Dockerized Python + PostgreSQL – Step-by-Step Project Documentation

1. Introduction

This project demonstrates how to use Docker to containerize a simple Python application and connect it to a PostgreSQL database running in another container.

By the end of the project, the app:

Connects to PostgreSQL

Creates a table

Inserts a row

Reads and prints the data

Everything runs inside Docker containers using a custom Docker network.

2. Tools & Technologies Used

Operating System: Windows 10/11 (64-bit)

Docker Desktop (Linux containers)

Visual Studio Code (VS Code)

Python 3.10 (Docker image: python:3.10-slim)

PostgreSQL (Docker image: postgres)

Python library: psycopg2-binary (PostgreSQL driver)

3. Verifying Docker Installation
Step 3.1 – Check Docker is installed and running

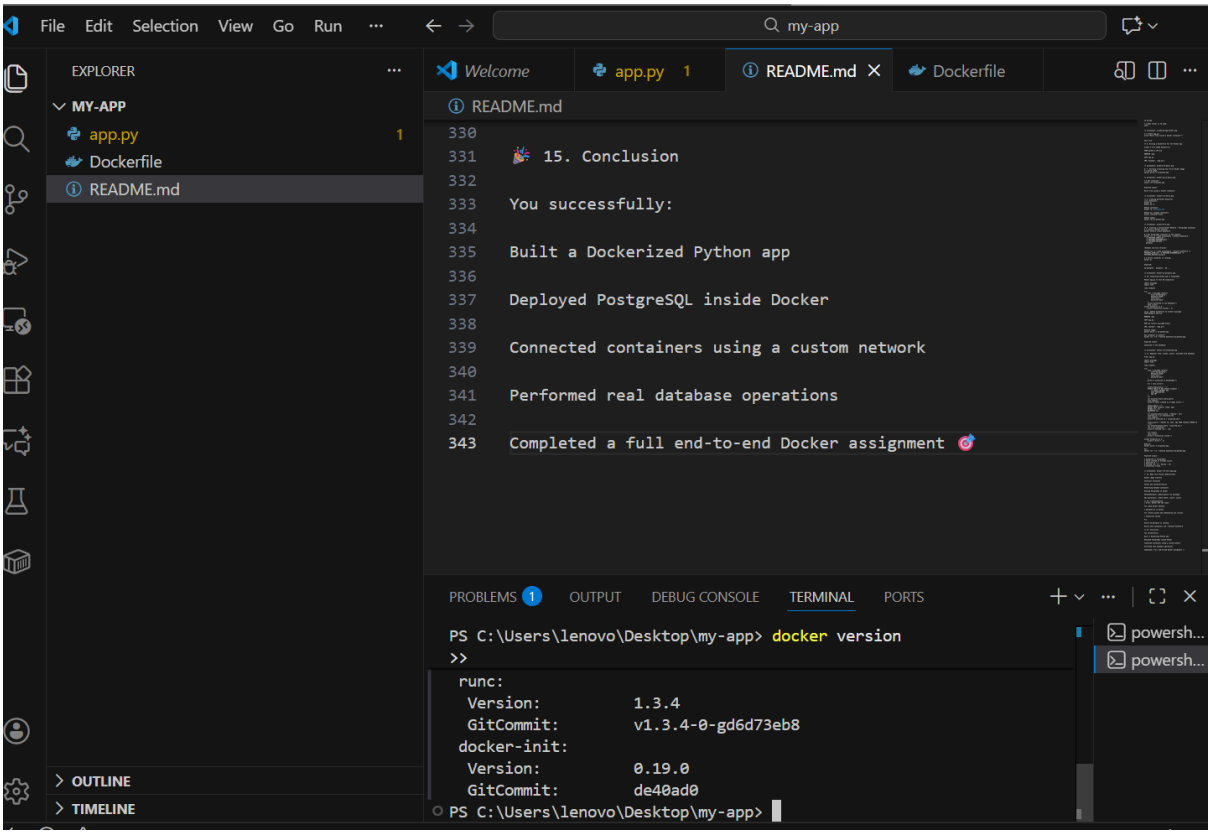
Open Command Prompt or PowerShell and run:

docker version

This shows both Client and Server information.

