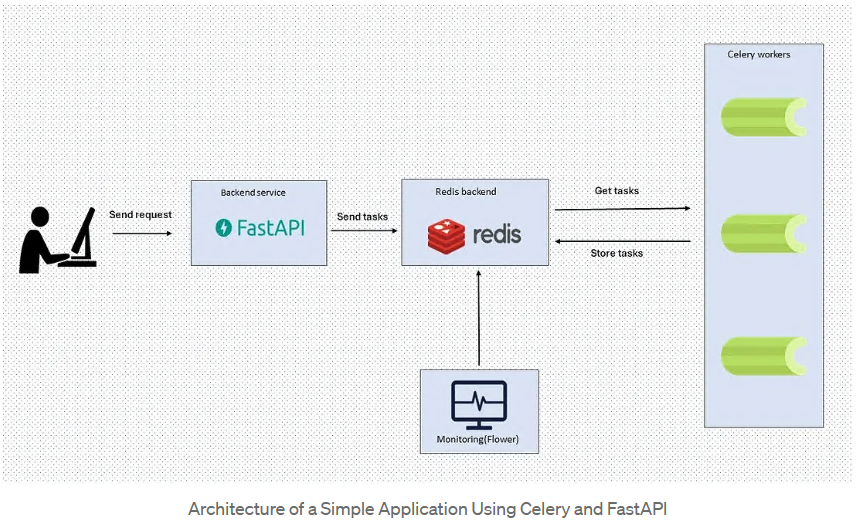
1. References

- <https://gitlab.tma.com.vn/tip2/dksh-dev>

- <https://medium.com/@youssefchamrah/empowering-applications-with-asynchronous-magic-the-celery-fastapi-docker-and-flower-ac119efc2e04>

1. Project components



* FastAPI service: Receive user’s request
* Celery backend: Consume tasks from Redis
* Databases:
  + Redis database: A message broker in Celery to manage task queues.
  + PostgreSQL database: Store task status for Portal consuming.

1. Setup

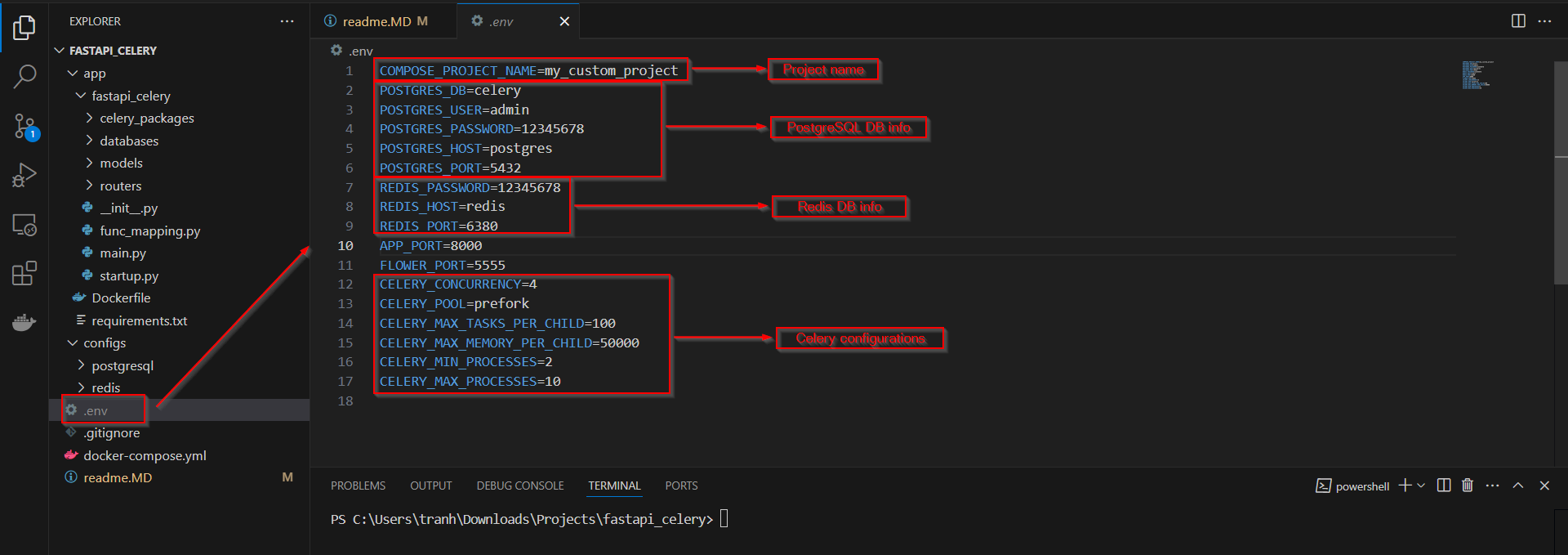
* Ensure the docker and docker-compose have been installed on the local machine already:

