

## Task 1 – Dockerization of Application

### Description:

This task focuses on containerizing an application using Docker to ensure consistency across development and deployment environments. Dockerization helps eliminate environment-related issues and makes application deployment faster and more reliable. The task demonstrates core containerization concepts used in modern DevOps workflows.

### Overview of the Task:

The application was packaged into a Docker container using a Dockerfile, and Docker Compose was used to manage multi-container services. The setup allows the application to run seamlessly on any system with Docker installed.

### Tools & Technologies Used:

- Docker
- Docker Compose
- FastAPI
- Linux shell commands

### Dockerization Workflow:

1. Created a Dockerfile to define the application image.
2. Built the Docker image using Docker CLI.
3. Defined services using docker-compose.yml.
4. Ran the application inside containers.
5. Verified container health and logs.

### Commands Used:

```
docker build -t app .
docker compose up --build
docker ps
docker logs <container_name>
docker compose stop
docker compose start
```

### Configuration Files:

- Dockerfile
- docker-compose.yml

- requirements.txt

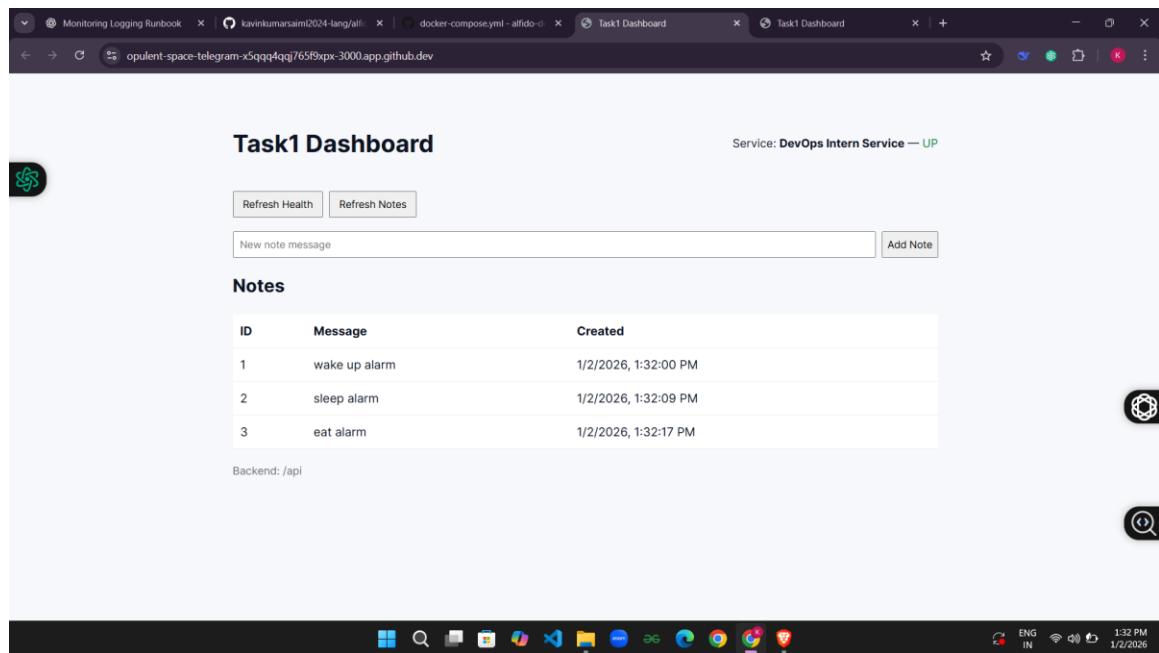
#### Outcome:

The application was successfully containerized and executed inside Docker containers. This task improved understanding of Docker images, containers, networking, and real-world deployment practices.

#### GitHub Repository:

<https://github.com/Kavi-n-alt/alfido-devops-labs>

#### Output ScreenShots:



```
app.py - alido-devops-labs [C:\] animated-train-g474rj47jq3ppqf6-5000.app.github.dev
Pretty-print □
("message": "Hello from Dockerized Flask App! (updated)")


[{"Docker User"}]
```

The screenshot shows a browser window displaying the output of the Python application. The terminal below shows the command used to run the application and the resulting JSON response.

```
curl -sS http://localhost:5000/user
[{"Docker User"}]
```

PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL PORTS 4

```
@Kavi-n-alt → /workspaces/alfido-devops-labs/task1-dockerize (main) $ docker compose up --build -d
=> [web] exporting to image
=> => exporting layers
=> => writing image sha256:282768fc284c05c79d58c7132fe562bfa0e88c4a9bfe9616e4ab661e8eb1574c
=> => naming to docker.io/library/task1-dockerize-web
=> [streamlit] exporting to image
=> => exporting layers
=> => writing image sha256:083033004155ff73f844d2c91a88e4d4479be34f7bf97f65cca6cd81050fb4d
=> => naming to docker.io/library/task1-dockerize-streamlit
=> [streamlit] resolving provenance for metadata file
=> [web] resolving provenance for metadata file
[+] Running 5/5
✓ task1-dockerize-streamlit          Built
✓ task1-dockerize-web                Built
✓ Container task1-dockerize-streamlit-1 Started
✓ Container task1-dockerize-db-1      Running
✓ Container task1-dockerize-web-1     Started
@Kavi-n-alt → /workspaces/alfido-devops-labs/task1-dockerize (main) $ 
✓ Container task1-dockerize-web-1     started                                     0.4S
@Kavi-n-alt → /workspaces/alfido-devops-labs/task1-dockerize (main) $ docker ps
CONTAINER ID   IMAGE               COMMAND                  CREATED             STATUS              PORTS
NAMES
64ab827b94723  task1-dockerize-streamlit "sh /app/dev-entrypo..."  About a minute ago   Up About a minute   5000/tcp, 0.0.0.0:8501->8501/tcp, [::]:8501->8501/tcp
bab5207755df   task1-dockerize-web    "sh /app/dev-entrypo..."  About a minute ago   Up About a minute   0.0.0.0:5000->5000/tcp, [::]:5000->5000/tcp
d7dc8739121a   postgres:15         "docker-entrypoint.s..."  9 minutes ago       Up 9 minutes       5432/tcp
@Kavi-n-alt → /workspaces/alfido-devops-labs/task1-dockerize (main) $ curl -ss http://localhost:5000/users
[[{"id": 1, "name": "Docker User"}]]
@Kavi-n-alt → /workspaces/alfido-devops-labs/task1-dockerize (main) $ 
```

