FastAPI File Management using PostgreSQL, MinIO, and Caching

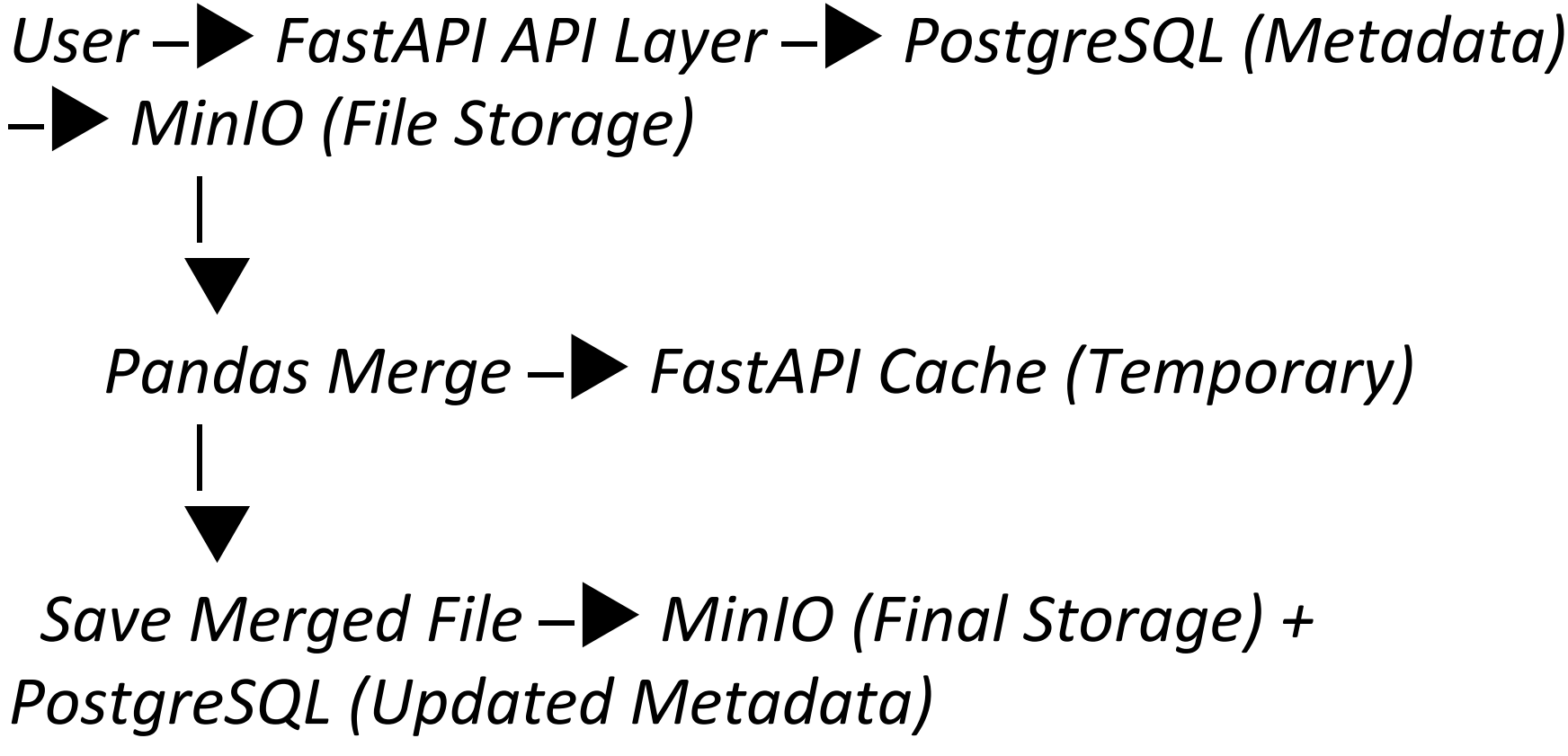
By: Shrabana Paul
Mentor: Suprava Das
Institution: IDEAS-TIH, ISI Kolkata
Internship Duration: Aug–Oct 2025