- Use Ubuntu or Windows Subsystem for Linux for the project

- Follow the guideline to install docker on Ubuntu distribution: [Ubuntu | Docker Docs](https://docs.docker.com/desktop/setup/install/linux/ubuntu/)

- Install docker-compose on Linux by command:

sudo curl -L "https://github.com/docker/compose/releases/download/`curl -fsSLI -o /dev/null -w %{url\_effective} https://github.com/docker/compose/releases/latest | sed 's#.\*tag/##g' && echo`/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose && sudo chmod +x /usr/local/bin/docker-compose

* Clone code to your local machine and switch to branch named “fastapi-postgresql”
* Create a new file named “.env” to store the project information and all necessary credentials locally in the project’s root directory  
    
    
  ”.env” file example:  
  COMPOSE\_PROJECT\_NAME=my\_custom\_project

POSTGRES\_DB=celery

POSTGRES\_USER=admin

POSTGRES\_PASSWORD=12345678

POSTGRES\_HOST=**postgres**

POSTGRES\_PORT=5432

REDIS\_PASSWORD=12345678

REDIS\_HOST=redis

REDIS\_PORT=6380

APP\_PORT=8000

FLOWER\_PORT=5555

CELERY\_CONCURRENCY=4

CELERY\_POOL=prefork

CELERY\_MAX\_TASKS\_PER\_CHILD=100

CELERY\_MAX\_MEMORY\_PER\_CHILD=50000

CELERY\_MIN\_PROCESSES=2

CELERY\_MAX\_PROCESSES=10

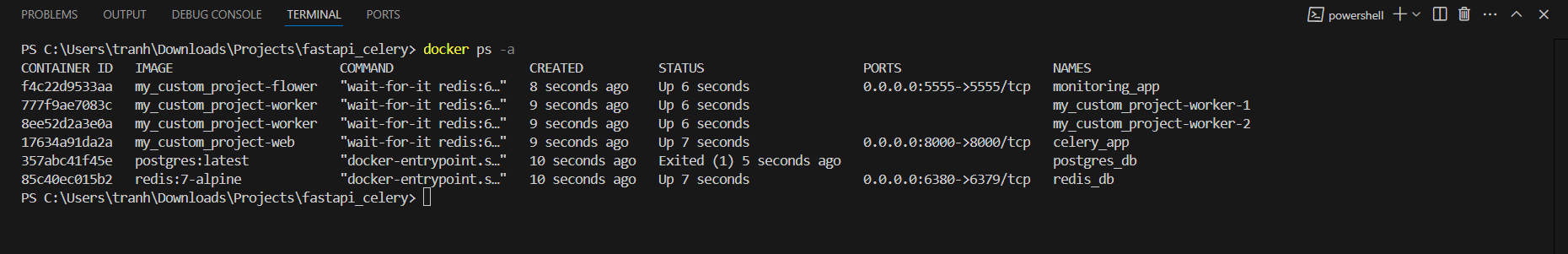
1. Test

* From the project’s root directory, run the command:

