 to create an S3 bucket if it doesn't already exist.
  + Upload the CSV file to the S3 bucket.
* **Redshift Database**:
  + Connect to Redshift using AWS credentials.
  + Create a table schema in Redshift.
  + Copy data from the S3 bucket to the Redshift table.

#### ****DETAILED CODE EXPLANATION****

**Web Scraping with Selenium and BeautifulSoup**

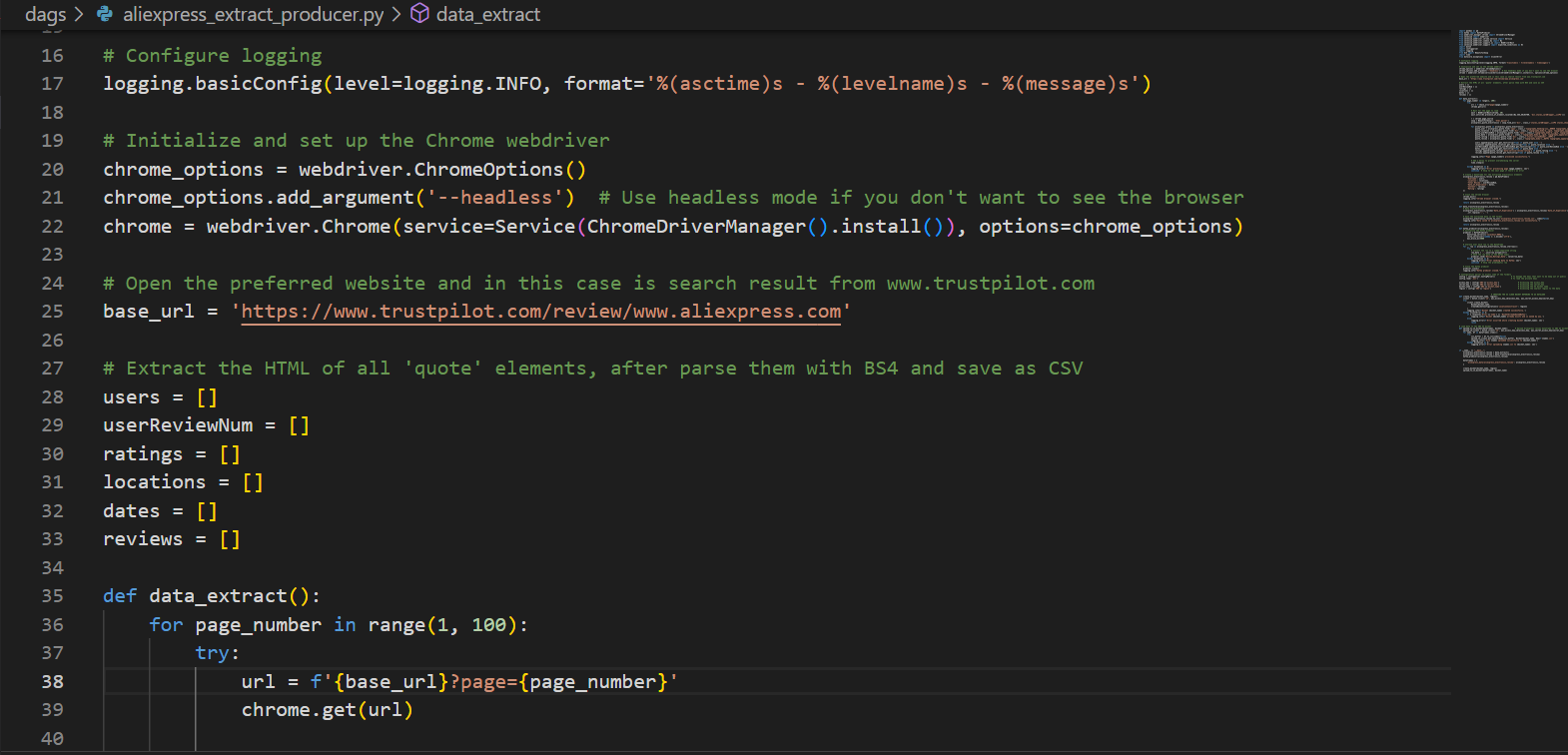


Figure 1: Snippet code of webscraping with selenium and BeautifulSoup

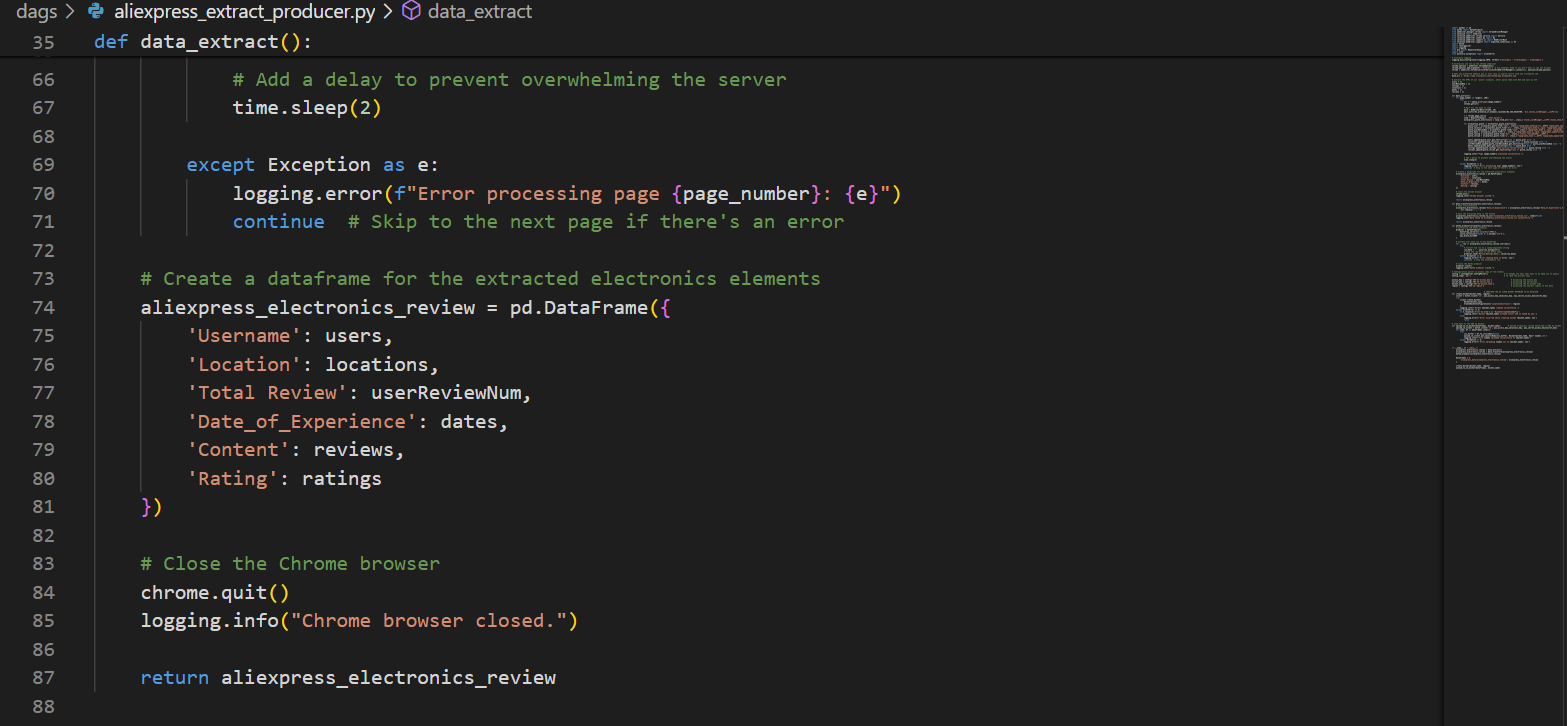