✅ If both sections appear → Docker Desktop is installed and running correctly.

Screenshot 1:



4. Setting Up VS Code With Docker

Step 4.1 – Install VS Code

Download from:
https://code.visualstudio.com/

Step 4.2 – Install Docker Extension in VS Code

1. Open VS Code
2. Click on the **Extensions** icon on the left sidebar
3. Search for **“Docker”** (by Microsoft)
4. Click **Install**

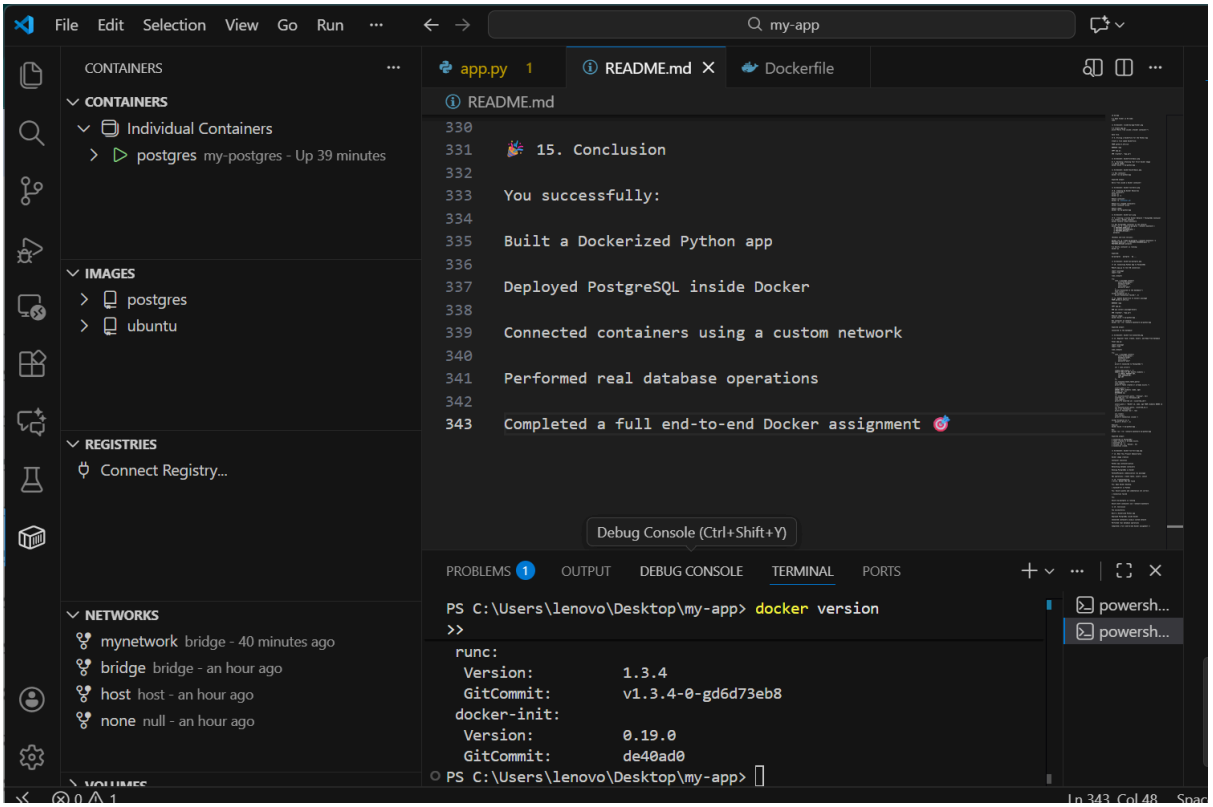
Screenshot 2:

Step 4.3 – Verify Docker view in VS Code

1. Click the **Docker** icon on the left sidebar
2. You should see sections like:

- Containers
- Images
- Volumes
- Networks

Screenshot 3:



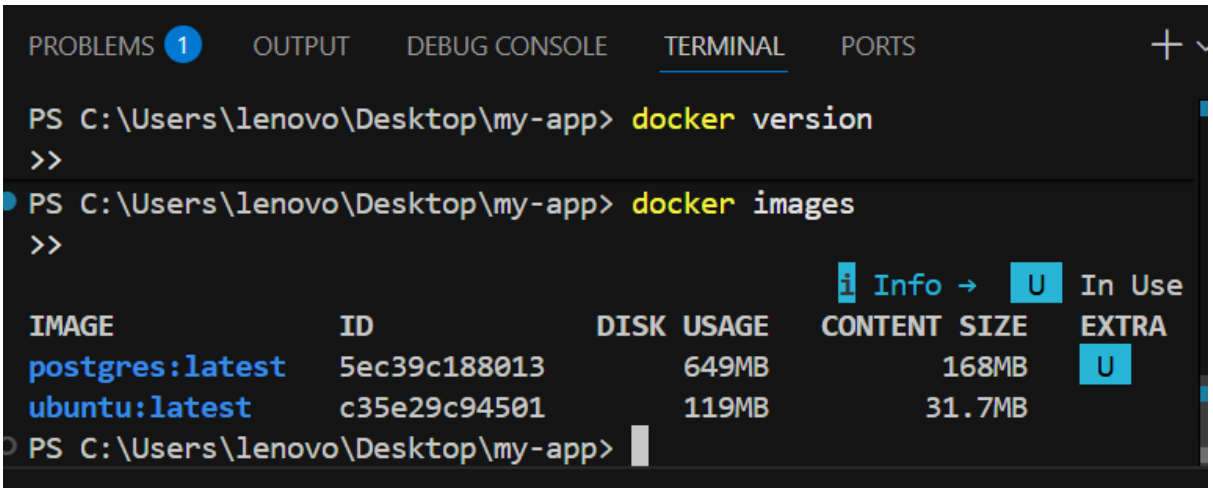
Step 5.1 – Pull a Base Image

Run the following command to pull an Ubuntu image:

```
docker pull ubuntu
```

Step 5.2 – View Downloaded Images

```
docker images
```



6. Creating a Simple Python App

Step 6.1 – Create a project folder

```
mkdir my-app
cd my-app
```

Step 6.2 – Open the folder in VS Code

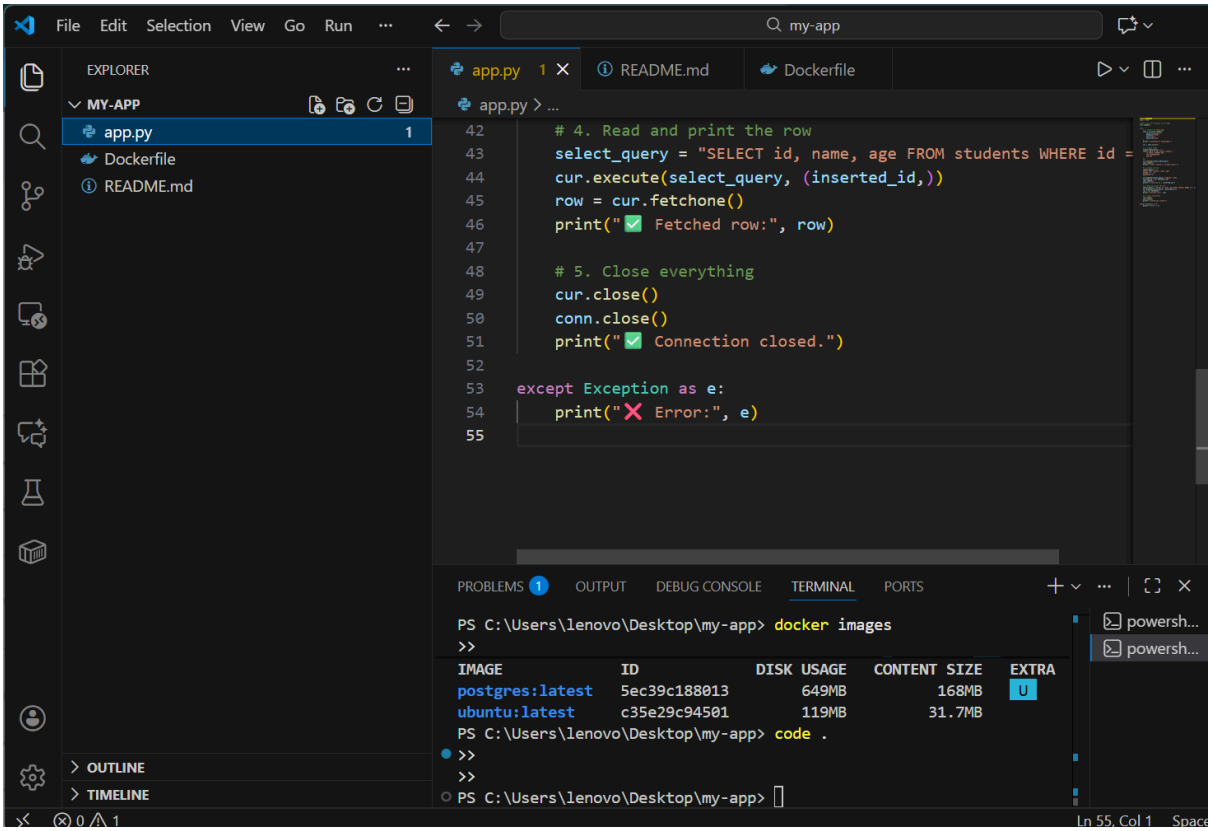
```
code .
```

Step 6.3 – Create `app.py`

Inside VS Code → New File → `app.py`

7. Creating the Dockerfile

Create a new file named **Dockerfile** (no extension):