docker-compose up -d --build --force-recreate

# To list out all containers running

docker ps -a



* How to test sending a request to FastAPI

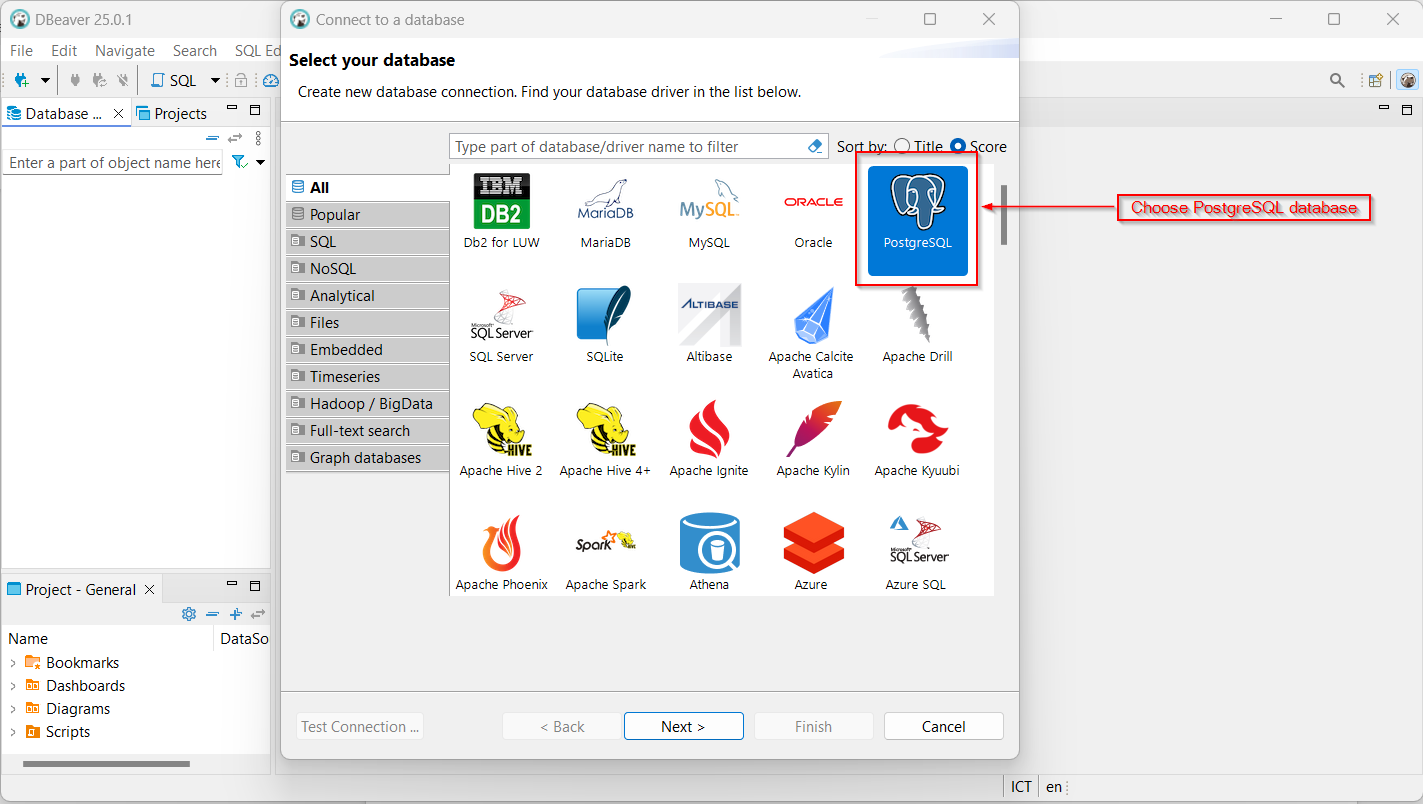
- Go to <http://localhost:8000/docs> and click on “Try out” button

* How to monitor the task status

- Go to <http://localhost:5555>. There will get information of all Celery workers and tasks running

* How to retrieve historical task information stored in the PostgreSQL database

- To connect to the PostgreSQL database using DBeaver tool:



- And provide database connection information following below then click on “Test Connection” button to validate provided information

