

FastAPI File Management using PostgreSQL, MinIO, and Caching

By: Shrabana Paul

Mentor: Suprava Das

Institution: IDEAS-TIH, ISI Kolkata

Internship Duration: Aug–Oct 2025

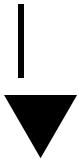
Abstract

- 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.

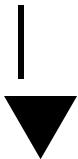
System Architecture / Workflow

FASTAPI FILE MANAGEMENT SYSTEM

*User → FastAPI API Layer → PostgreSQL (Metadata)
→ MinIO (File Storage)*



Pandas Merge → FastAPI Cache (Temporary)



*Save Merged File → MinIO (Final Storage) +
PostgreSQL (Updated Metadata)*

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

The screenshot shows the pgAdmin 4 interface. The left sidebar is the Object Explorer, displaying the database structure under 'file_management'. The main area is a query editor window titled 'file_management/postgres@PostgreSQL 17'. It contains a SQL query:

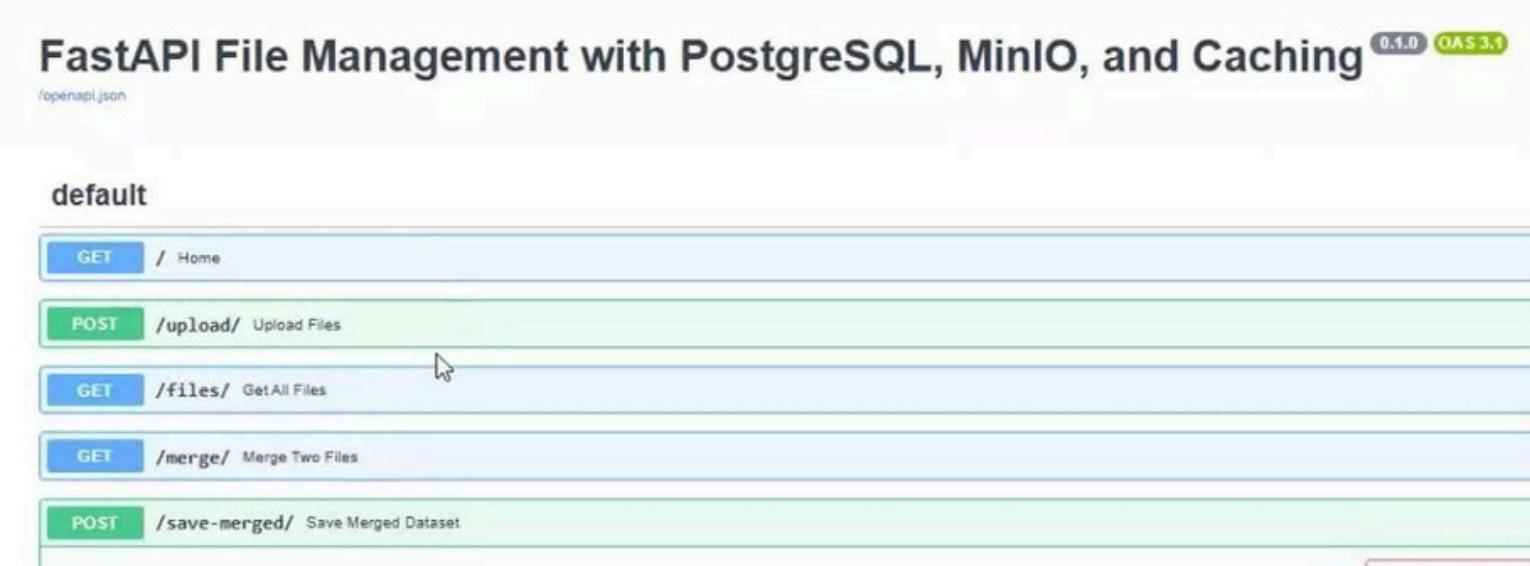
```
1 SELECT * FROM public.files ORDER BY id DESC;
```

The results of the query are displayed in a data grid below:

	id [PK] integer	file_name character varying (255)	file_format character varying (10)	upload_time timestamp without time zone
1	13	merged_20251109_20470...	csv	2025-11-09 20:47:07.424061
2	12	Retail_Data_Transactions.c...	csv	2025-11-09 20:40:27.758175
3	11	Retail_Data_Response.csv	csv	2025-11-09 20:40:27.758175
4	10	merged_20251108_22042...	csv	2025-11-08 22:04:22.18222
5	9	Retail_Data_Transactions.c...	csv	2025-11-08 22:02:52.083722
6	8	Retail_Data_Response.csv	csv	2025-11-08 22:02:52.083722

At the bottom of the pgAdmin window, there is a status bar with the message "Awesome Screen Recorder & Screenshot is sharing your screen." and a "Stop sharing" button.

API Testing Environment



The screenshot shows the Swagger UI interface for a FastAPI application. At the top, the title is "FastAPI File Management with PostgreSQL, MinIO, and Caching". Below the title, there are two small buttons: "openapi.json" and "0.1.0 OAS 3.1". The main area is titled "default". It lists several API endpoints:

- GET / Home
- POST /upload/ Upload Files
- GET /files/ GetAll Files
- GET /merge/ Merge Two Files
- POST /save-merged/ Save Merged Dataset

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

GET /merge/ Merge Two Files

Parameters

Name	Description
file_id_1 * required integer (query)	11
file_id_2 * required integer (query)	12

Execute

File IDs 11 and 12 selected for merging.

Cached Merge Preview

```
Response body
{
    "cache_key": "d29dbfff-381a-4eb4-8fa8-64b9591ab47f",
    "preview": [
        {
            "customer_id": "CS1112",
            "response": 0,
            "trans_date": "14-Jan-15",
            "tran_amount": 30
        },
        {
            "customer_id": "CS1112",
            "response": 0,
            "trans_date": "16-Jul-14",
            "tran_amount": 90
        },
        {
            "customer_id": "CS1112",
            "response": 0,
            "trans_date": "29-Apr-14",
            "tran_amount": 60
        },
        {
            "customer_id": "CS1112",
            "response": 0,
            "trans_date": "04-Dec-14",
            "tran_amount": 30
        }
    ]
}
```

FastAPI returns preview of merged cached dataset.

Save Merged Dataset

Response body

```
{  
  "message": "Merged dataset saved successfully to MinIO!",  
  "file_name": "merged_2825109_284787.csv"  
}
```

Merged dataset saved successfully to MinIO.

Final Files in MinIO

The screenshot shows the MinIO storage interface with the following details:

- Path:** files / merged_20251109_204707.csv
- Created on:** Sun, Nov 09 2025 20:38:05 (GMT+5:30)
- Access:** PRIVATE
- Size:** 2.6 MiB
- Actions:** Rewind, Create new path

<input type="checkbox"/>	Name	Last Modified	Size
<input type="checkbox"/>	merged_20251109_204707.csv	Today, 20:47	2.7 MiB
<input type="checkbox"/>	Retail_Data_Response.csv	Today, 20:40	67.2 KiB
<input type="checkbox"/>	Retail_Data_Transactions.csv	Today, 20:40	2.5 MiB

MinIO storage shows merged and raw datasets.

Results

- 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