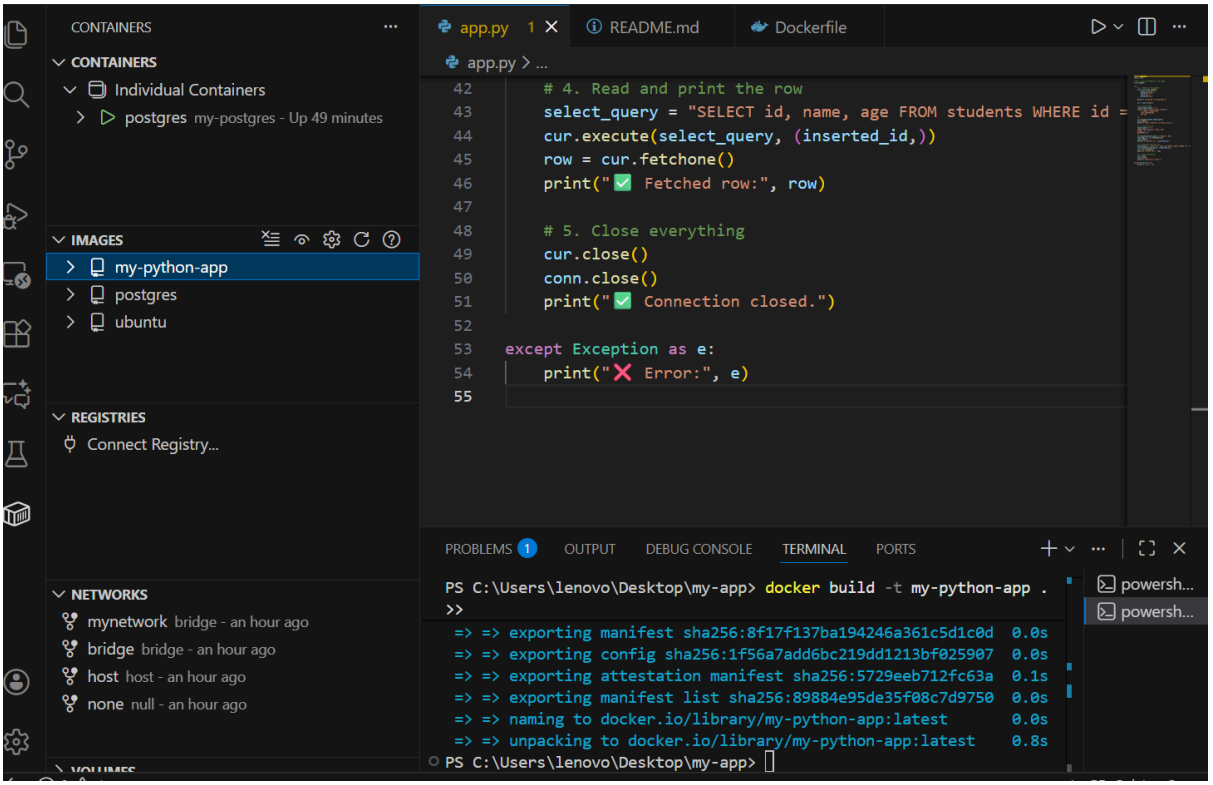
8. Building the Docker Image

Run:

```
docker build -t my-python-app .