Abstract

- *Developed a scalable backend File Management System using FastAPI.*
- *Integrated PostgreSQL for metadata and MinIO for object storage.*
- *Implemented caching with FastAPI-Cache2 for performance.*
- *Supported real-time merging and retrieval of datasets.*
- *Ensures automation, reliability, and modular design.*

System Architecture / Workflow

FASTAPI FILE MANAGEMENT SYSTEM



Introduction

- *Organizations face challenges in handling large datasets.*
- *FastAPI + PostgreSQL + MinIO + Caching = Efficient backend for data ingestion.*
- *Simplifies ETL workflows with modular, microservice-based design.*

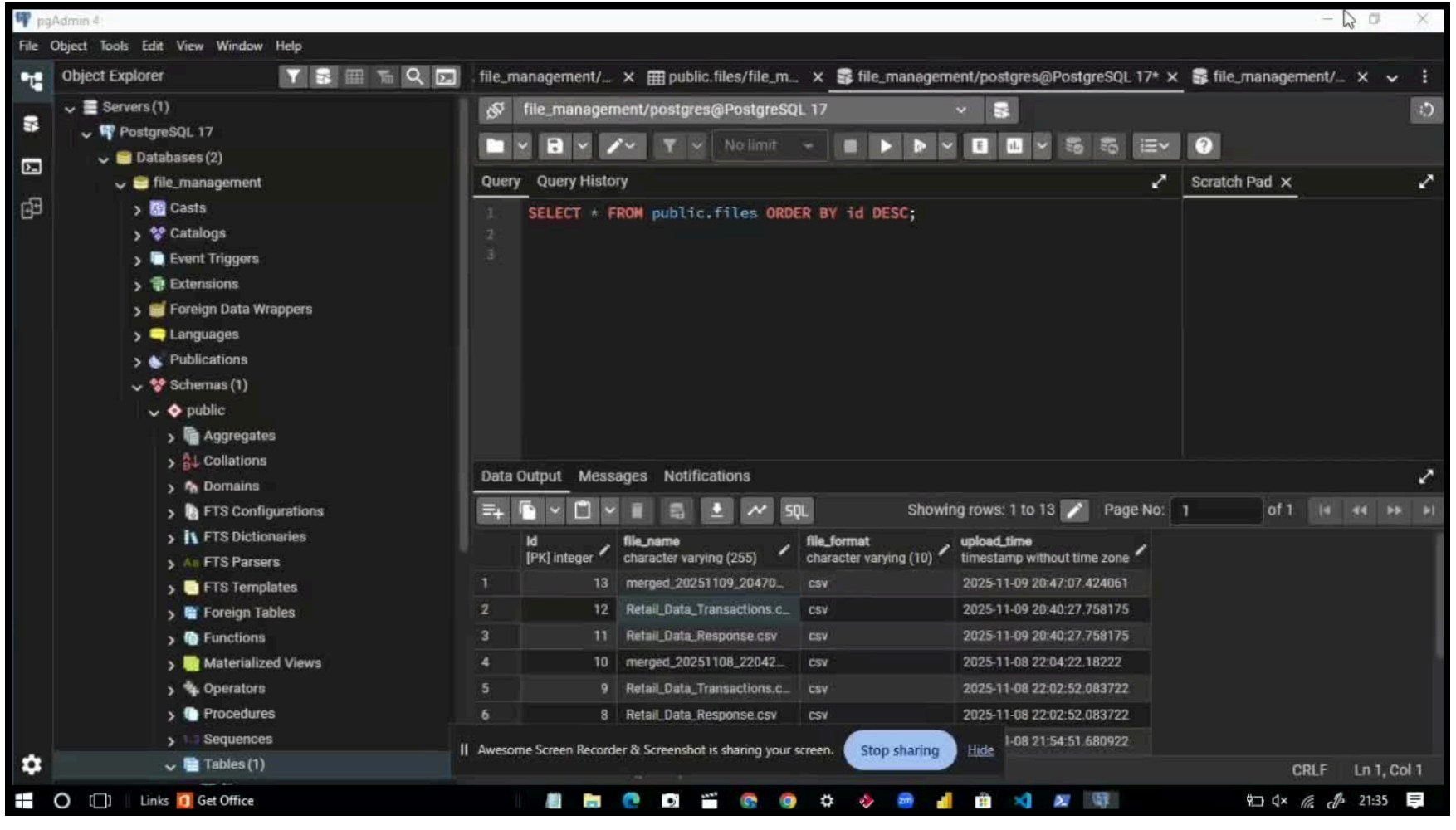
Project Objectives

1. *Develop API-driven file management system.*
2. *Integrate FastAPI, PostgreSQL, and MinIO.*
3. *Implement caching for merged datasets.*
4. *Automate data merging & validation.*
5. *Ensure scalability & reproducibility.*

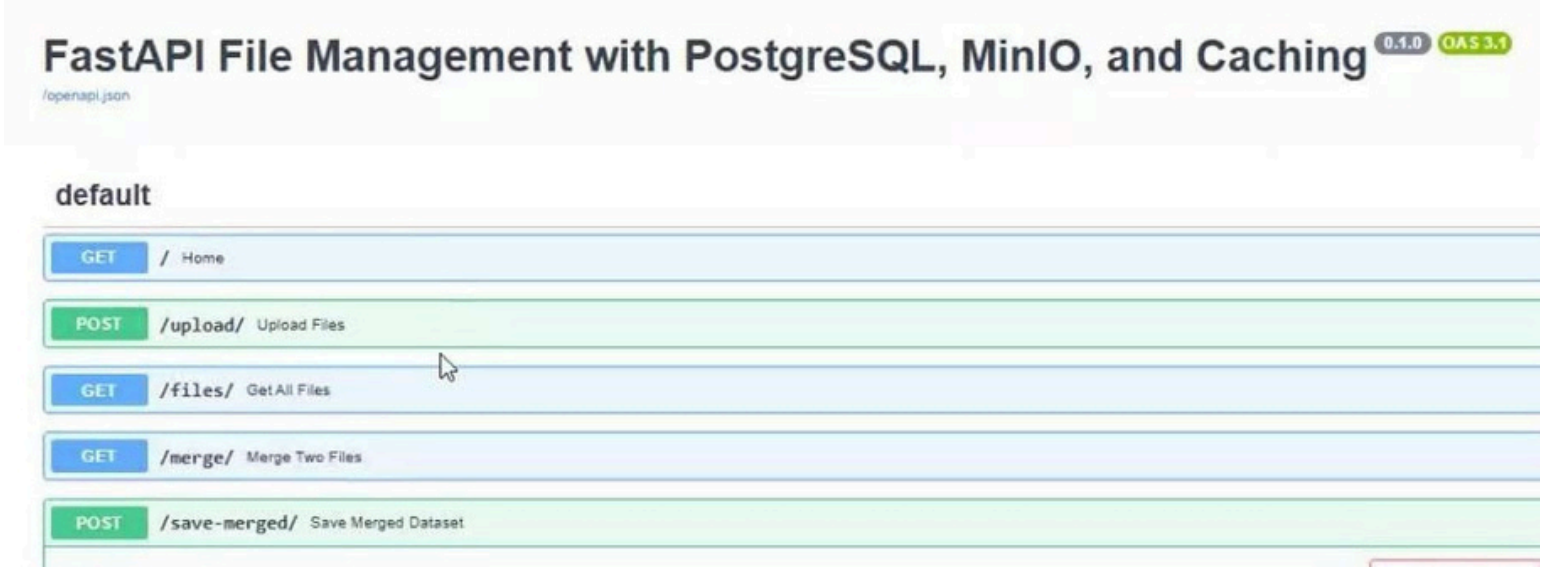
Methodology and tools used

Tool	Purpose
FastAPI	Backend framework
PostgreSQL	Metadata storage
MinIO	Object storage
Pandas	Data merging
SQLAlchemy	ORM for DB operations
FastAPI-Cache2	Temporary caching

