

User-Friendly Step-by-Step Guide

1. Create a local copy of the repository

Download or clone the project to your computer.

```
root@vbox:/home/usuario1/Documents/Docker# mkdir wellnes-ops-guide
root@vbox:/home/usuario1/Documents/Docker# cd wellnes-ops-guide/
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide# git clone https://github.com/luisrodvillada/wellnes-ops.git
Cloning into 'wellnes-ops'...
remote: Enumerating objects: 7664, done.
remote: Counting objects: 100% (7664/7664), done.
remote: Compressing objects: 100% (5650/5650), done.
remote: Total 7664 (delta 1727), reused 7509 (delta 1572), pack-reused 0 (from 0)
Receiving objects: 100% (7664/7664), 17.88 MiB | 18.07 MiB/s, done.
Resolving deltas: 100% (1727/1727), done.
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide#
```

2. Prepare the environment variables

Create the required `.env` file(s) with the configuration values needed for the application to run.

```
GNU nano 7.2
POSTGRES_DB=wellness
POSTGRES_USER=postgres
POSTGRES_PASSWORD=postgres
POSTGRES_PORT=5432
POSTGRES_HOST=postgres

JWT_SECRET=super-secret-key-change-me
JWT_EXPIRES_IN=1h
env/dev/.env *
```

3. Review the project structure

Check the folders and files included in the project to understand how the stack is organized

```
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide# ls -l
total 4
drwxr-xr-x 9 root root 4096 Jan  9 11:06 wellnes-ops →
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide# cd wellnes-ops/
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops# ls -l
total 40
drwxr-xr-x 4 root root 4096 Jan  9 11:06 backend
drwxr-xr-x 2 root root 4096 Jan  9 11:06 db
-rwxr-xr-x 1 root root 2849 Jan  9 11:06 docker-compose.dev.yml
-rwxr-xr-x 1 root root 1448 Jan  9 11:06 docker-compose.prod.yml
-rwxr-xr-x 1 root root 241 Jan  9 11:06 Dockerfile.dev
drwxr-xr-x 3 root root 4096 Jan  9 11:06 frontend
drwxr-xr-x 3 root root 4096 Jan  9 11:06 monitoring
drwxr-xr-x 3 root root 4096 Jan  9 11:06 nginx
-rwxr-xr-x 1 root root 2629 Jan  9 11:06 README.md →
```

4. Start the container stack

Run the Docker Compose command to build and start all services.

```
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops# docker compose -f docker-compose.dev.yml up -d --build
[+] Building 0.9s (28/28) FINISHED
 => [internal] load local bake definitions 0.0s
=> => reading from stdin 1.64kB 0.0s
=> [backend internal] load build definition from Dockerfile.dev 0.0s
=> => transferring dockerfile: 364B 0.0s
=> [nginx internal] load build definition from Dockerfile.dev 0.0s
=> => transferring dockerfile: 2848 0.0s
=> [frontend internal] load build definition from Dockerfile.dev 0.0s
=> => transferring dockerfile: 583B 0.0s
=> [backend internal] load metadata for docker.io/library/node:20-alpine 0.0s
=> [frontend internal] load metadata for docker.io/library/nginx:stable-alpine 0.0s
=> [backend internal] load .dockerrcignore 0.0s
=> => transferring context: 2B 0.0s
=> [frontend internal] load .dockerrcignore 0.0s
=> => transferring context: 2B 0.0s
=> [nginx internal] load .dockerrcignore 0.0s
=> => transferring context: 2B 0.0s
=> [nginx 1/3] FROM docker.io/library/nginx:stable-alpine@sha256:1ce7f79aab3e1d651dce1ff05b1bcd471cd0d5086838a77961feab6b49fb0b5 0.0s
=> => resolve docker.io/library/nginx:stable-alpine@sha256:1ce7f79aab3e1d651dce1ff05b1bcd471cd0d5086838a77961feab6b49fb0b5 0.0s
=> [frontend internal] load build context 0.0s
=> => transferring context: 2.91kB 0.0s
=> [backend 1/5] FROM docker.io/library/node:20-alpine@sha256:658d0f63e501824d6c23e06d4bh95c71e7f704537c9d9272f488ac03a370d448 0.0s
```

✓ wellnes-ops-frontend	Built	0.0s
✓ wellnes-ops-backend	Built	0.0s
✓ wellnes-ops-nginx	Built	0.0s
✓ Container wellnes-frontend-container	Started	0.0s
✓ Container wellnes-backend-container	Started	0.0s
✓ Container wellnes-nginx-proxy	Started	0.0s
✓ Container wellnes-postgres-db	Started	0.0s
✓ Container wellnes-prometheus	Started	0.0s
✓ Container wellnes-grafana	Started	0.0s