```

9. Running the Python Container

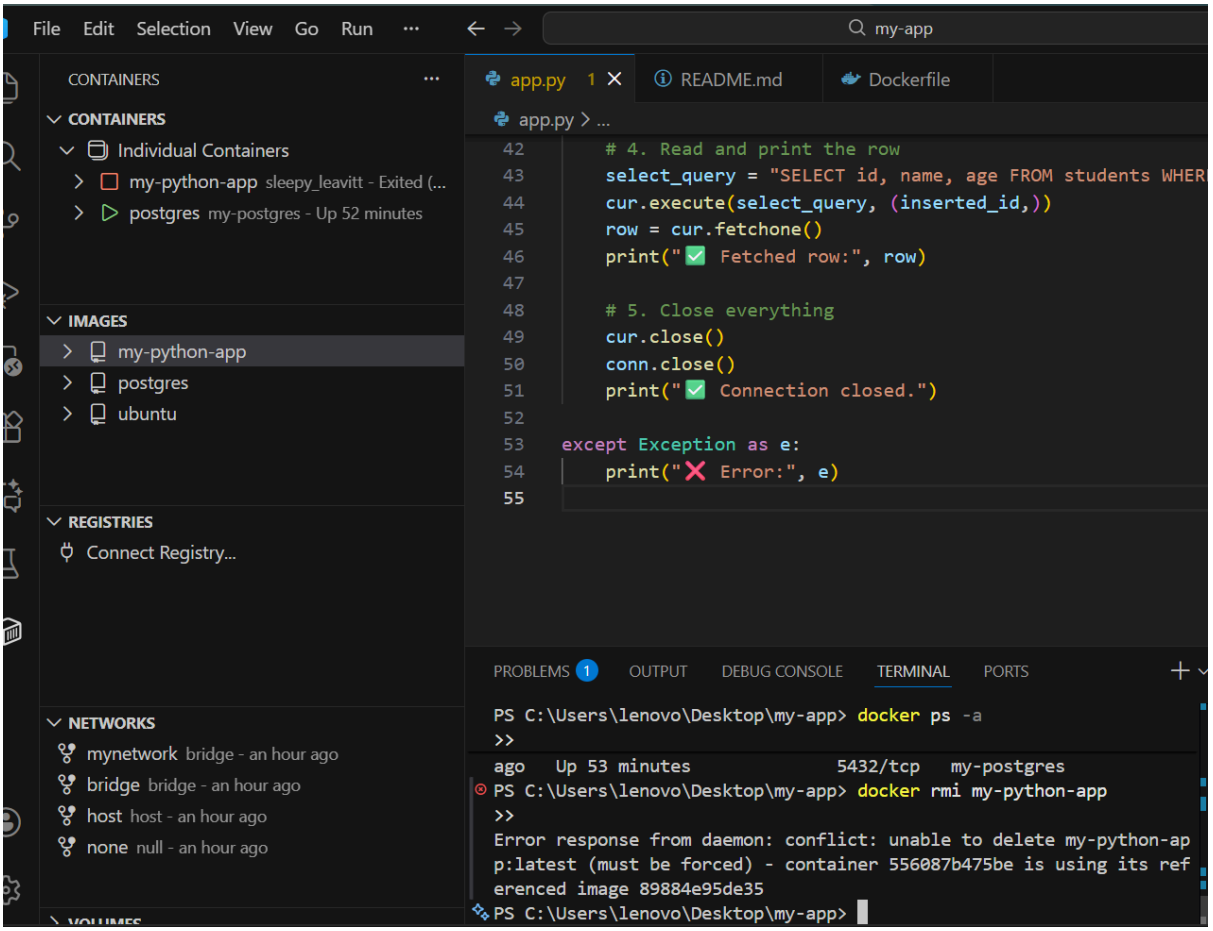
Run the container:



Cleaning Up Docker Resources (Optional)

View containers:

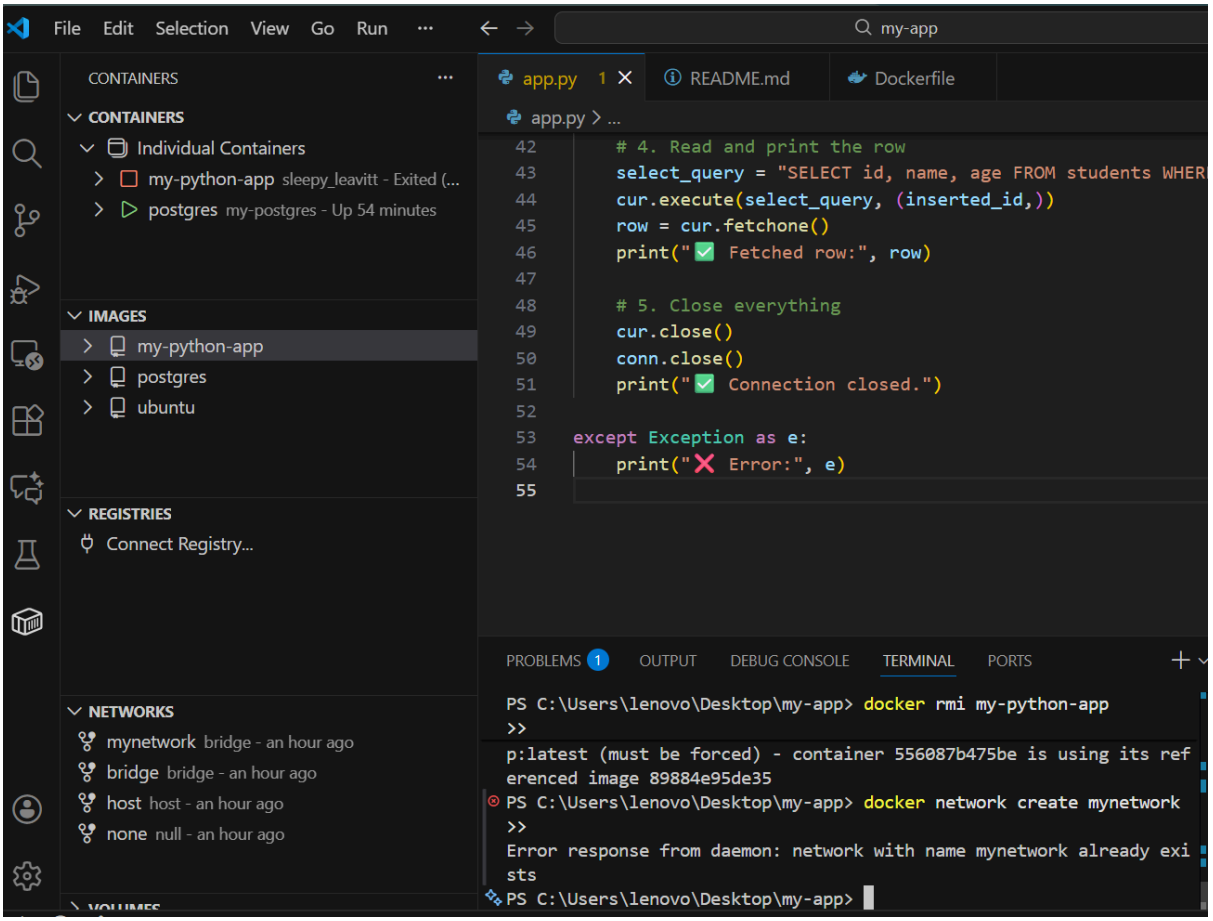
Remove an image:



11. Creating a Custom Docker Network

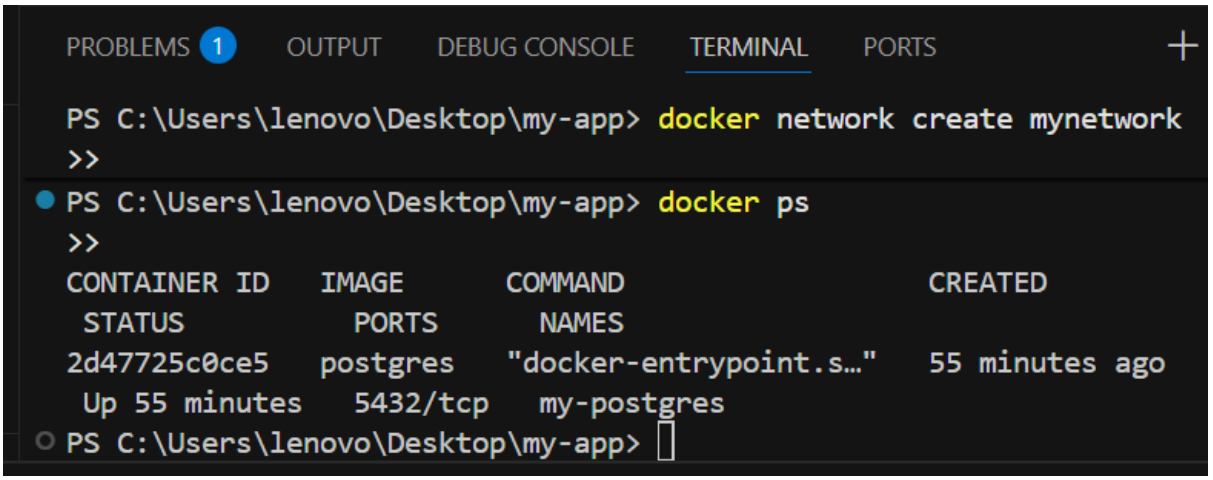
A network allows containers to communicate with each other.