Figure 2: Extraction phase that create columns for the data

Data Transformation with Pandas

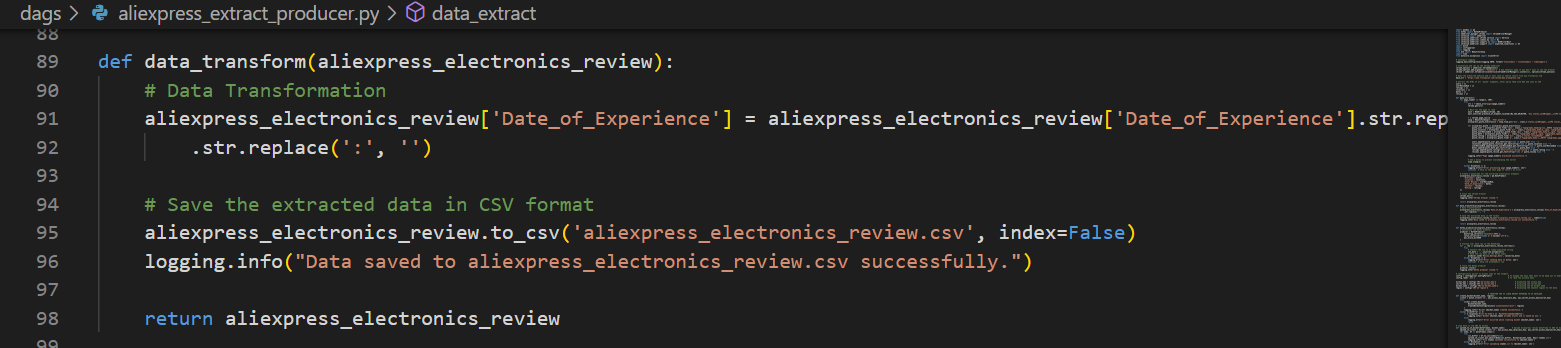


Figure 3: Transformation and cleaning data to fit the columns attributes

Kafka Producer and Kafka Consumer