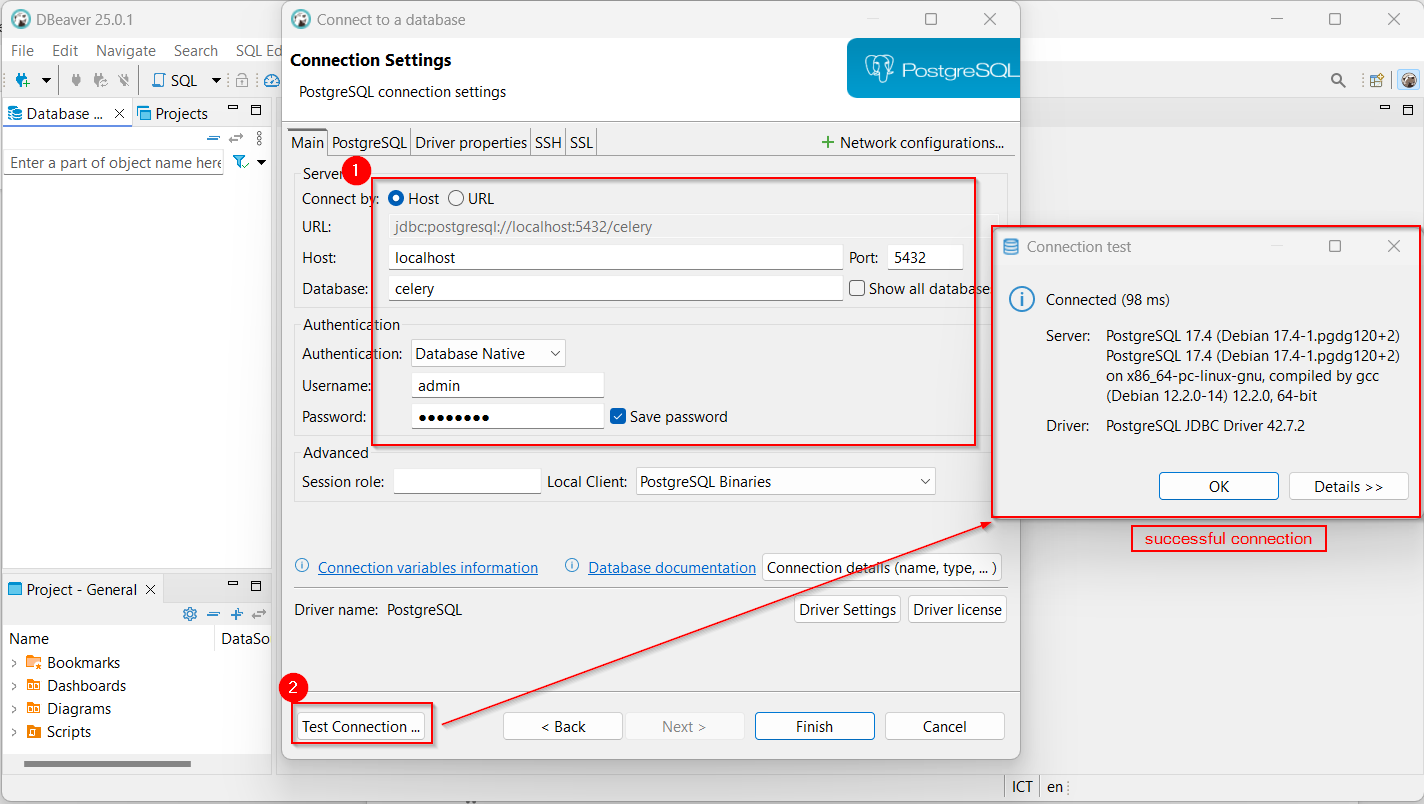
Host: localhost/<Ubuntu-IP-address>

Port: 5432

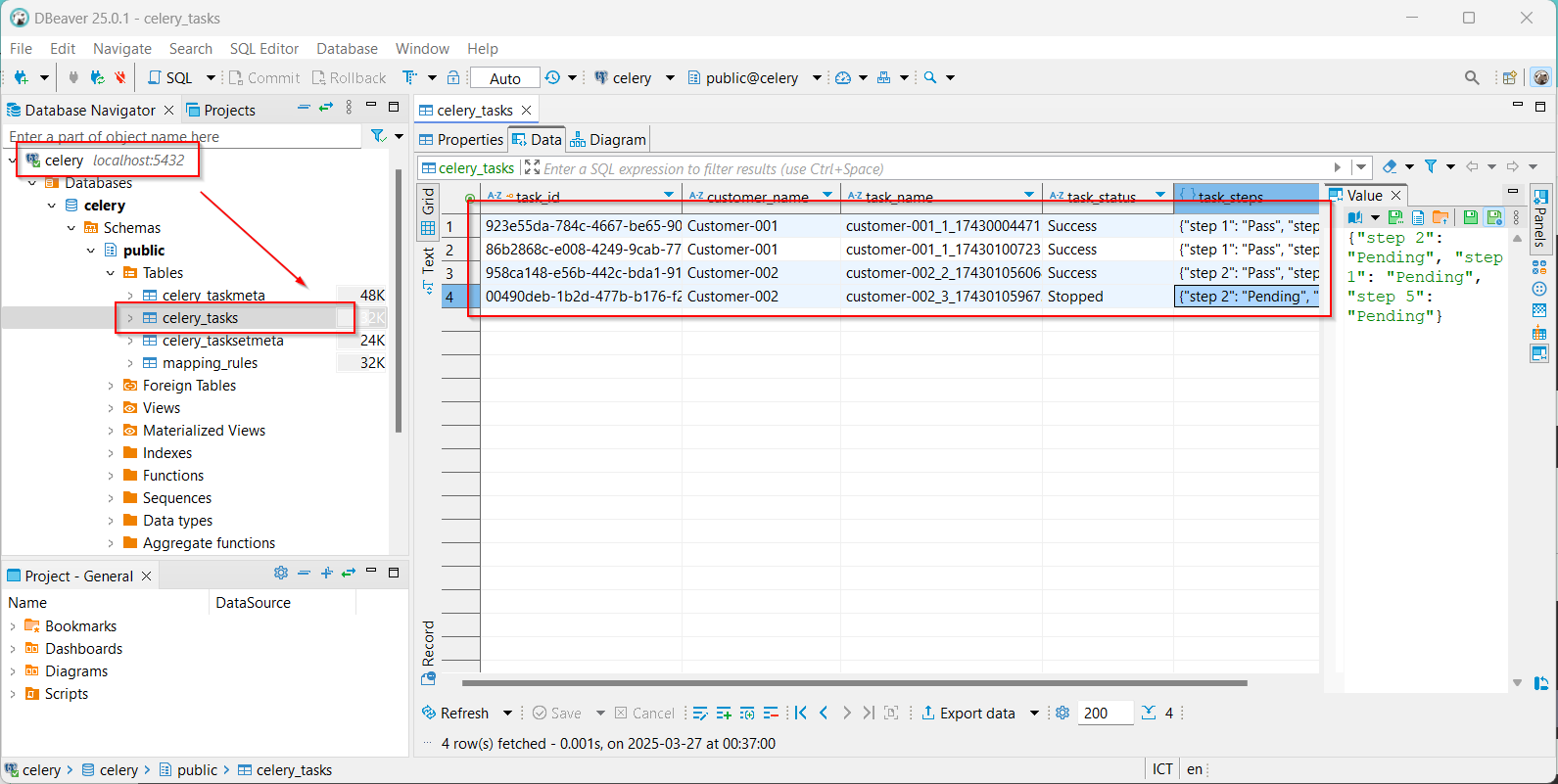
Authentication: Database Native

Username: admin

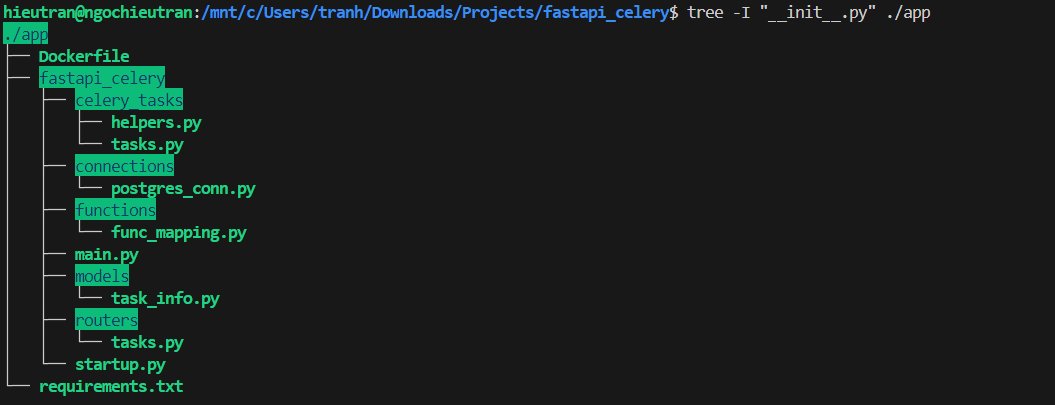
Password: 12345678



- Retrieve all necessary information of all historical tasks by querying the celery\_tasks table



1. Code structure



- **celery\_tasks**: Contains core logic for asynchronous tasks

- **connections**: Contains all database connection information and methods

- **functions**: Contains specific functions defined for specific jobs.

- **models**: Contains all data structures

- **routers**: Contains all defined FastAPI methods

- **utils**: Contains some extra modules such as helpers module for debugging, etc.

1. Debug

* Update .env file to use on local machine

**”.env” file example:**

COMPOSE\_PROJECT\_NAME=my\_custom\_project

POSTGRES\_DB=celery

CELERY\_SCHEMA=celery

POSTGRES\_USER=admin

POSTGRES\_PASSWORD=12345678

POSTGRES\_HOST=**localhost**

POSTGRES\_PORT=5432

REDIS\_PASSWORD=12345678

REDIS\_HOST=**localhost**

REDIS\_PORT=6380

APP\_PORT=8000

FLOWER\_PORT=5555

CELERY\_CONCURRENCY=4

CELERY\_POOL=prefork

CELERY\_MAX\_TASKS\_PER\_CHILD=100

CELERY\_MAX\_MEMORY\_PER\_CHILD=50000

CELERY\_MIN\_PROCESSES=2

CELERY\_MAX\_PROCESSES=10

* Install environment for testing:
  + Open Linux Terminal in your Ubuntu and run the command