Execution of the program

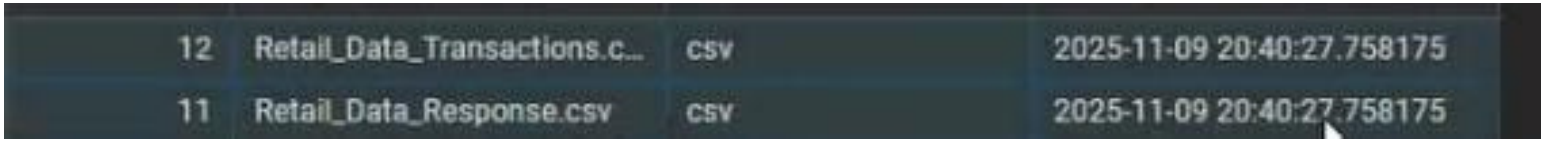


API Testing Environment



Swagger UI showing all FastAPI endpoints.

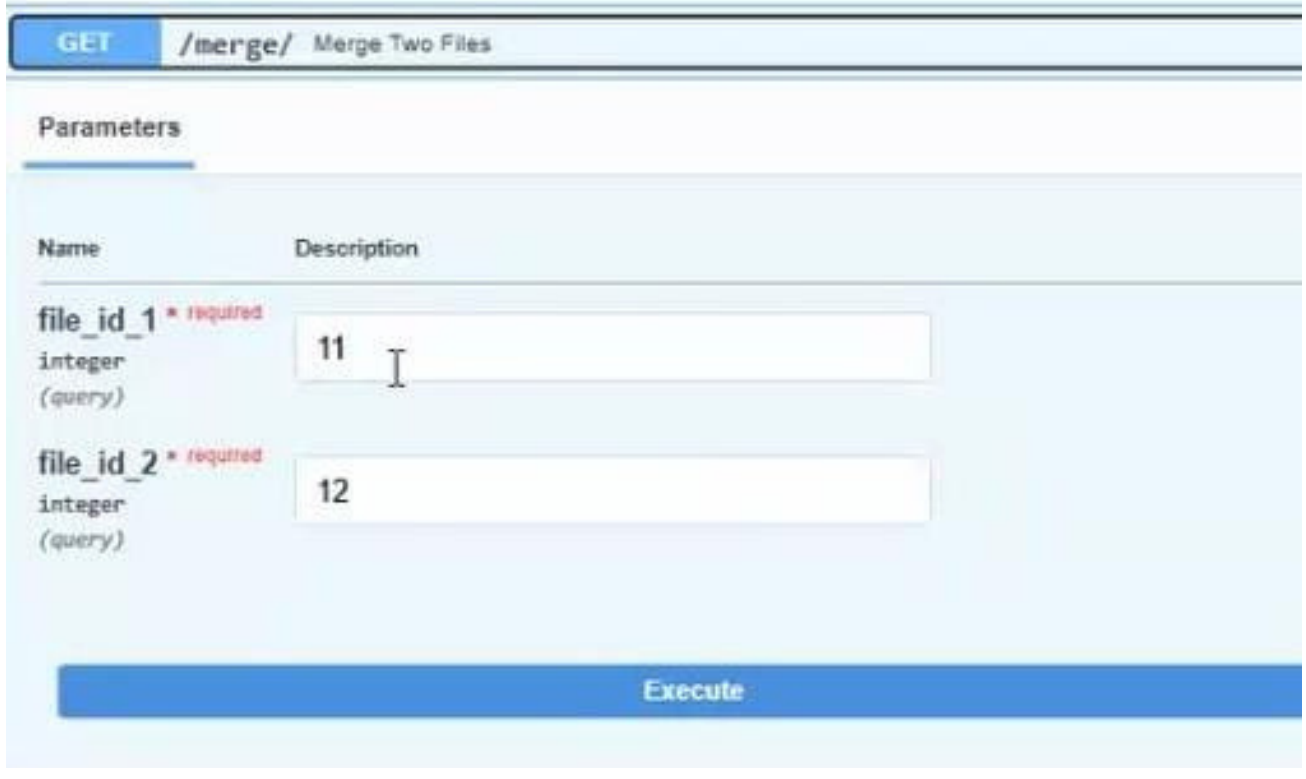
File Upload Verification



12	Retail_Data_Transactions.c...	csv	2025-11-09 20:40:27.758175
11	Retail_Data_Response.csv	csv	2025-11-09 20:40:27.758175