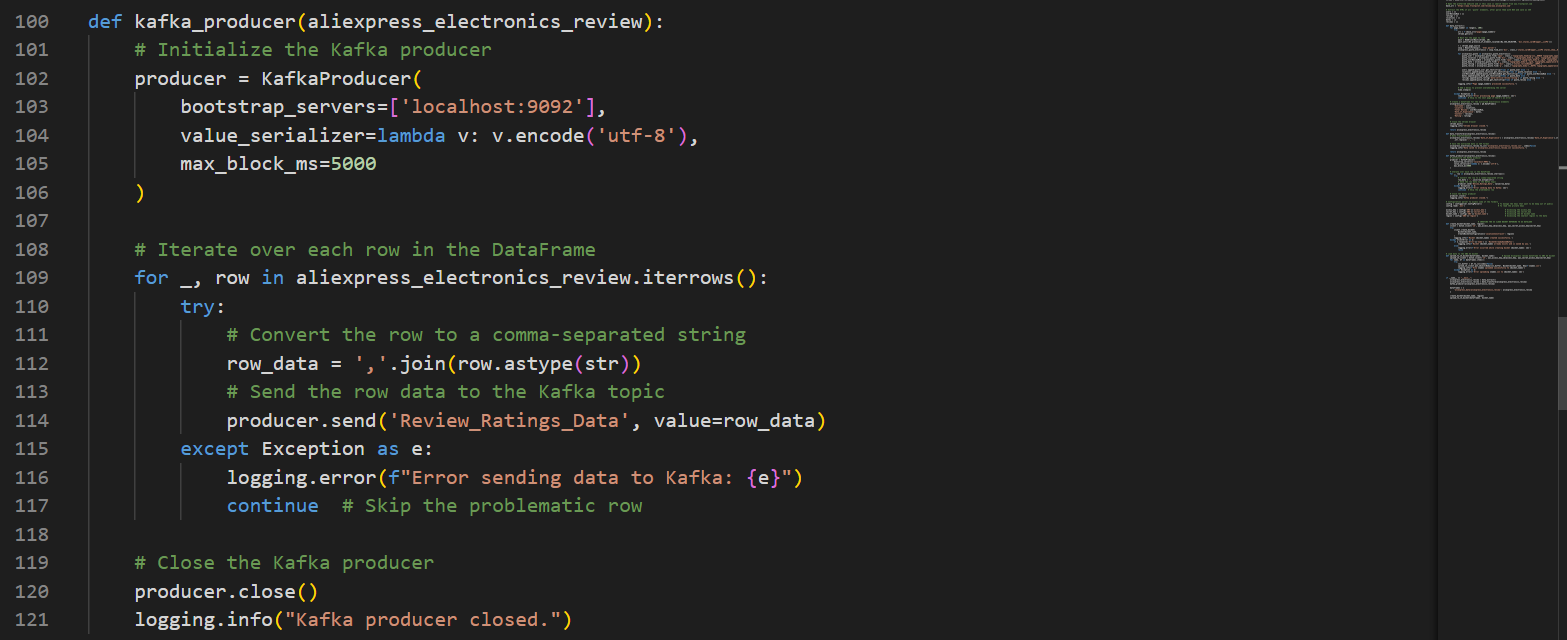


Figure 4: Snippet code for Kafka producer and sending the data to AWS bucket

AWS S3 Bucket Management

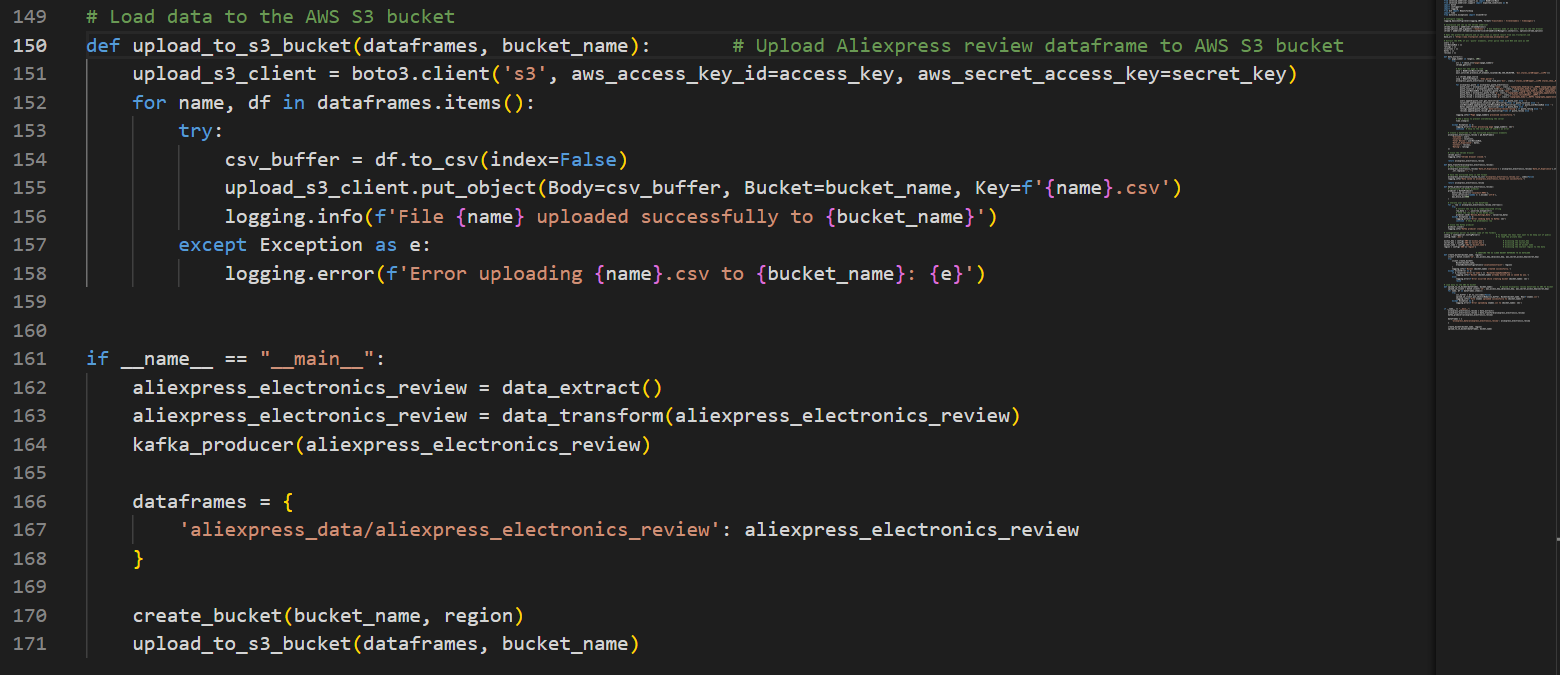
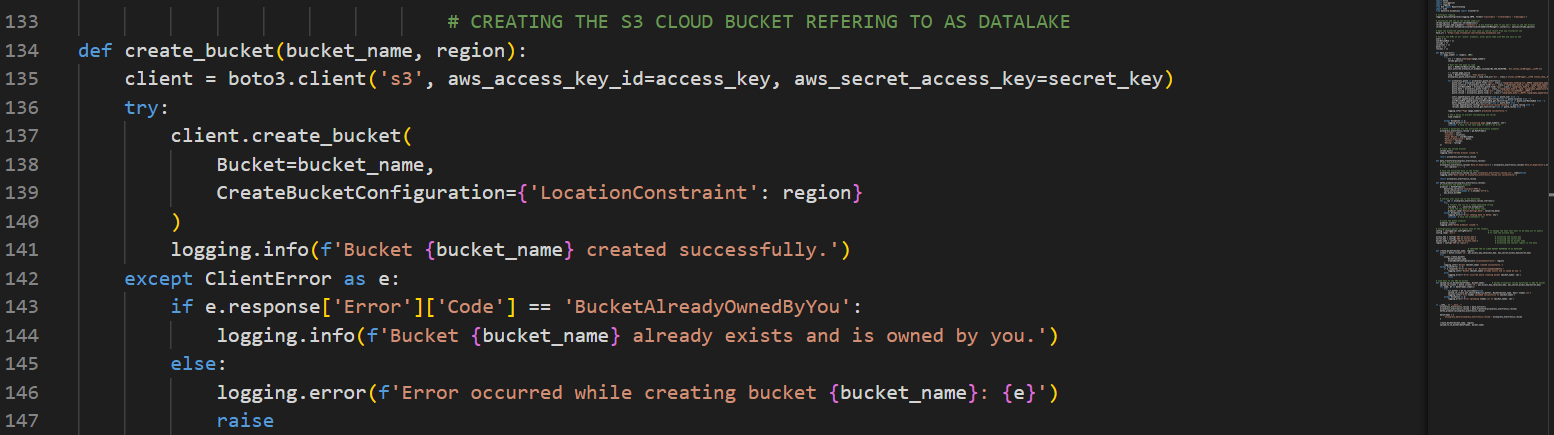


