Documentation Technique - Projet de Chat Vidéo

Introduction

Description du Projet

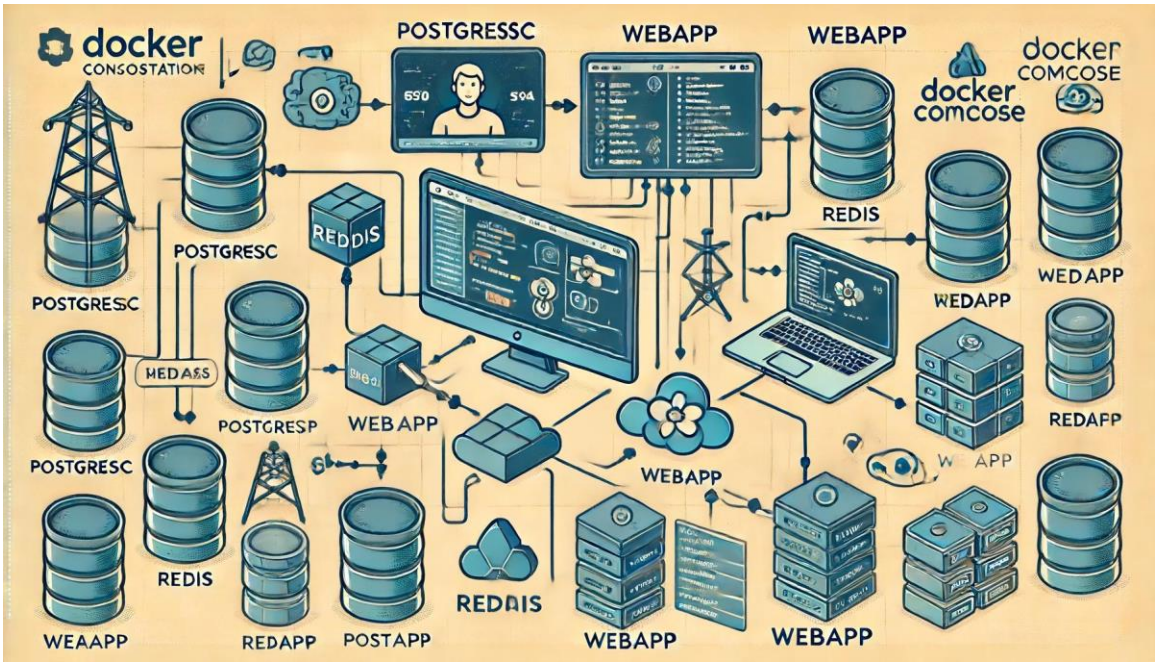
L'application de chat vidéo permet aux utilisateurs de communiquer via des vidéos en temps réel.
Elle utilise Docker pour la containerisation et inclut des services pour la gestion des utilisateurs, la communication en temps réel, et le stockage des messages.

Technologies Utilisées

- Docker
- PostgreSQL
- Redis
- Node.js

Architecture du Système

Diagramme d'Architecture



Description des Composants

1. PostgreSQL : Base de données pour stocker les données de l'application.
2. Redis : Stockage temporaire pour la gestion des sessions et des messages.
3. Webapp : Application web qui gère la logique de chat vidéo.
4. pgAdmin4 : Interface de gestion pour PostgreSQL.

Installation et Configuration

Prérequis

- Docker version 20.x ou supérieure
- Docker Compose version 1.27 ou supérieure
- 4 Go de RAM minimum
- 2 CPU minimum

Instructions d'Installation

1. Cloner le dépôt Git :

    Git clone https://github.com/BaptSoulier/chat-app.git

2. Construire et démarrer les conteneurs :

```
docker-compose up --build -d
```

Configuration Initiale

- PostgreSQL : Accessible via `localhost:5432` avec les identifiants configurés dans `.env`
- Redis : Accessible via `localhost:6379`
- Webapp : Accessible via `localhost:8000`
- pgAdmin4 : Accessible via `localhost:5050`

Développement

Structure du Code

1. logs/ : Contient les logs d'accès et d'erreurs.
2. my-httpd.conf/ : Fichiers de configuration HTTP.
3. web/ :
   - node_modules/ : Modules Node.js.
   - public/ : Fichiers statiques (HTML, CSS, JS).
   - Dockerfile : Instructions pour créer l'image Docker de l'application web.
   - docker-compose.yml : Fichier pour orchestrer les conteneurs Docker.
   - server.js : Point d'entrée principal de l'application web.

Standards de Codage

- Utiliser ESLint pour JavaScript.
- Suivre les conventions de style de code définies dans le projet.

Outils et Environnements

- IDE : Webstorm JetBrain
- Gestion de versions : Git
- Gestion des dépendances : npm

API Documentation

Endpoints

1. GET /api/messages : Récupérer tous les messages.
2. POST /api/messages : Envoyer un nouveau message.

Méthodes et Paramètres

- GET /api/messages :
  - Paramètres : Aucun
- POST /api/messages :
  - Paramètres :
    - content : Contenu du message (string)
    - sender : Expéditeur du message (string)

Exemples de Réponses

- GET /api/messages :
  ```json
  [
    {
      "id": 1,
      "content": "Hello",
      "sender": "Alice"
    },
    ...
  ]
  ```

Tests

Types de Tests

- Tests unitaires : Vérification des fonctions individuelles.
- Tests d'intégration : Vérification des interactions entre les composants.
- Tests end-to-end : Vérification du flux complet de l'application.

Outils de Test

- Jest : Pour les tests unitaires.
- Cypress : Pour les tests end-to-end.

Déploiement

Environnements de Déploiement

- Staging : Environnement de pré-production pour les tests.
- Production : Environnement de production pour les utilisateurs finaux.

Processus de Déploiement

1. Construire les images Docker :

```
docker-compose build
```

2. Démarrer les conteneurs :

```
docker-compose up -d
```

Maintenance et Support

Guide de Maintenance

- Vérifier régulièrement les logs pour détecter les anomalies.
- Mettre à jour les dépendances et les images Docker.

Dépannage et Résolution des Problèmes

- Consulter les logs d'erreur pour diagnostiquer les problèmes.
- Utiliser des outils comme pgAdmin4 pour vérifier l'état de la base de données.