Ecole Supérieure de Technologie de Salé
Université Mohammed V de Rabat

RAPPORT TECHNIQUE

2024- 2025

Développement Fullstack
CI/CD avec GitHub Actions

Réalisé par :

IBNITTOU Leila

sommaire

1. Présentation générale du projet

Objectif du projet

- Développer une application fullstack (React.js pour le frontend, Express.js pour le backend, MySQL pour la base de données).
- Conteneuriser l'application avec Docker pour faciliter le déploiement.
- Mettre en place un workflow CI/CD avec GitHub Actions pour automatiser les tests, la construction des images Docker et le déploiement.

Fonctionnalités principales de l'application :

- Afficher les utilisateurs : Liste des utilisateurs.
- Ajouter un utilisateur : Formulaire pour ajouter un nouvel utilisateur.
- Modifier un utilisateur : Formulaire pour modifier un utilisateur existant.
- Supprimer un utilisateur : Bouton pour supprimer un utilisateur.

Technologies utilisées :

- Backend : Express.js
- Frontend : React.js
- Base de données : MySQL
- Dockerisation : Docker et Docker Compose
- CI/CD : GitHub Actions

2. Étapes de mise en place du backend et frontend

- **Backend – Express.js :**

1. **Création de l'API Express :**

- Route GET /api/users : Récupère tous les utilisateurs.
- Route POST /api/users : Ajoute un nouvel utilisateur.
- Route PUT /api/users/:id : Modifie un utilisateur existant.
- Route DELETE /api/users/:id : Supprime un utilisateur.

2. Gestion des variables d'environnement :

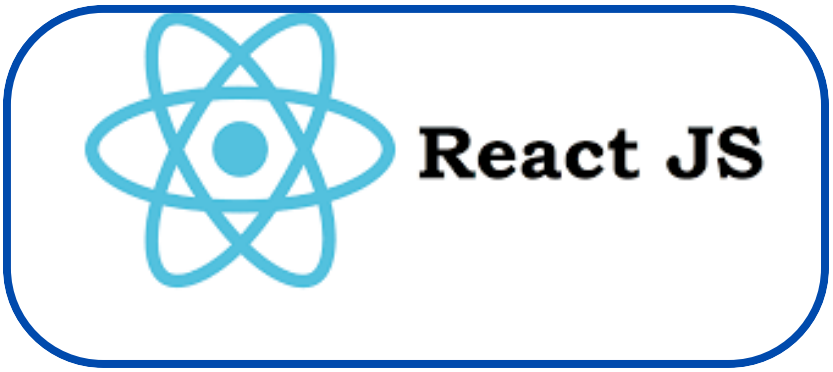
- Utilisation de ***dotenv*** pour gérer les informations sensibles (comme la connexion à la base de données).

3. Connexion à MySQL :

- Utilisation de Sequelize pour la gestion de la base de données.
- Base de données MySQL hébergée dans un conteneur Docker.

4. Gestion des erreurs :

- Ajout d'une gestion des erreurs pour assurer la robustesse de l'API.



3. Explication de la base de données

- **Structure de la base de données :**

1. **Table users :**

- id : Identifiant unique de type INT, auto-incrémenté.
- name : Nom de l'utilisateur, de type VARCHAR.
- email : Email de l'utilisateur, de type VARCHAR.

2. Connexion et gestion via Docker :

- Utilisation de Docker pour exécuter MySQL dans un conteneur et garantir l'isolation de l'environnement de base de données.
- La base de données est persistée à l'aide de volumes Docker pour éviter la perte des données lors du redémarrage des conteneurs.

3. Connexion à la base de données :

- Sequelize facilite les requêtes et les opérations sur la base de données.
- Gestion des données avec des migrations et des modèles pour une gestion souple de la base.

4. Dockerisation : étapes et choix faits

- **Dockerisation du Backend :**

1. **Dockerfile pour le backend :**

- Utilisation de l'image officielle Node.js.
- Installation des dépendances avec npm install.
- Lancement de l'application avec la commande node app.js.

