Data Pipeline and API Challenge

A simple data pipeline that ingests data from an SFTP server, processes it, and exposes the data through a REST API with filtering and pagination capabilities. Code for ingesting data are provided and tested.However for the pipeline to work easily ,I used a sample file from Kaggle (https://www.kaggle.com/datasets/shohinurpervezshohan/techcorner-mobile-purchase-and-engagement-data) ,proceesed it,saved it to a sqlite db and exposed it through the API.This is all done locally and can be reproduced.I really tried my best with the limited time and resources to come up with something close to the reuirements. :)

Overview

This project implements:

1. Data ingestion from an SFTP source
2. Data processing and cleaning
3. Storage in a database
4. REST API with filtering and pagination

Project Structure

```
data-pipeline/
├── README.md                # Project documentation
├── requirements.txt         # Python dependencies
├── ingest.py                # SFTP data ingestion
├── process.py               # Data processing logic
├── database.py              # Database operations
├── api.py                   # FastAPI implementation
├── main.py                  # Main entry point
└── sample_data/             # Sample data for testing
    ├── TechCorner_Sales_update.csv
```

Setup Instructions

1. Prerequisites

- Python 3.8 or higher
- Access to an SFTP server (or use direct file processing for testing)

2. Installation

Clone the repository and install dependencies:

```
git clone https://github.com/yourusername/data-pipeline.git
cd data-pipeline
pip install -r requirements.txt
```

3. Configuration

Configuration is managed through environment variables or command-line arguments:

```
# SFTP Configuration
export SFTP_HOST=localhost
export SFTP_PORT=22
export SFTP_USERNAME=user
export SFTP_PASSWORD=password
export SFTP_REMOTE_DIR=/data

# Database Configuration
export DATABASE_URL=sqlite:///data_pipeline.db
```

4. Download the Dataset

Download the TechCorner dataset from Kaggle (https://www.kaggle.com/datasets/shohinurpervezshohan/techcorner-mobile-purchase-and-engagement-data) and place the CSV file in your project directory.

5. Process the Data

For testing purposes, you can directly process the CSV file without an SFTP server:

```
# process_local.py
from process import process_file
from database import initialize_database, store_dataframe

# Initialize database
initialize_database()

# Process file directly
file_path = "./sample/TechCorner_Sales_update.csv"
processed_data = process_file(file_path)

if processed_data is not None:
    # Store in database
    success = store_dataframe(processed_data)
    print(f"Processed {len(processed_data)} rows from {file_path}")
else:
    print("Failed to process file")
```

Run this script to populate the database:

```
python process_local.py
```

6. Running the API Server

Start the API server:

```
uvicorn api:app --reload --host 0.0.0.0 --port 8000
```

The API will be available at http://localhost:8000

You can access the interactive API documentation at http://localhost:8000/docs

API Documentation

Authentication

All API requests require an API key in the `X-API-Key` header:

```
X-API-Key: test_api_key
```

For testing, the following API keys are accepted:

- `test_api_key`
- `demo_key`

Endpoints

GET /data

Retrieve processed data with date filtering and cursor-based pagination.

Query Parameters:

- `start_date` (optional): Filter data from this date (format: YYYY-MM-DD)
- `end_date` (optional): Filter data until this date (format: YYYY-MM-DD)
- `cursor` (optional): Pagination cursor for retrieving the next set of results
- `limit` (optional): Number of records to return (default: 50, max: 100)

Example Request:

```
GET /data?start_date=2022-01-01&end_date=2022-12-31&location=New%20York&gender=Male&min_age=25&max_age=35&mobile_name=iPhone&limit=1(
Host: localhost:8000
X-API-Key: test_api_key
```

Example Response:


```
{
  "items": [
    {
      "id": 1,
      "customer_id": 10245,
      "date": "2022-01-15T00:00:00",
      "customer_location": "New York",
      "age": 28,
      "gender": "Male",
      "mobile_name": "iPhone 13 Pro",
      "sell_price": 999.99,
      "from_facebook": "Yes",
      "followed_page": "Yes",
      "previous_purchase": "No",
      "heard_of_shop": "Yes",
      "source_file": "TechCorner_Sales_update.csv",
      "processed_at": "2023-01-15T12:30:45"
    },
    // More items...
  ],
  "next_cursor": "10",
  "total_count": 250
}
```


GET /health

Health check endpoint.

Example Response:


```
{
  "status": "healthy",
  "timestamp": "2023-01-15T14:30:45.123456"
}
```


Assumptions

1. **SFTP Server Configuration**: The project assumes basic SFTP authentication with username/password.
2. **Data Source**: This implementation uses the TechCorner_Sales_update.csv dataset from Kaggle (linked above).
3. **Database**: For simplicity, SQLite is used by default, but it can be replaced with any SQL database.
4. **Security**: For a production environment, additional security measures would be implemented.

Sample Ouput

For visual reference, sample API outputs are available in the project repository:

Future Improvements

1. Add support for incremental data loading
2. Implement data validation rules
3. Add support for schema evolution
4. Implement more sophisticated error recovery
5. Add unit and integration tests