5. Verify that all services are healthy

Check the container status to ensure every service is running and marked as “healthy”.

```
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
c413465a3b4b        wellnes-ops-nginx   "/docker-entrypoint..."   About a minute ago   Up About a minute   0.0.0.0:80->80/tcp, [::]:80->80/tcp
s-nginx-proxy
bc800cbda75b        wellnes-ops-frontend   "/docker-entrypoint..."   About a minute ago   Up About a minute (healthy)  0.0.0.0:8080->80/tcp, [::]:8080->80/tcp
s-frontend-container
db50ac6194da        wellnes-ops-backend    "/docker-entrypoint..."   About a minute ago   Up About a minute (healthy)  3000/tcp
wellnes
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops# mkdir -p env/dev
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops# touch env/dev/.env
s-postgres-db
5d0aed6a1498        grafana/grafana:latest   "/run.sh"           18 hours ago       Up About a minute   0.0.0.0:3001->3000/tcp, [::]:3001->3000/tcp
s-grafana
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops#
```

6. Open the application in your browser

Go to:

<https://localhost/>

This loads the main frontend interface.



Login

Usuario

Contraseña

Entrar

7. Check the Backend container

Confirm that the backend service is running correctly inside Docker.

```
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops# docker logs wellness-backend-container
Backend running on port 3000 ←
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops#
```

8. Test the Backend endpoint through Nginx

Important: The backend is **not exposed directly**.

It is only accessible **through Nginx**.

```
/ # curl http://backend:3000/api/health
{"status": "OK"} / #
/ # curl http://backend:3000/health
{"status": "OK"} / #
/ # ping backend
PING backend (172.19.0.3): 56 data bytes
64 bytes from 172.19.0.3: seq=0 ttl=64 time=0.061 ms
64 bytes from 172.19.0.3: seq=1 ttl=64 time=0.134 ms
64 bytes from 172.19.0.3: seq=2 ttl=64 time=0.089 ms
^C
--- backend ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.061/0.094/0.134 ms
```

9. Check your host machine's IP address

Identify your local IP (for example: **192.168.1.11:3001**) to access services from another device if needed.

```
root@vbox:/home/usuario1/Documents/Docker/wellnes-ops-guide/wellnes-ops# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c2:d2:4b brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.11/24 brd 192.168.1.255 scope global dynamic noprefixroute enp0s3 ←
        valid_lft 73990sec preferred_lft 73990sec
    inet6 fe80::a00:27ff:fec2:d24b/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: tailscale0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1280 qdisc fq_codel state UNKNOWN group default qlen 500
    link/tailscale0 brd 0.0.0.0
```

10. Access Prometheus (port 9090)

Open:

http://localhost:9090

Prometheus collects metrics from the backend.

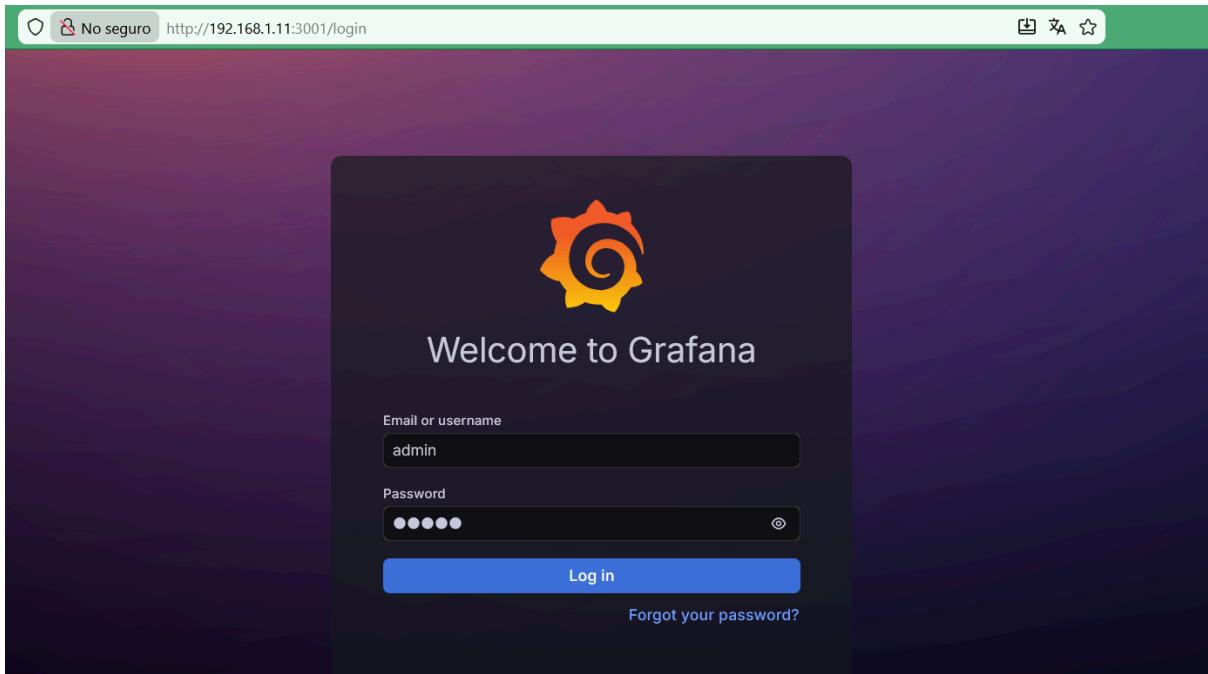
The screenshot shows the Prometheus web interface. At the top, the URL bar displays "http://192.168.1.11:9090/query". A red arrow points from the text above to this URL bar. Below the URL bar, there's a navigation menu with tabs for "Query", "Alerts", and "Status". The "Query" tab is active. A search bar below the menu contains the placeholder text "Enter expression (press Shift+Enter for newlines)". Underneath the search bar are three buttons: "Table", "Graph", and "Explain", with "Table" being the selected option. A "Evaluation time" dropdown is shown below these buttons. The main content area is currently empty, indicating no results for the query. At the bottom of the interface, there's another navigation bar with tabs for "Query", "Alerts", and "Status". The "Status" tab is active. It shows a table titled "backend" with one row. The table has columns for "Endpoint" (http://backend:3000/metrics), "Labels" (instance="backend:3000", job="backend"), "Last scrape" (8.49s ago), and "State" (UP). There are also dropdown menus for "Select scrape pool" and "Filter by target health".