2. Dockerisation du Frontend :

- Dockerfile utilise Node.js pour la construction du frontend React.
- Nginx est utilisé pour servir les fichiers statiques générés après le build React.

3. Docker Compose :

- Fichier docker-compose.yml utilisé pour orchestrer les conteneurs.
- Définition des services pour le backend, la base de données MySQL, et, optionnellement, le frontend.
- Configuration des variables d'environnement et des volumes pour la persistance des données.

5. GitHub Actions : pipeline expliqué étape par étape

1. **Fichier de workflow CI/CD :**

- Création d'un fichier ci.yml dans .github/workflows pour définir les étapes du pipeline CI/CD.

2. Étapes du pipeline :

- Checkout du code avec actions/checkout@v2.
- Installation de Node.js
- Installation des dépendances via npm install.
- Lancement des tests avec la commande npm test.
- Build de l'image Docker via docker build.
- Login Docker et configuration des secrets Docker.

3. Configuration des secrets GitHub :

- Ajout des secrets DOCKER_USERNAME et DOCKER_PASSWORD pour l'authentification à Docker Hub.

6. Difficultés rencontrées et solutions

- Problème : Échec d'authentification Docker Hub dans GitHub Actions.
- Solution : Configuration correcte des secrets GitHub (DOCKER_USERNAME et DOCKER_PASSWORD) et utilisation de docker/login-action@v2.
- Erreurs dans le pipeline CI/CD GitHub Actions :
- Problème : Échec des tests et construction de l'image Docker.
- Solution : Mise en place de services Docker dans GitHub Actions pour les tests et préparation de l'environnement avant l'exécution.
- Problèmes de gestion des variables d'environnement :
- Problème : Difficulté à gérer les variables d'environnement dans Docker et GitHub Actions.
- Solution : Utilisation de fichiers .env et configuration adéquate dans Docker et GitHub Actions.

6. Difficultés rencontrées et solutions

- Difficulté avec le déploiement automatique :
- Problème : Échec du déploiement automatique via SSH.
- Solution : Correction de la configuration des secrets et vérification de l'accès SSH pour le déploiement.
- Problèmes de performance de Docker et MySQL :
- Problème : Temps de démarrage long des conteneurs.
- Solution : Optimisation des Dockerfiles et réduction de la taille des images pour accélérer le processus.
- Problèmes de Dockerisation :
- Problème : Difficultés à configurer correctement les Dockerfiles pour le backend (Express) et le frontend (React).
- Solution : Optimisation des Dockerfiles en utilisant des images officielles et multi-stage builds pour réduire la taille des images et garantir un bon fonctionnement.

- Persistance de données dans MySQL (Docker) :
- Problème : Perte de données dans la base de données MySQL lors du redémarrage des conteneurs.
- Solution : Utilisation de volumes Docker dans le fichier docker-compose.yml pour assurer la persistance des données.
- Communication entre frontend (React) et backend (Express) :
- Problème : Erreurs CORS et mauvaises configurations des URLs.
- Solution : Mise en place du middleware CORS et ajout d'un proxy dans le frontend pour gérer les appels API.
- Problèmes avec les secrets GitHub Actions :

7. Conclusion et axes d'amélioration

- **Bilan du projet :**

- e projet a permis de créer une application complète Fullstack (Frontend React, Backend Express, MySQL) et de l'automatiser avec Docker et GitHub Actions.
- Un workflow CI/CD a été mis en place, automatisant les tests, le build des images Docker, et le push vers Docker Hub.

  - **Axes d'amélioration :**

1. **Tests unitaires :**

- Ajouter davantage de tests unitaires pour couvrir toutes les routes API et les composants React.
- Intégrer un outil de couverture de code comme Istanbul ou Jest pour garantir une couverture complète.

2.Déploiement automatique :

- Mettre en place un déploiement automatique sur un serveur (ex. via SSH sur un VPS ou un service comme Heroku ou AWS).