Dokumentacija projekta - Dockerizacija Flask-Vue-PostgreSQL aplikacije

Opis aplikacije i namjena

Ova aplikacija je jednostavni CRUD (Create, Read, Update, Delete) sistem za upravljanje knjigama. Omogućava korisnicima:

- pregled svih knjiga u bazi podataka
- dodavanje novih knjiga
- ažuriranje postojećih knjiga
- brisanje knjiga iz baze

Aplikacija služi kao primjer višeslojne web arhitekture implementirane kroz Docker kontejnere, demonstrirajući kako različite tehnologije (Vue.js, Flask, PostgreSQL) mogu raditi zajedno u kontejneriziranom okruženju.

Link GitHub repozitorija

Aplikacija je bazirana na: https://github.com/testdrivenio/flask-vue-crud/tree/main

Originalni repozitorij ne sadrži Docker konfiguraciju, koja je dodatno implementirana u sklopu ovog projekta.

Softverski preduvjeti za pokretanje

Za uspješno pokretanje aplikacije potrebno je:

- **Docker Engine** (verzija 20.10.0 ili novija)
- **Docker Compose** (verzija 1.29.0 ili novija)
- **Git** (za kloniranje repozitorija)
- **Operativni sistem**:
    - Linux (Ubuntu, Debian, Fedora, itd.)
    - macOS
    - Windows s WSL2 (Windows Subsystem for Linux)
- **Minimalno 2GB RAM-a**
- **Minimalno 2GB slobodnog prostora na disku**
- **Otvoreni portovi**: 8080, 5001 i 5432 na lokalnom računaru

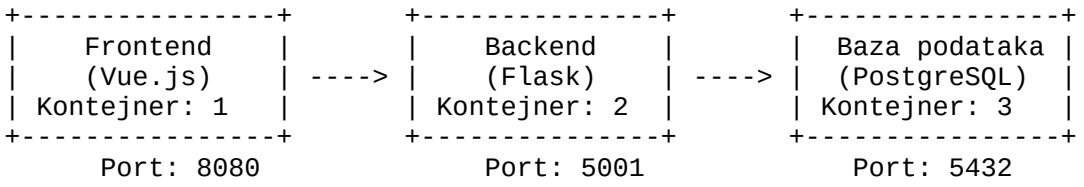
Arhitektura aplikacije

Aplikacija koristi klasičnu troslojnu arhitekturu:

1. **Frontend** (Vue.js)

    - Single Page Application (SPA)
    - Korisnički interfejs za interakciju s knjigama
    - Komunicira s backendom putem REST API-ja
2. **Backend** (Flask/Python)
    - REST API s endpointima za CRUD operacije
    - Poslovna logika aplikacije
    - Komunikacija s bazom podataka
3. **Baza podataka** (PostgreSQL)
    - Relacijska baza za trajnu pohranu podataka
    - Sadrži tabelu knjiga sa svim informacijama

Dijagram arhitekture:



Opis servisa, mreža i volumena

Servisi

1. **Frontend (client)**
    - **Tehnologija**: Vue.js (Vite dev server)
    - **Kontejner**: Baziran na node:14-alpine slici
    - **Port**: 8080:5173 (vanjski:unutarnji)
    - **Funkcije**: Grafički interfejs, komunikacija s API-jem
    - **Hot-reload**: Omogućen za razvoj
2. **Backend (server)**
    - **Tehnologija**: Flask (Python)
    - **Kontejner**: Baziran na python:3.7-slim slici
    - **Port**: 5001:5000 (vanjski:unutarnji)
    - **Funkcije**: REST API, komunikacija s bazom podataka
    - **Hot-reload**: Omogućen za razvoj
3. **Baza podataka (db)**
    - **Tehnologija**: PostgreSQL
    - **Kontejner**: Koristi službenu postgres:13-alpine sliku
    - **Port**: 5432:5432 (vanjski:unutarnji)
    - **Funkcija**: Trajna pohrana podataka

Mreža

- **app-network**: Docker bridge mreža
    - Povezuje sva tri kontejnera
    - Omogućava komunikaciju među servisima

- Izolira aplikaciju od ostalih sistema

Volumen

- **postgres_data**: Docker named volume
  - Mapiran na `/var/lib/postgresql/data/` u db kontejneru
  - Osigurava trajnu pohranu podataka iz baze
  - Podaci ostaju sačuvani i nakon zaustavljanja kontejnera

Uputstva za pokretanje

1. Priprema aplikacije

```
./pripremi_aplikaciju.sh
```

Ova skripta:

- Provjerava da li su Docker i Docker Compose instalirani
- Klonira GitHub repozitorij s izvornim kodom
- Kopira Docker konfiguraciju
- Kreira Docker mrežu i volumen
- Izgrađuje Docker slike

2. Pokretanje aplikacije

```
./pokreni_aplikaciju.sh
```

Ova skripta:

- Pokreće sve kontejnere u pozadini
- Provjerava status kontejnera
- Ispisuje informacije o pristupu aplikaciji

3. Zaustavljanje aplikacije

```
./zaustavi_aplikaciju.sh
```

Ova skripta:

- Zaustavlja sve kontejnere
- Čuva podatke (ne briše bazu)
- Omogućava kasnije ponovno pokretanje

4. Brisanje aplikacije

```
./obrisi_aplikaciju.sh
```

Ova skripta:

- Zaustavlja i briše sve kontejnere
- Briše sve Docker slike
- Briše volume s podacima
- Briše mrežu

- Vraća sistem u početno stanje

Kako pristupiti aplikaciji

Nakon pokretanja aplikacije, možete joj pristupiti na:

- **Frontend**: http://localhost:8080
- **Backend API**: http://localhost:5001/books
- **Ping test za backend**: http://localhost:5001/ping

Za direktan pristup bazi podataka:

```
docker exec -it flask-vue-crud_db_1 psql -U postgres -d books
```

Demonstracija funkcionalnosti

1. Perzistentnost podataka

- Dodajte nekoliko knjiga kroz frontend interfejs
- Zaustavite aplikaciju (`./zaustavi_aplikaciju.sh`)
- Ponovno pokrenite aplikaciju (`./pokreni_aplikaciju.sh`)
- Provjerite da su dodane knjige i dalje prisutne

2. Hot-reload

- Promijenite naslov u nekoj Vue komponenti dok aplikacija radi
- Primjetite kako se promjena automatski prikazuje u pregledniku
- Za backend, promijenite odgovor u `/ping` endpointu
- Pozovite endpoint i vidite novu poruku

3. Korištenje službene Docker slike

- Baza podataka koristi službenu PostgreSQL sliku s Docker Hub-a
- Provjera: `docker images | grep postgres`