ChatGPT

Documentation du projet Fullstack Dockerisé : ToDo App Flask & React

---

Sommaire

1. Présentation du projet
2. Structure des répertoires
3. Configuration Docker
4. Détails sur les services
5. Reverse proxy avec NGINX
6. Base de données & migrations
7. Accès et routes du projet
8. Tests & vérifications

---

1. Présentation du projet

Application web de type "ToDo List" composée de : - Un **frontend** développé avec React + Vite - Un **backend** développé avec Flask (API REST + JWT + Flask-Smorest) - Une **base de données MySQL** avec PHPMyAdmin - Un reverse proxy **NGINX** - Un outil de monitoring : **cAdvisor**

Déployée dans des conteneurs via Docker & Docker Compose.

---

2. Structure des répertoires

```
📦todo-app-flask-reactjs
 ┣ backend
 ┣ frontend
 ┣ migrations
 ┣ docker-compose.yml
 ┣ nginx.conf
 ┗ .env
```

---

3. Configuration Docker

Le fichier `docker-compose.yml` contient les services suivants : - frontend - backend - db (MySQL) - phpmyadmin - nginx - cadvisor

Le `Dockerfile` du frontend effectue un build de React puis utilise `serve` pour le servir. Le backend utilise Gunicorn pour exécuter Flask.

---

4. Détails sur les services

• **Frontend** :
• Port : 3000 (interne)
• Accessible via NGINX sur le port 80

• Utilise Vite + React

• **Backend** :

• Port : 5000
• Framework : Flask avec Blueprints
• API REST versionnée (/api/v1/)

• Sécurisé avec JWT

• **MySQL** :

• Base de données `todoapp`
• Utilisateur : root / root

• Volume nommé `mysql-data`

• **PHPMyAdmin** :

• Accessible sur le port 8080

• Connecté à `db`

• **NGINX** :

• Reverse proxy entre frontend et backend
• Route `/` → frontend

• Route `/api/` → backend

• **cAdvisor** :

• Monitoring des conteneurs sur le port 8081

---

5. Reverse Proxy avec NGINX

Contenu de `nginx.conf` :

```
events {}

http {
    upstream frontend {
        server frontend:3000;
```

```
    }

    upstream backend {
        server backend:5000;
    }

    server {
        listen 80;

        location / {
            proxy_pass http://frontend;
        }

        location /api/ {
            proxy_pass http://backend;
        }
    }
}
```

---

6. Base de données & Migrations

Utilisation de Flask-Migrate :

```
# Depuis le conteneur backend
flask db init
flask db migrate -m "initial migration"
flask db upgrade
```

---

7. Accès et routes du projet

- **Frontend** : http://localhost
- **Backend** (via NGINX) : http://localhost/api/v1/
- **PHPMyAdmin** : http://localhost:8080
- **cAdvisor** : http://localhost:8081

---

8. Tests & vérifications

- Vérification du build du frontend :
- `npm run build` → OK → dist généré
- Vérification backend :
- `curl http://localhost/api/v1/users` → répond avec [] si vide
- Tests SQL :
- Table `users` bien créée avec Flask-Migrate
- Accès via navigateur → tout fonctionne via NGINX

✔ Projet complet et fonctionnel