Figure 5: AWS bucket creation and upload the data extracted

Redshift Database Loading

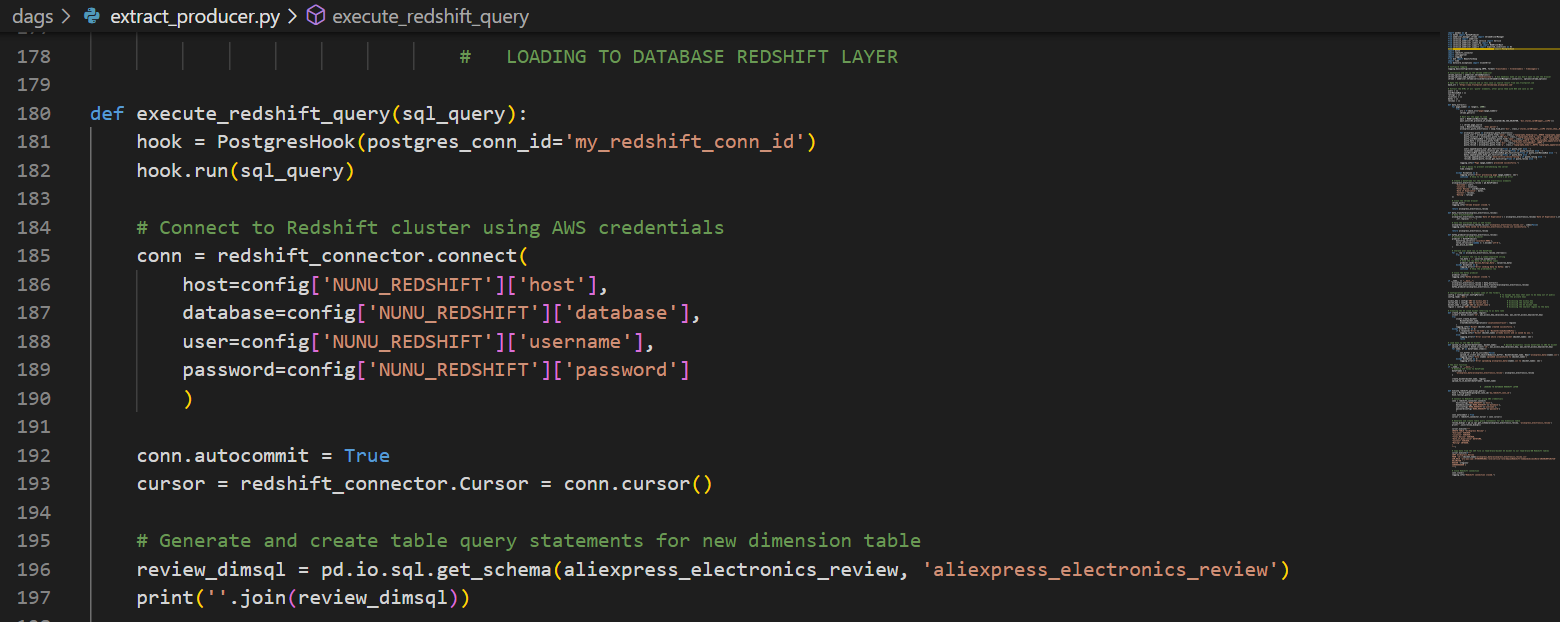


Figure 6: Trafficking data from AWS bucket to AWS redshift for analysis

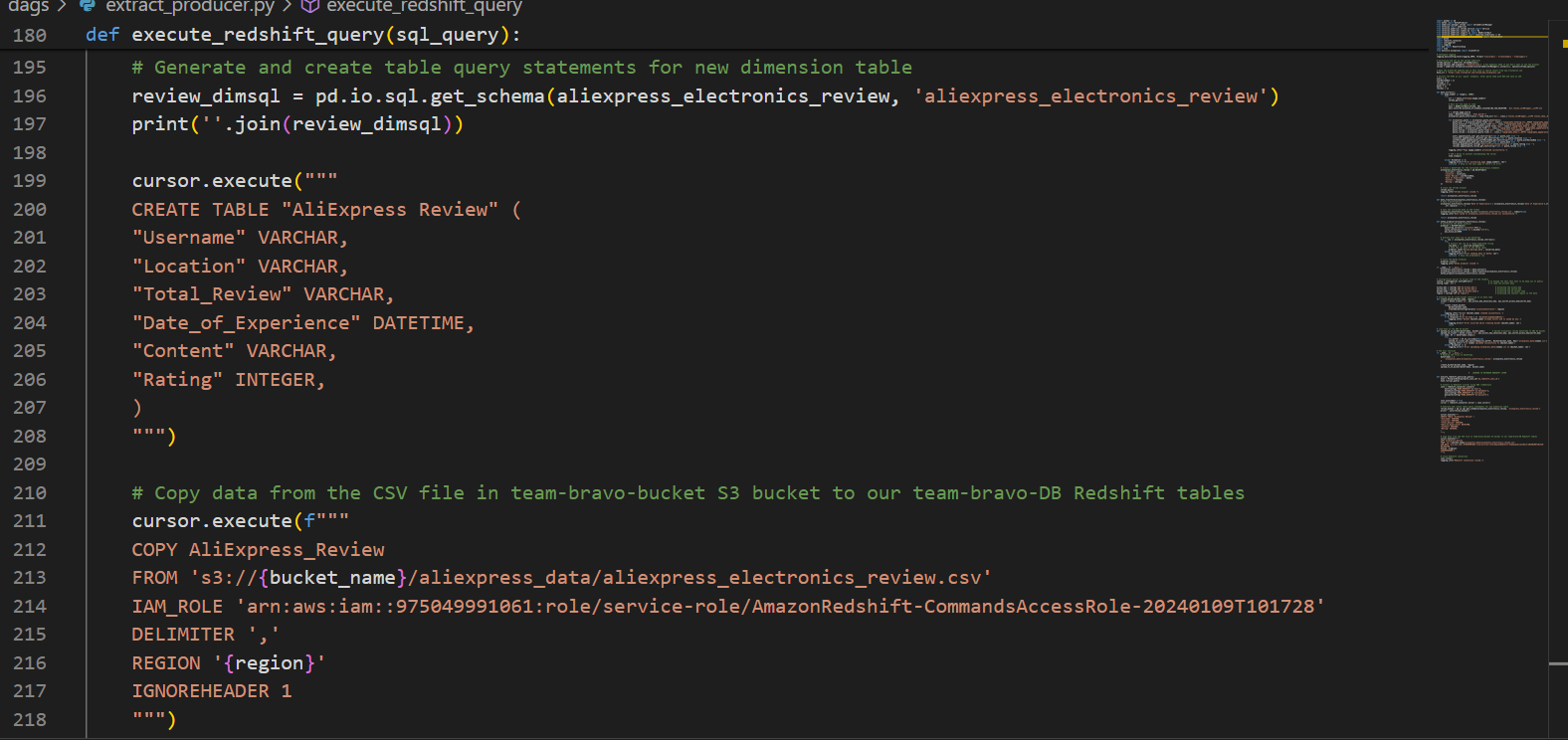


Figure 7: Details for database created and loaded to the AWS Redshift

**CONFIGURATION MANAGEMENT**

* **ConfigParser** reads sensitive keys from a .env file:

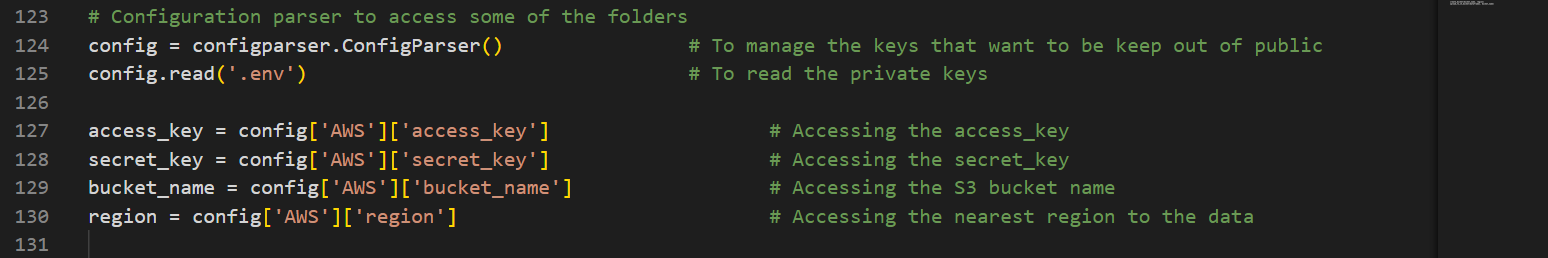


Figure 8: Accessing the confidential details from. Env