sudo apt install -y virtualenv

virtualenv -p python3 <name-of-the-env>

# Or using with python

python3 -m pip install virtualenv

Python3 -m venv <name-of-the-env>

* Activate and De-activate the virtual environment

# Activate the environment

source <name-of-the-env>/bin/activate

# De-activate the environment

deactivate

* + Install the requirement libraries

pip install -r ./app/requirements.txt

# Install uvicorn for the API server simulation

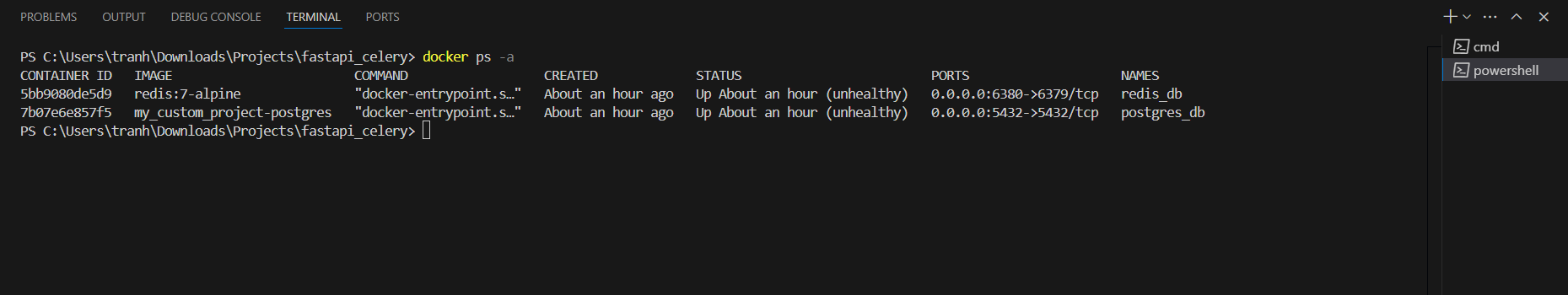
pip install uvicorn[standard] python-dotenv

* Debug
  + Start the docker containers for PostgreSQL and Redis databases

docker-compose -f docker-compose.dev.yml up -d --build --force-recreate

# And check all docker containers running

docker ps -a



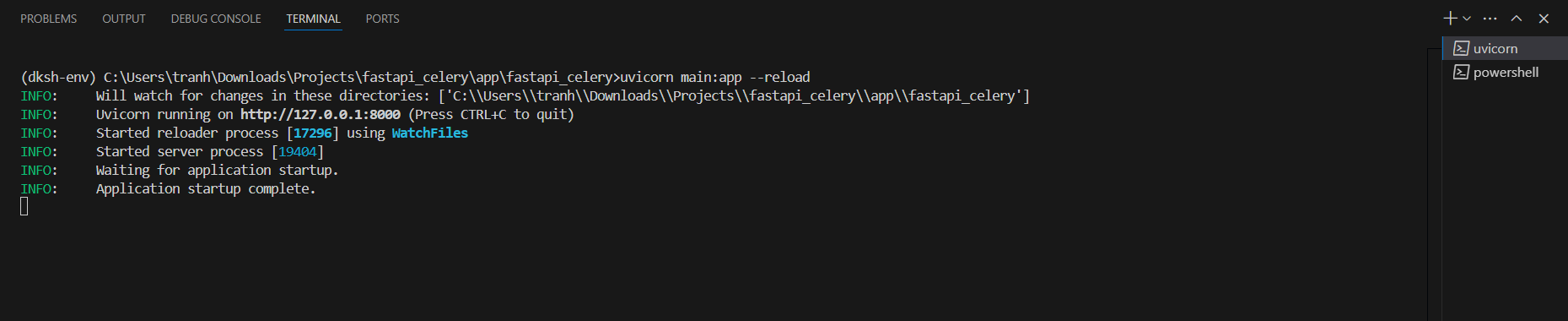
* Start the uvicorn to simulate the API server

