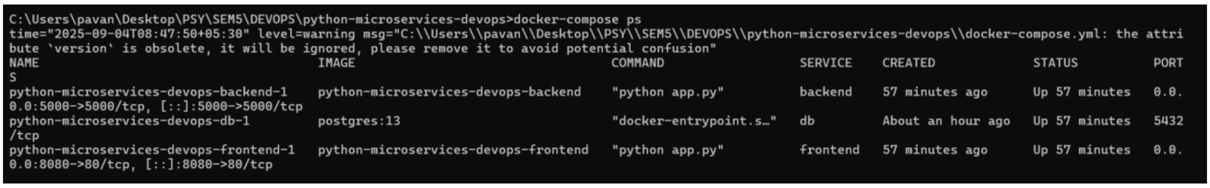
DevOps Assignment Deliverables

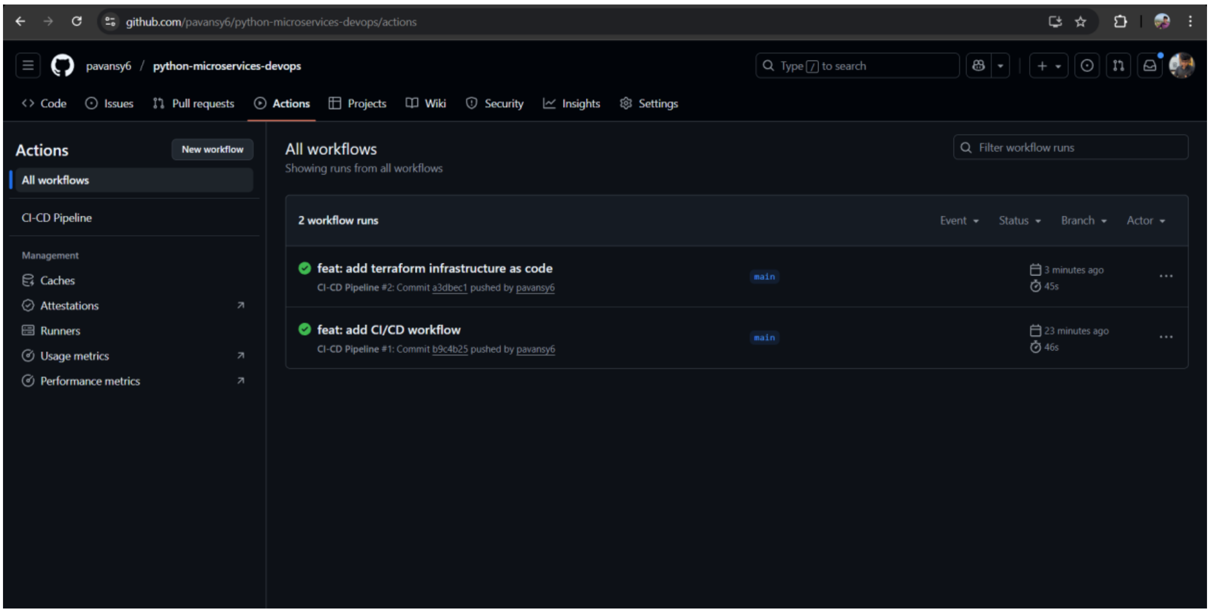
GitHub Link: https://github.com/pavansy6/python-microservices-devops

1st



```
C:\Users\pavan\Desktop\PSY\SEM5\DEVOPS\python-microservices-devops>docker-compose ps
time="2025-09-04T08:47:50+05:30" level=warning msg="C:\\Users\\pavan\\Desktop\\PSY\\SEM5\\DEVOPS\\python-microservices-devops\\docker-compose.yml: the attri
bute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
NAME                                     IMAGE                                    COMMAND                    SERVICE     CREATED              STATUS          PORT
S
python-microservices-devops-backend-1    python-microservices-devops-backend     "python app.py"            backend     57 minutes ago       Up 57 minutes   0.0.
0.0:5000->5000/tcp, [::]:5000->5000/tcp
python-microservices-devops-db-1         postgres:13                              "docker-entrypoint.s…"    db          About an hour ago    Up 57 minutes   5432
/tcp
python-microservices-devops-frontend-1   python-microservices-devops-frontend    "python app.py"            frontend    57 minutes ago       Up 57 minutes   0.0.
0.0:8080->80/tcp, [::]:8080->80/tcp
```

2nd



All workflows

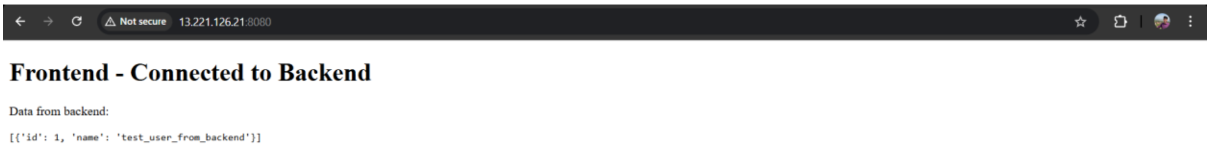
Showing runs from all workflows

2 workflow runs

feat: add terraform infrastructure as code — CI-CD Pipeline #2: Commit a3dbec1 pushed by pavansy6 — main — 3 minutes ago — 45s

feat: add CI/CD workflow — CI-CD Pipeline #1: Commit b9c4b25 pushed by pavansy6 — main — 23 minutes ago — 46s

3rd



Frontend - Connected to Backend

Data from backend:

```
[{'id': 1, 'name': 'test_user_from_backend'}]
```

1. What was the hardest part?
   The hardest part was figuring out the AWS permissions error. The Terraform code was correct, but the deployment failed because the cloud user didn't have the right permissions, which required logging into the AWS console to attach the correct security policy before the server could be built.

2. How does CI/CD + IaC help reduce manual effort?
   CI/CD with GitHub Actions saves time by automatically building and publishing our Docker images whenever we push code. Infrastructure as Code with Terraform lets us create and delete an entire cloud server with a single command, which is much faster and more reliable than setting it up manually through the AWS website.