12. Access Grafana

Open:

<http://localhost:3001>

Log in and view dashboards or create new ones.



13. View real-time logs

Use Docker logs to monitor all services in real time.

```
ion=132.59µs
wellness-grafana | logger=migrator t=2026-01-09T10:54:38.884559017Z level=info msg="Executing migration" id="managed folder permissions alert actions repeat
d fixed migration"
wellness-grafana | logger=migrator t=2026-01-09T10:54:38.884701897Z level=info msg="Migration successfully executed" id="managed folder permissions alert act
ions repeated fixed migration" duration=143.278µs
wellness-grafana | logger=migrator t=2026-01-09T10:54:38.885991148Z level=info msg="Executing migration" id="managed folder permissions library panel actions
migration"
wellness-grafana | logger=migrator t=2026-01-09T10:54:38.88615791Z level=info msg="Migration successfully executed" id="managed folder permissions library pa
nel actions migration" duration=167.27µs
wellness-grafana | logger=migrator t=2026-01-09T10:54:38.888286861Z level=info msg="Executing migration" id="migrate external alertmanagers to datasource"
wellness-grafana | logger=migrator t=2026-01-09T10:54:38.888424978Z level=info msg="Migration successfully executed" id="migrate external alertmanagers to da
```

14. Access the database and inspect tables

Connect to PostgreSQL and check that tables and records exist.

```
wellness=#  
wellness=# SELECT * FROM entries;  
 id |      title      |           description           |      created_at  
---+-----+-----+-----+-----+  
 1 | First entry    | Initial test entry          | 2026-01-08 16:13:28.66163  
 2 | Docker ready   | Database initialized via init.sql | 2026-01-08 16:13:28.66163  
 3 | Ejemplo en Español | Este es un ejemplo en Castellano | 2026-01-08 16:13:28.66163  
 4 | Nuevo          | cinco                      | 2026-01-08 16:21:58.6686  
(4 rows)  
wellness=#
```

15. Verify data persistence in the application

Create or modify data in the app and confirm that it remains stored in the database.

```
wellness=# SELECT * FROM entries;  
 id |      title      |           description           |      created_at  
---+-----+-----+-----+-----+  
 1 | First entry    | Initial test entry          | 2026-01-08 16:13:28.66163  
 2 | Docker ready   | Database initialized via init.sql | 2026-01-08 16:13:28.66163  
 3 | Ejemplo en Español | Este es un ejemplo en Castellano | 2026-01-08 16:13:28.66163  
 4 | Nuevo          | cinco                      | 2026-01-08 16:21:58.6686  
(4 rows)  
wellness=#
```

Wellness Journal

Latest Entries

Nuevo — cinco

First entry — Initial test entry

Docker ready — Database initialized via init.sql

Ejemplo en Español — Este es un ejemplo en Castellano

Add New Entry

Title

16. Confirm that the CI/CD pipeline runs automatically

Push changes to the repository and verify that the pipeline executes successfully.

<p>✓ Make project public Backend CI #25: Commit 1d37257 pushed by luisrodvillada</p>	main	1 minute ago	28s	...
<p>✓ docs: add DevOps-oriented README Backend CI #24: Commit 0dfdce pushed by luisrodvillada</p>	main	Today at 10:30 AM	30s	...
<p>✓ Remove env folder from tracking Backend CI #23: Commit 19f362f pushed by luisrodvillada</p>	main	Today at 10:21 AM	30s	...
<p>✓ quit env Backend CI #22: Commit b037078 pushed by luisrodvillada</p>	main	Today at 10:19 AM	39s	...
<p>✓ Quit node_modules Backend CI #21: Commit baa1e2e pushed by luisrodvillada</p>	main	Today at 10:14 AM	31s	...
<p>✓ Auth stable, CI created, Backend organized</p>	...	Jan 8, 5:25 PM GMT+1		