# Change directory to fastapi\_celery folder

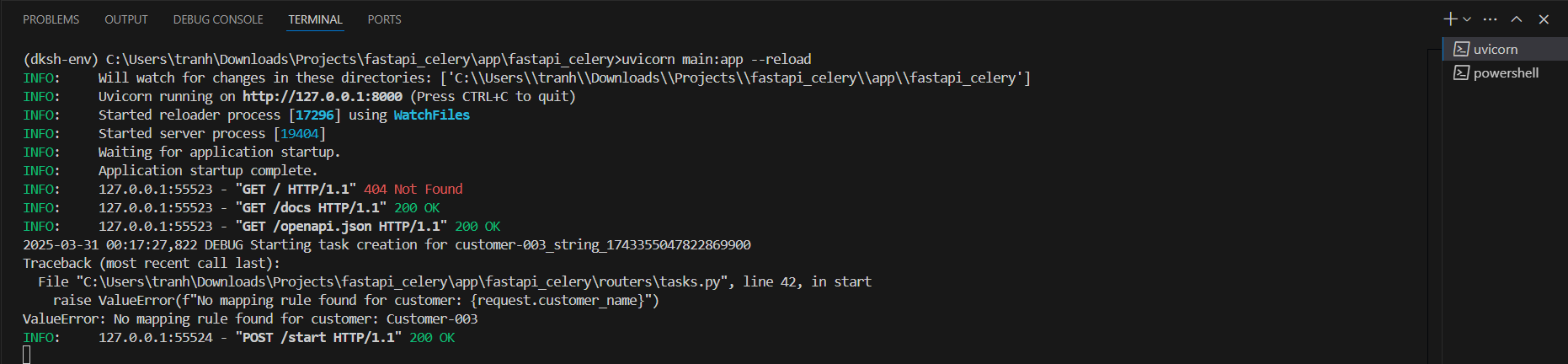
cd ./app/fastapi\_celery

# Start the simulated API server

uvicorn main:app --reload



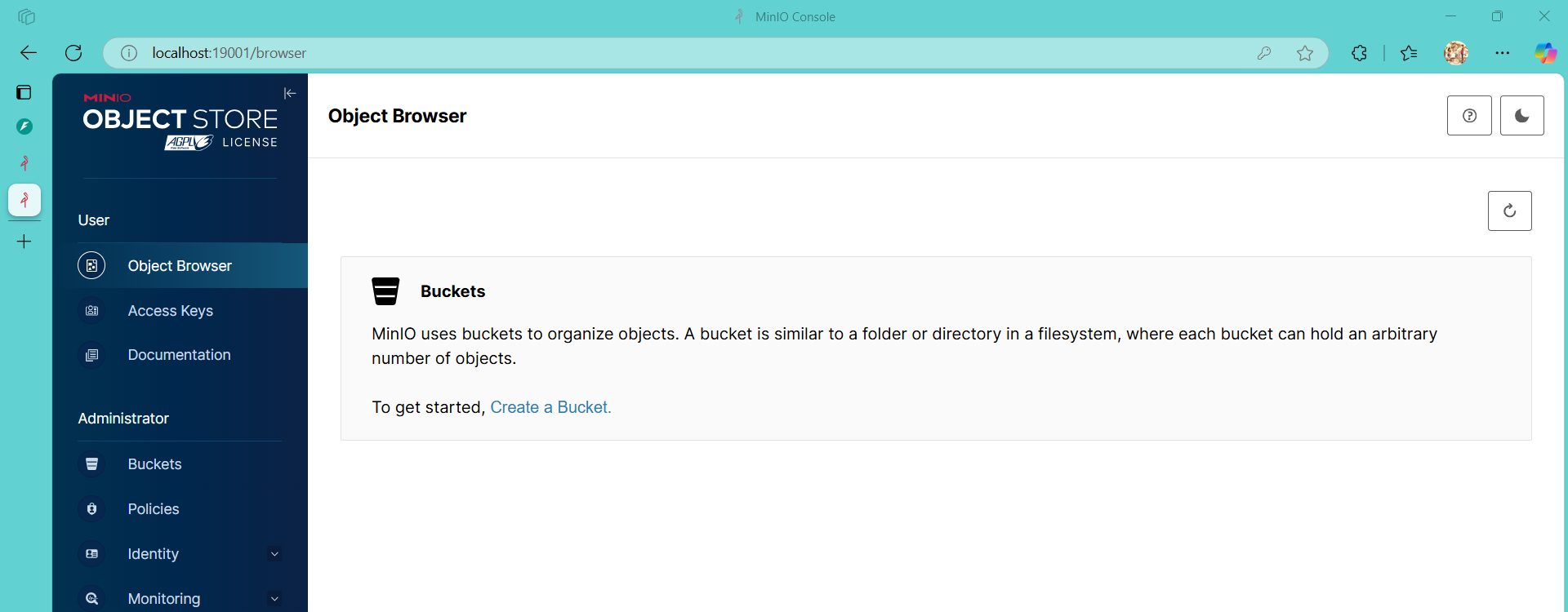
From the command line, we can see the log when a request’s sent to the simulated API sever