**APACHE AIRFLOW**

**DAG Definition**

The ETL process is defined in an Airflow DAG (aliexpress\_review\_dags.py). The DAG orchestrates the execution of the ETL tasks.

**Tasks**

* **Extract Data Task**: Runs the *data\_extract* function to extract reviews.
* **Transform Data Task**: Runs the *data\_transform* function to clean and format the data.
* **Create S3 Bucket Task**: Runs the *create\_bucket* function to ensure the S3 bucket exists.
* **Upload to S3 Task**: Runs the *upload\_to\_s3\_bucket* function to upload the transformed data to S3.
* **Kafka Producer Task**: Runs the *kafka\_producer* function to publish the data to a Kafka topic.

**Task Dependencies**

The tasks are set to execute in a specific order using the >> operator:

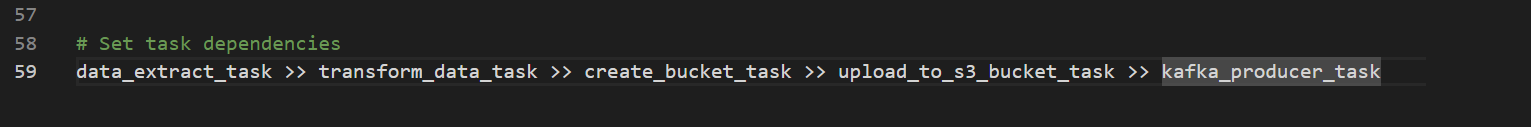


Figure 9: Orderly list of task dependencies

**EXECUTION FLOW**

1. **Extract** data from Trustpilot using Selenium and BeautifulSoup.
2. **Transform** the data using Pandas and save it to a CSV file.
3. **Load** the transformed data to a Kafka topic and an AWS S3 bucket.
4. **Log** the process for monitoring and debugging.

