Inhalt des Projekts

Der Ordner enthält den Quellcode für die wp-park, einen Bestandteil einer Bachelorarbeit. Das Projekt ist in verschiedene Ordner und Dateien gegliedert, die unter anderem die Einrichtung der Frontend-, Backend- und Datenbankvorgänge organisieren und erleichtern.

1 Struktur

1.1 Docker-Compose-Datei

Zweck: Wird zum Erstellen aller erforderlichen Images für die Anwendung verwendet.

Frontend- Image: Enthält Dateien zum Erstellen des Frontends der Anwendung.
Backend-Image: Enthält die erforderlichen Dateien für die Backend-Einrichtung.
DB-Image: Enthält Datenbank-Setup-Dateien.
FHIR- Image: Enthält die erforderlichen Dateien für die FHIR-Einrichtung.
Container: Containerkonfigurationen, die aus den Bildern erstellt wurden.
Volumes: Verwaltet Datenpersistenz und Speichereinstellungen.

1.2 env-Ordner

Zweck: Hostet die virtuelle Umgebung für Python und ermöglicht die Ausführung von Django.
Es wird verwendet, wenn das Projekt nicht auf dem Docker bereitgestellt.

1.3 MSD-Ordner (Backend)

Zweck: Dient als zentraler Backend-Ordner und hostet alle technischen Backend-Setups.

Requirements: Listet alle erforderlichen Python-Pakete und Abhängigkeiten auf.
Docker-Datei: Enthält die Anweisungen zum Erstellen des Backend-Docker-Images.
Intern MSD: Speichert wichtige Skripte wie Einstellungen und manage.py für die Projekteinrichtung und -vorbereitung.
Rest der Ordnern: Logik und Funktionalitäten, die für den Backend-Aspekt der Web-App wichtig sind.

1.4 MSDV-Ordner (Frontend)

Zweck: Für die Front-End-Entwicklung vorgesehen, wobei „V“ für Vue.js steht.

Requirements: Enthält die für die Frontend-Entwicklung erforderlichen Setup-Dateien und Pakete.
Vite-Setup: Enthält Konfigurations- und Setup-Dateien für Vite.
Tailwind CSS: Enthält Setup-Dateien und Konfigurationen für Tailwind CSS.
Package: Enthält Paketdateien, die für die Frontend-Entwicklung wichtig sind.
Docker-Datei: Enthält Anweisungen zum Erstellen eines Docker-Images, um Vue.js-Vorgänge zu erleichtern.

1.4.1 src-Ordner

Zweck: Enthält die gesamte Frontend-Logik und Ansichtseinstellungen.

App.vue: Die Stammkomponente, in die andere Komponenten eingefügt werden.
Assets: Speichert Mediendateien wie Bilder und Stylesheets.
Komponenten: Hostet verschiedene wiederverwendbare Komponenten, die im Frontend verwendet werden.
errors: Verwaltet Fehleranzeigen und -behandlung innerhalb der Anwendung.
Main.js: Der Einstiegspunkt der Vue.js-Anwendung.
Router: Verwaltet die Routing-Logik für das Frontend der Anwendung.
Setup: Enthält Setup-Dateien, die für die Frontend-Konfiguration wichtig sind.
Stores: Enthält die JavaScript-Logik, die zum Abrufen von Daten aus dem Backend/der Datenbank erforderlich ist.
views: Speichert Dateien, die verschiedene Ansichten im Frontend verwalten.

2 Getting Started

Für die Einrichtung und Ausführung der wp-park-Anwendung wird hauptsächlich Docker verwendet. Die folgenden Schritte beschreiben, wie das Projekt eingerichtet und ausgeführt werden kann:

2.1 Docker-Installation

Docker kann von der offiziellen Docker-Website heruntergeladen und installiert werden.

2.2 Navigation im Projektverzeichnis

Auf das Projektverzeichnis kann zugegriffen werden, indem ein Terminal geöffnet und der folgende Befehl verwendet wird:

- **cd path/to/your/project-directory**

2.3 Docker Images-Erstellung

Alle Docker-Images, wie in der Docker-Compose-Datei definiert, können durch Ausführen des folgenden Befehls im Terminal erstellt werden:

- **docker-compose build**

2.4 Ausführung von Docker-Containern

Die Docker-Container können mit dem folgenden Befehl initiiert werden:

- **docker-compose up**

2.5 Herunterfahren der Docker-Container

Die laufenden Container können mit dem folgenden Befehl im Terminal gestoppt werden:

- **docker-compose down**

kann über einen Webbrowser unter der angegebenen URL auf die Anwendung zugegriffen werden, typischerweise: Localhost:port

2.6 Ausführen von Migrationen (Django)

Bevor die Anwendung verwendet werden kann, müssen Migrationen in Django durchgeführt werden, um die erforderlichen Datenbankänderungen anzuwenden. Dies kann mit den folgenden Befehlen im Django-Container erfolgen:

- **Docker-Compose run Backend Python Manage.py Makemigrationen**
- **Docker-Compose run Backend Python Manage.py migrieren**

2.7 Neue Pakete installieren

2.7.1 Django

Neue Pakete, die für die Django-Anwendung erforderlich sind, können installiert werden, indem die Datei „requirements.txt“ im Django-Anwendungsverzeichnis geändert und anschließend das Docker-Image neu erstellt wird. So geht's:

- Die neuen Paketdetails in die Datei „requirements.txt“ einfügen.
- Das Docker-Image neu erstellen und die Container neustarten mit:
    - **Docker-Compose-Build**
    - **docker-compose up**

2.7.2 Vue

Für die Vue-Anwendung können neue Pakete installiert werden, indem die Datei package.json im Vue-Anwendungsverzeichnis geändert und anschließend das Docker-Image neu erstellt wird. Folge diesen Schritten:

- Die neuen Paketdetails in der Datei package.json hinzufügen.
- Das Docker-Image neu erstellen und die Container neustarten mit:
    - **Docker-Compose-Build**
    - **docker-compose up**