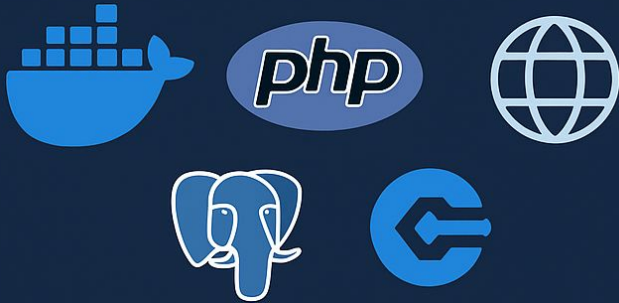


✓ Just Dockerized My Custom PHP REST API

From Local Friction to Clean & Reproducible Setup





Stack Overview

 PHP 8.3 (FPM)

 NGINX

 PostgreSQL

 .env config

 SQL auto import

 All running in Docker containers

Why I Dockerized It



No more manual setup

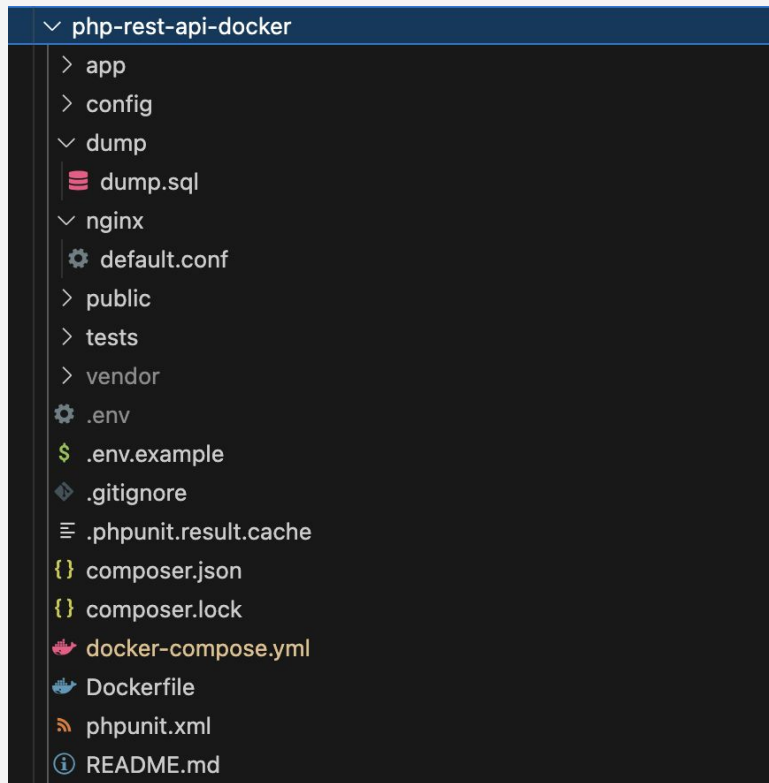


Same config everywhere (local, staging, prod)



Easy to deploy to cloud/VPS

Project Structure



Dockerfile

```
1 FROM php:8.3-fpm
2
3 WORKDIR /var/www
4
5 # Install dependencies
6 RUN apt-get update && apt-get install -y \
7     libpq-dev \
8     unzip \
9     zip \
10    git \
11    && docker-php-ext-install pdo pdo_pgsql
12
13 # Install Composer
14 COPY --from=composer:latest /usr/bin/composer /usr/bin/composer
15
16 # Copy project files
17 COPY . /var/www
18
19 # Install PHP dependencies
20 RUN composer install
21
22 # Set permissions (optional)
23 RUN chown -R www-data:www-data /var/www
24
25 EXPOSE 9000
26
```

docker-compose.yml Part I

```
1 version: '3.9' # Compose file format version
2
3 services:
4   app: # PHP application service
5     build:
6       context: . # Build context is the current directory
7       dockerfile: Dockerfile # Use the Dockerfile in the current directory
8     container_name: php-app # Set container name to 'php-app'
9     volumes:
10      - ../var/www # Mount current directory into container at /var/www
11     environment:
12       DB_HOST: ${DB_HOST} # Pass environment variable DB_HOST to container
13       DB_PORT: ${DB_PORT} # Pass environment variable DB_PORT to container
14       DB_NAME: ${DB_DATABASE} # Pass DB name from .env
15       DB_USER: ${DB_USERNAME} # Pass DB user from .env
16       DB_PASSWORD: ${DB_PASSWORD} # Pass DB password from .env
17     networks:
18      - appnet # Connect to the custom network 'appnet'
```

docker-compose.yml Part II

```
19
20 webserver: # NGINX web server service
21   image: nginx:alpine # Use lightweight Alpine-based NGINX image
22   container_name: nginx-app # Set container name to 'nginx-app'
23   ports:
24     - "${APP_PORT}:80" # Map host port to container port 80 (NGINX default)
25   volumes:
26     - ../var/www # Mount source code for NGINX to serve
27     - ../nginx/default.conf:/etc/nginx/conf.d/default.conf # Use custom NGINX config
28   depends_on:
29     - app # Start this service only after 'app' service is ready
30   networks:
31     - appnet # Connect to the custom network 'appnet'
32
```