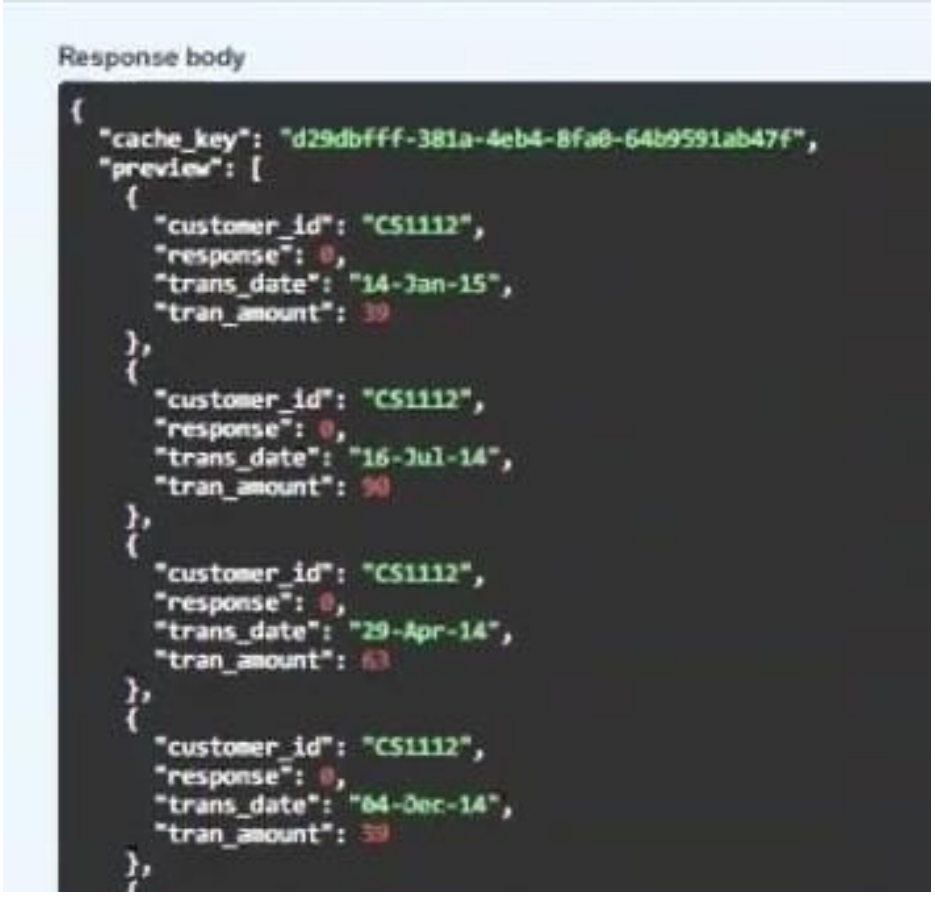
Uploaded files stored with metadata in PostgreSQL.

Merge Endpoint



File IDs 11 and 12 selected for merging.