**MONITOR AND DATA TRACK ACTIVITIES**

Using Kafka confluent control center GUI to monitor and track the data activities from extraction to uploading to the AWS bucket. Identify a possible fault and clearly shows the speed, latency request, stability of the network, data disk level of consumption and topics created as messages extracted.

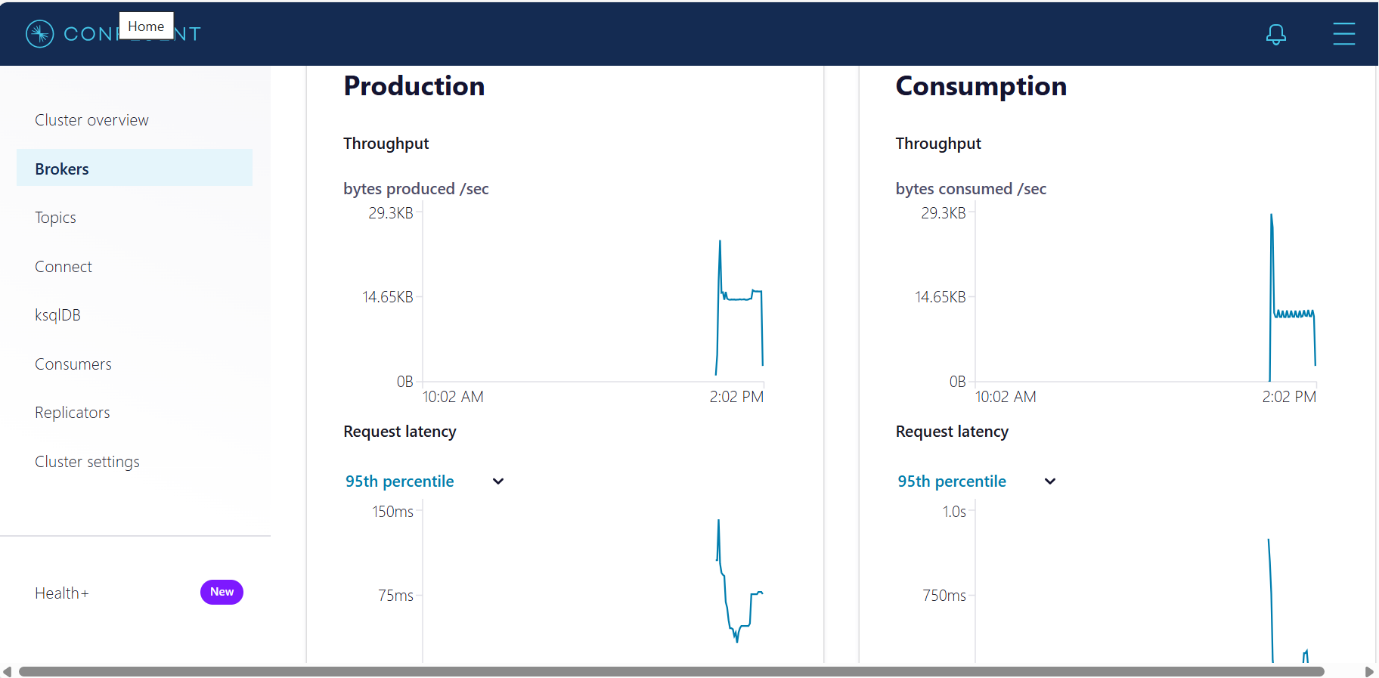


Figure 10: The GUI of Kafka Confluent Control Center

**CONCLUSION**

This ETL pipeline, implemented using Apache Airflow, efficiently extracts, transforms, and loads data from Trustpilot into Kafka, AWS S3, and Redshift. It leverages a variety of tools and libraries to ensure data is processed and stored reliably, making it a robust solution for data ingestion, storage, and analysis needs. By automating the process with Airflow, the pipeline ensures timely and consistent data updates, enhancing the overall data management workflow for Team Bravo.