docker-compose.yml Part III

```
32
33 db: # PostgreSQL database service
34     image: postgres:15 # Use official Postgres 15 image
35     container_name: postgres-db # Set container name to 'postgres-db'
36     environment:
37         POSTGRES_DB: ${DB_DATABASE} # Set initial database name from .env
38         POSTGRES_USER: ${DB_USERNAME} # Set initial DB username from .env
39         POSTGRES_PASSWORD: ${DB_PASSWORD} # Set initial DB password from .env
40     volumes:
41         - pgdata:/var/lib/postgresql/data # Persistent volume for Postgres data
42         - ./dump:/docker-entrypoint-initdb.d # Auto-import SQL files at container startup
43     networks:
44         - appnet # Connect to the custom network 'appnet'
45     ports:
46         - "5431:5432" # Map host port 5431 to container port 5432 (Postgres default)
47
```


docker-compose.yml Part IV

```
47
48 volumes:
49 | pgdata: # Define named volume 'pgdata' for persistent database storage
50
51 networks:
52 | appnet:
53 | | driver: bridge # Use default bridge network driver
54
```

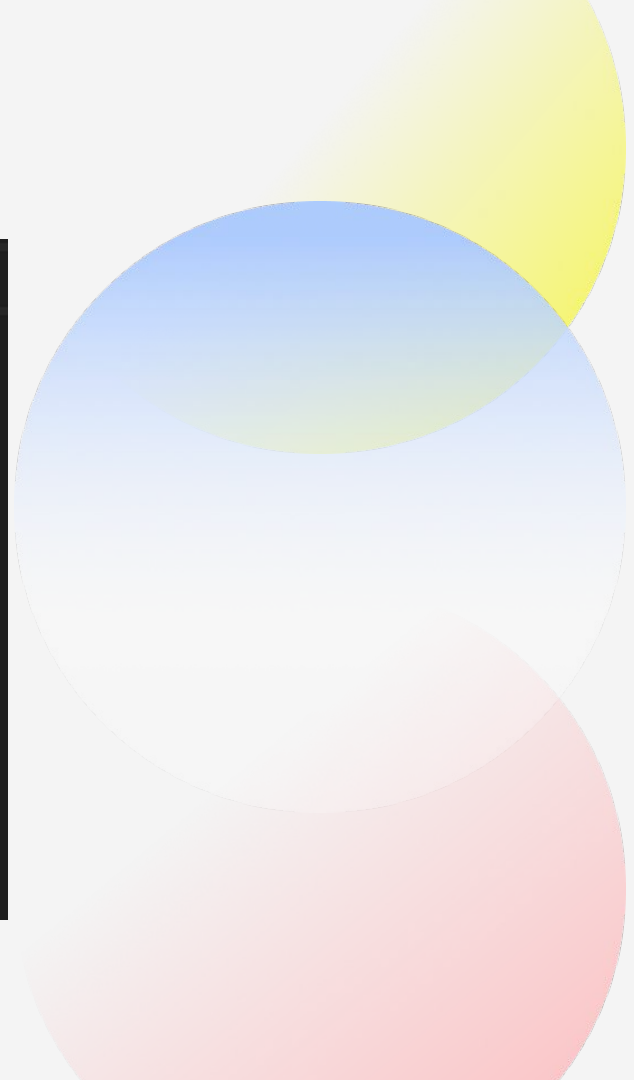
nginx configuration

```
php > php-rest-api-docker > nginx > default.conf
1  server {
2  |   # Listen on port 80 for HTTP requests
3  |   listen 80;
4  |
5  |   # Define the default files to serve when a directory is requested
6  |   index index.php index.html;
7  |
8  |   # Server name (domain) to respond to
9  |   server_name localhost;
10 |
11 |   # Root directory for the website files
12 |   root /var/www/public;
13 |
14 |   # Handle requests for the root and any other URI
15 |   location / {
16 |       # Try to serve the requested URI, a directory, or fallback to index.php
17 |       try_files $uri $uri/ /index.php?$query_string;
18 |   }
19 |
20 |   # Handle PHP file requests
21 |   location ~ /\.php$ {
22 |       # Include default FastCGI parameters
23 |       include fastcgi_params;
24 |
25 |       # Pass PHP requests to the container/service named "app" on port 9000
26 |       fastcgi_pass app:9000;
27 |
28 |       # Default index file for FastCGI
29 |       fastcgi_index index.php;
30 |
31 |       # Set the full path to the script to execute
32 |       fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
33 |   }
34 |
35 |   # Deny access to .htaccess and other hidden files
36 |   location ~ /\.ht {
37 |       deny all;
38 |   }
39 }
```