Create the network:



12. Running PostgreSQL in a Container

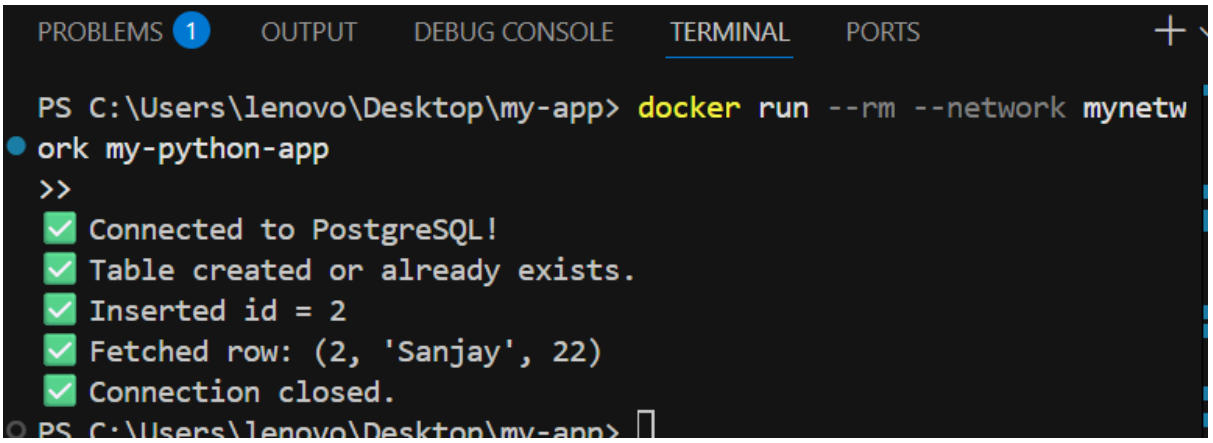
Run PostgreSQL and attach it to the custom network:



15. Rebuilding and Running the Connected App
Rebuild:

docker build -t my-python-app .

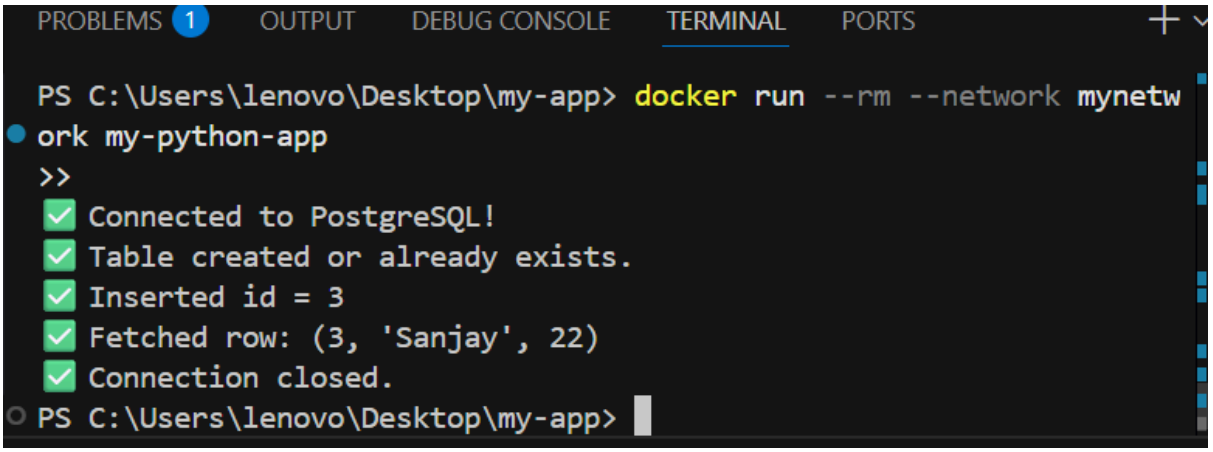
Run:
docker run --rm --network mynetwork my-python-app



16. Final Beginner Task – Full Database Operations

Update app.py to:

- Create a table
- Insert a row
- Read data
- Print the result



17. Conclusion

This project successfully demonstrates:

✔ Pulling images from Docker Hub
✔ Creating Dockerfiles
✔ Building and running custom Docker images
✔ Running PostgreSQL as a container
✔ Creating Docker networks
✔ Connecting two containers
✔ Performing SQL operations inside the Dockerized Python app