

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

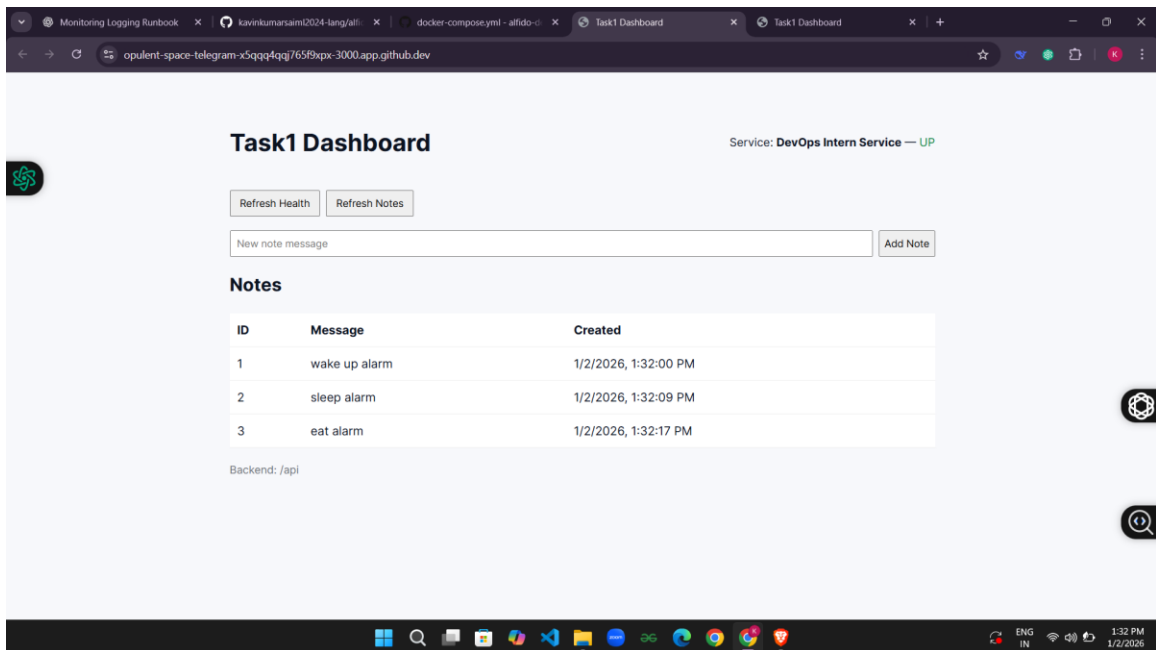
Outcome:

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:



The screenshot displays a web browser window with multiple tabs. The active tab is titled 'Task1 Dashboard' and shows the URL 'opulent-space-telegram-x5qq4qq765f9px-3000.app.github.dev'. The dashboard itself has a header with the title 'Task1 Dashboard' and a service status indicator 'Service: DevOps Intern Service — UP'. Below the header, there are two buttons: 'Refresh Health' and 'Refresh Notes'. A text input field labeled 'New note message' is followed by an 'Add Note' button. A section titled 'Notes' contains a table with three rows of data. The table has columns for 'ID', 'Message', and 'Created'. The data rows are: ID 1 with message 'wake up alarm' created at '1/2/2026, 1:32:00 PM'; ID 2 with message 'sleep alarm' created at '1/2/2026, 1:32:09 PM'; and ID 3 with message 'eat alarm' created at '1/2/2026, 1:32:17 PM'. At the bottom left of the dashboard, it says 'Backend: /api'. The browser's taskbar at the bottom shows various application icons and the system clock indicating 1:32 PM on 1/2/2026.

ID	Message	Created
1	wake up alarm	1/2/2026, 1:32:00 PM
2	sleep alarm	1/2/2026, 1:32:09 PM
3	eat alarm	1/2/2026, 1:32:17 PM

app.py - alfido-devops-labs [C... X animated-train-g474rj47pq3p... X animated-train-g474rj47pq3p... X +

animated-train-g474rj47pq3ppq6-5000.app.github.dev

Pretty-print

["message":"Hello from Dockerized Flask App! (updated)"]

6:46 PM 12/27/2025

app.py - alfido-devops-labs [C... X animated-train-g474rj47pq3p... X +

animated-train-g474rj47pq3ppq6.github.dev

alfido-devops-labs [Codespaces: animated train]

EXPLORER

ALFIDO-DEVOPS-LABS [C...> .github> core> docker-webapp> infra> logs> task1-dockerize> __pycache__> logs> ul> app.py 2. M> architecture> config.py> dashboard.py 4. U> dev-entrypoint.sh> docker-compos...> Dockerfile 1. M> executor.py 2. U> health.py> logger.py> main.py> metrics.py> op1.png> op2.png> op3.png> OUTLINE> TIMELINE

task1-dockerize > app.py > ...

28 def users():
38 conn.commit()
39 cur.execute("SELECT * FROM users")
40 rows = cur.fetchall()
41 cur.close()
42 conn.close()
43 return jsonify(rows)
44
45
46
47 if __name__ == "__main__":
48 # debug=True enables the built-in reloader for development
49 app.run(host="0.0.0.0", port=5000, debug=True)
50

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

Port	Forwarded Address	Running Process	Visibility	Origin
3126	https://animated-train...	/usr/bin/linux-x64/994fd128d3a5aa16f17d42...	Private	Auto Forwarded
3554	https://animated-train...	https://animated-train...	Private	Auto Forwarded
5000	https://animated-train...	/usr/local/bin/python -m flask run --host=0.0.0.0 ...	Private	Auto Forwarded
8501	https://anim...	/usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 ...	Private	Auto Forwarded

Add Port

ENABLING FILE WATCH IN DO...

http://localhost:5000/u
run this

Used 1 reference

curl -sS http://localhost:5000/user
s

Proceeding to call the /users endpoint and show the response.

Result:
[[{"Docker User"}]]

Describe what to build next

Agent Auto

Codespaces: animated train main Ln 46, Col 1 Spaces: 4 UTF-8 LF Python 3.12.1 Layout: US

6:51 PM 12/27/2025

