**Scenario - Backstage, PostgreSQL 15, and pgAdmin on Ubuntu (Docker Compose)**

**Prerequisites**

* **Docker** (Install using: sudo apt install docker.io -y)
* **Docker Compose** (Install using: sudo apt install docker-compose -y)
* **Git** (Optional, if cloning from GitHub)

**Step 1: Clone the Repository (Optional)**

git clone https://github.com/Krishnamurtyp/docker-backstage.git

cd docker-backstage

Or, create a new directory manually:

mkdir backstage-setup && cd backstage-setup

**Step 2: Create the docker-compose.yml File**

Inside the backstage-setup directory, create a file named **docker-compose.yml** and add the following content:

version: '3.8'

services:

backstage:

image: roadiehq/community-backstage-image:latest

ports:

- "7000:7000" # Backstage UI at http://localhost:7000

environment:

- GITHUB\_TOKEN=[YOUR\_GITHUB\_TOKEN] # Replace with your token

- POSTGRES\_HOST=db

- POSTGRES\_PORT=5432

- POSTGRES\_USER=backstage\_user

- POSTGRES\_PASSWORD=backstage\_password

- POSTGRES\_DB=backstage\_db

depends\_on:

db:

condition: service\_healthy

networks:

- backstage-net

db:

image: postgres:15

environment:

- POSTGRES\_USER=backstage\_user

- POSTGRES\_PASSWORD=backstage\_password

- POSTGRES\_DB=backstage\_db

ports:

- "5432:5432" # Optional: Expose PostgreSQL port locally

volumes:

- postgres-data:/var/lib/postgresql/data

healthcheck:

test: ["CMD-SHELL", "pg\_isready -U backstage\_user -d backstage\_db"]

interval: 10s

timeout: 5s

retries: 5

networks:

- backstage-net

pgadmin:

image: dpage/pgadmin4:latest

ports:

- "5050:80" # pgAdmin UI at http://localhost:5050

environment:

- PGADMIN\_DEFAULT\_EMAIL=admin@admin.com

- PGADMIN\_DEFAULT\_PASSWORD=admin

depends\_on:

- db

networks:

- backstage-net

networks:

backstage-net:

driver: bridge

volumes:

postgres-data:

**Step 3: Start the Services**

Run the following command to start all containers in detached mode:

docker-compose up -d

Check if all containers are running:

docker ps

You should see:

* backstage running on **port 7000**
* postgres-db running on **port 5432**
* pgadmin running on **port 5050**

**Step 4: Access the Services**

1. **Backstage UI:** http://your-vm-ip:7000
2. **pgAdmin UI:** http://your-vm-ip:5050
   * **Login Credentials**
     + Email: admin@admin.com
     + Password: admin

**Step 5: Connect PostgreSQL in pgAdmin**

1. Open **pgAdmin** at http://your-vm-ip:5050
2. **Create a New Server** → Name: PostgreSQL
3. **Connection Tab:**
   * Hostname: postgres
   * Port: 5432
   * Username: backstage
   * Password: backstage
4. **Save & Connect**

**Step 6: Verify PostgreSQL Data**

Run the following query in **pgAdmin → Query Tool**:

SELECT \* FROM catalog\_entities;

If no data is present, restart the **Backstage** container:

docker restart backstage

**Step 7: Test Backstage**

1. Open **Backstage UI** → Click **Create Component**
2. Select **Example Templates** → Create an entity
3. Go to **Catalog** → View the entity

**Step 8: Stop and Restart Services (If Needed)**

To stop all services:

docker-compose down

To restart:

docker-compose up -d

**Verification**

| **Service** | **URL** | **Status Check Command** |
| --- | --- | --- |
| **Backstage UI** | http://your-vm-ip:7007 | docker logs backstage |
| **pgAdmin UI** | http://your-vm-ip:5050 | docker logs pgadmin |
| **PostgreSQL** | Connected in pgAdmin | docker logs postgres-db |