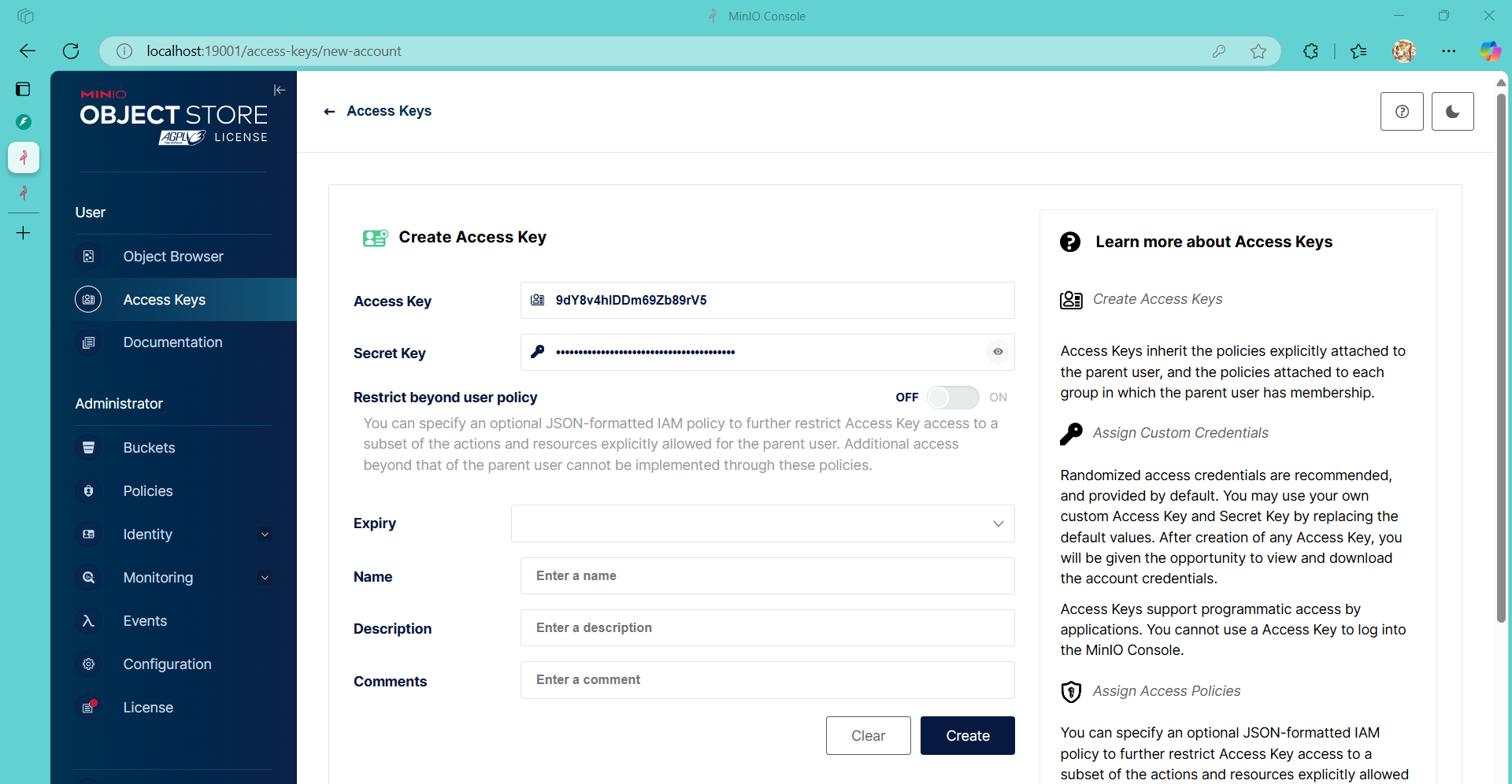
* To work with MinIO service to test S3 connection and file processing, we following some steps
  + Run docker compose to create MinIO container:

sudo docker-compose -f docker-compose.minio.yml up

* + Then access the MinIO service from your browser following the link  
    **<ubuntu-server-ip>:19001** or **localhost:19001** if using Docker Desktop on Windows



And create ACCESS\_KEY and ACCESS\_SECRET\_KEY



Then update the .env file

**”.env” file example:**

COMPOSE\_PROJECT\_NAME=my\_custom\_project

POSTGRES\_DB=celery

CELERY\_SCHEMA=celery

POSTGRES\_USER=admin

POSTGRES\_PASSWORD=12345678

POSTGRES\_HOST=localhost

POSTGRES\_PORT=5432

REDIS\_PASSWORD=12345678

REDIS\_HOST=localhost

REDIS\_PORT=6379

APP\_PORT=8000

FLOWER\_PORT=5555

CELERY\_CONCURRENCY=4

CELERY\_POOL=prefork

CELERY\_MAX\_TASKS\_PER\_CHILD=100

CELERY\_MAX\_MEMORY\_PER\_CHILD=50000

CELERY\_MIN\_PROCESSES=2

CELERY\_MAX\_PROCESSES=10

MINIO\_ROOT\_USER=minioadmin

MINIO\_ROOT\_PASSWORD=minioadmin

MINIO\_PORT=19000

MINIO\_CONSOLE\_PORT=19001

S3\_ACCESS\_KEY=**<ACCESS\_KEY>**

S3\_ACCESS\_SECRET=**<SECRET\_ACCESS\_KEY>**