.env File

```
1  APP_PORT=8080|
2
3  DB_HOST=db
4  DB_PORT=5432
5  DB_PASSWORD=secret
6  DB_DATABASE=exercise
7  DB_USERNAME=postgres
8  DB_DRIVER=pgsql
```



One Command to Run All











```
○ ferripradana@Ferris-MacBook-Air php-rest-api-docker % docker-compose up --build -d
[+] Running 11/15
:: db 14 layers [#####] 17.26MB/107.9MB Pulling
✓ b3407f3b5b5b Pull complete
✓ 4a219808dd63 Pull complete
✓ 66b08a071c80 Pull complete
✓ 8396dda39ded Pull complete
✓ ea0fffb7e0d0 Pull complete
✓ 3c87731a2c16 Pull complete
✓ faf67cecec19 Pull complete
✓ 650bae63a44d Pull complete
:: 2e090fe828e7 Downloading [=====>] 17.26MB/1
✓ ee111489f815 Download complete
✓ ea3955338812 Download complete
✓ 1a7c42eef5da Download complete
:: c8f0d7f3882f Waiting
```

- ✓ Brings up PHP, NGINX, DB
- ✓ Auto-imports SQL
- ✓ Ready-to-code in seconds

Composer Install

```
○ ferripradana@Ferris-MacBook-Air php-rest-api-docker % docker exec -it php-app bash
root@249205cadd27:/var/www# composer install
Installing dependencies from lock file (including require-dev)
Verifying lock file contents can be installed on current platform.
Package operations: 55 installs, 0 updates, 0 removals
- Installing laravel/serializable-closure (v2.0.4): Extracting archive
- Installing voku/portable-ascii (2.0.3): Extracting archive
- Installing symfony/translation-contracts (v3.6.0): Extracting archive
- Installing symfony/polyfill-mbstring (v1.32.0): Extracting archive
- Installing symfony/deprecation-contracts (v3.6.0): Extracting archive
- Installing symfony/translation (v7.3.0): Extracting archive
- Installing symfony/polyfill-php83 (v1.32.0): Extracting archive
- Installing psr/clock (1.0.0): Extracting archive
- Installing symfony/clock (v7.3.0): Extracting archive
- Installing carbonphp/carbon-doctrine-types (3.2.0): Extracting archive
- Installing nesbot/carbon (3.10.1): Extracting archive
- Installing illuminate/macroable (v12.19.3): Extracting archive
- Installing psr/simple-cache (3.0.0): Extracting archive
```

Result (Docker Dashboard)

]  php-rest-api-docker		Running (3/3)	N/A	 
]  php-app 49619f5286be 		php-rest-ap Running	N/A	 
]  postgres-db df2c1d4f721d 		postgres:1! Running	N/A 5431:5432 	 
]  nginx-app 206fbd6e3ee5 		nginx:alpin Running	N/A 8080:80 	 

Result (Postgresql)

exercise - localhost:5431

Indexes databases

exercise

Schemas

public

- > Tables
- > Foreign Tables
- > Views
- > Materialized Views
- > Indexes
- > Functions
- > Sequences
- > Data types
- > Aggregate functions



....



<http://localhost:8080/login>

Authorization

Body ●

Scripts

Settings

☐ none

☐ form-data

☐ x-www-form-urlencoded

☐ binary

☐ GraphQL

JSON

```
1 {
2     "username": "ferri.pradana@gmail.com",
3     "password": "password"
4 }
```

Body

Cookies

Headers (9)

Test Results

200 OK

{ } JSON ✓

▶ Preview

Visualize

```
1 {
2     "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybWFnFtZSI6ImZlcnJpLnByYWRhbmFAZ2Z1haWwuY29tIiwiaXhwIjozNzUzNjcwOTkyfQ==.
        RyLX4Tv0xw7Q28Wi3MEhbEVBY9lHPTATWjmQFIKovg4=",
3     "exp": 1753670992
4 }
```

Result (Users API)

HTTP <http://localhost:8080/users>

GET <http://localhost:8080/users>

Params Authorization **Headers (7)** Body Scripts Settings

Headers 6 hidden

	Key	Value	Description
<input checked="" type="checkbox"/>	Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybWVtZS...	
	Key	Value	Description

Body Cookies Headers (9) Test Results ↺

{ } JSON ▾ ▶ Preview 🔄 Visualize ▾

```
1 {
2   "current_page": 1,
3   "data": [
4     {
5       "id": 1,
6       "name": "ferri",
7       "email": "ferri.pradana@gmail.com"
8     }
9   ],
10  "first_page_url": "/?page=1",
11  "from": 1,
12  "last_page": 1,
```

Conclusion / Final Thoughts



I started this just to simplify my local dev...



But it ended up teaching me about real-world deployment prep.



I'm not trying to preach — just sharing what worked for me.



Hope this helps anyone trying to build their own backend from scratch.