Cached Merge Preview

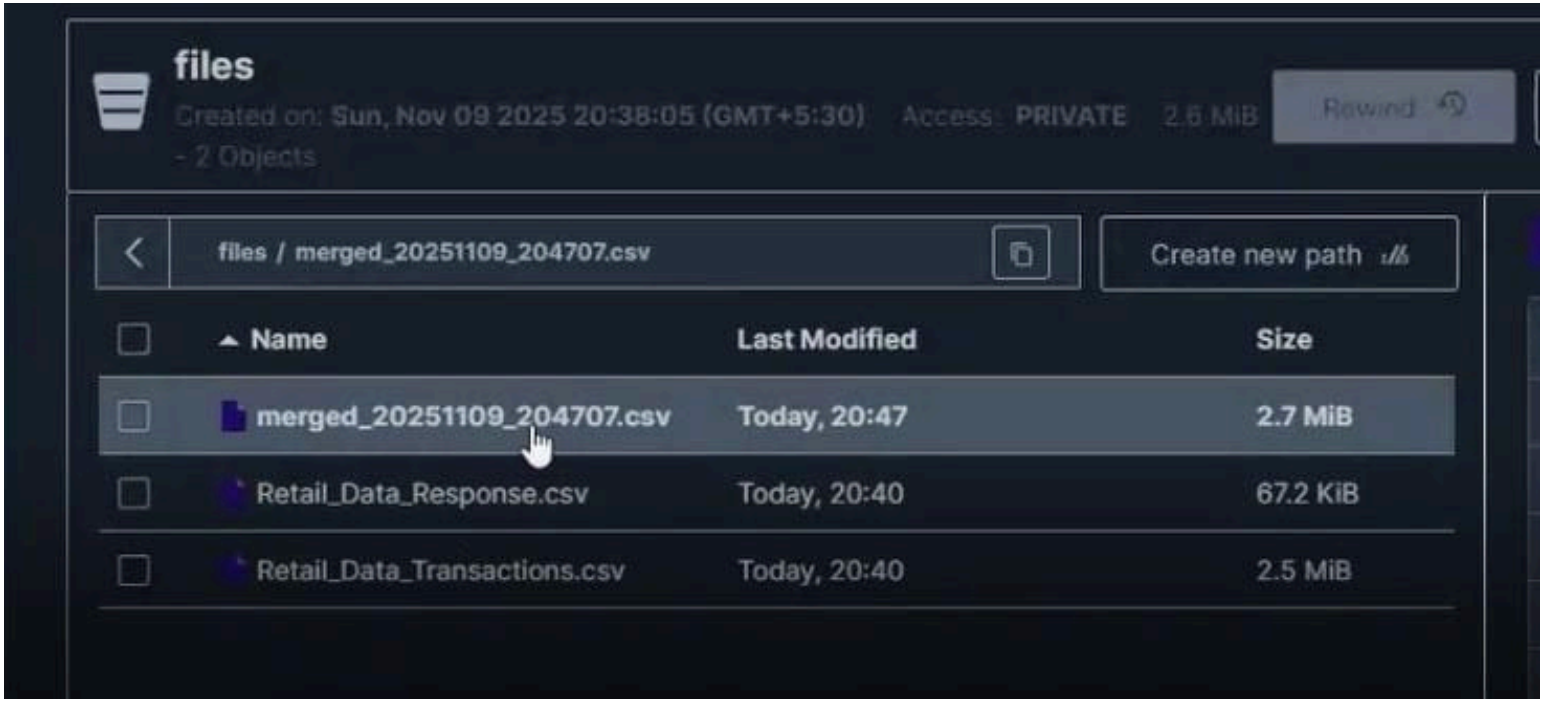


FastAPI returns preview of merged cached dataset.

Save Merged Dataset

Merged dataset saved successfully to MinIO.

Final Files in MinIO



MinIO storage shows merged and raw datasets.

Resullts

- All FastAPI endpoints (/upload/, /files/, /merge/, /save-merged/) executed successfully through Swagger UI.
- Uploaded datasets were stored correctly in MinIO and their metadata (ID, filename, format, timestamp) was inserted into PostgreSQL.
- Merge operation (file_id_1 = 11, file_id_2 = 12) executed smoothly:
  - Files fetched from MinIO
  - Merged via Pandas on `customer_id`
  - Preview returned with a unique cache_key
- Cached merged dataset was successfully saved back to MinIO as: merged_20251109_204707.csv
- PostgreSQL updated with new metadata entry for the merged file.
- Full backend workflow validated end-to-end: Upload → Retrieve → Merge (Cached) → Save → Verify

References & GitHub

- *• FastAPI Docs — https://fastapi.tiangolo.com*
- *• MinIO Docs — https://min.io/docs*
- *• PostgreSQL Docs — https://www.postgresql.org/docs*
- *• SQLAlchemy Docs — https://docs.sqlalchemy.org*
- *• GitHub Repo — https://github.com/shrabanapaul9/fastapi_file_manager*