📌 Dokumentasi API Flask & Integrasi React

📖 Deskripsi Proyek

Proyek ini merupakan API berbasis Flask yang terhubung dengan database PostgreSQL dan menyediakan endpoint untuk CRUD (Create, Read, Update, Delete) data item. API ini diintegrasikan dengan React sebagai frontend.

⚙️ Persiapan & Instalasi

📌 1. Clone Repository

```
git clone https://github.com/username/repo-name.git
cd repo-name
```

📌 2. Setup Virtual Environment (Opsional)

```
python -m venv venv
source venv/bin/activate  # Untuk Linux/Mac
venv\Scripts\activate  # Untuk Windows
```

📌 3. Install Dependencies

```
pip install flask flask-cors psycopg2
```

📌 4. Konfigurasi Database PostgreSQL

```
CREATE DATABASE cc;

CREATE TABLE items (
    id SERIAL PRIMARY KEY,
    name TEXT NOT NULL,
    description TEXT NOT NULL
);
```

📌 5. Menjalankan Server Flask

```
python app.py
```

API akan berjalan di http://localhost:5000.

🚀 Endpoint API

1️⃣ GET /api/items

Menampilkan semua item dari database

📌 Request:

```
GET /api/items
```

📌 Response:

```
[
  { "id": 1, "name": "contoh", "description": "lorem ipsum" },
  { "id": 2, "name": "contoh2", "description": "lorem ipsum2" }
]
```

2️⃣ POST /api/items

Menambahkan item baru ke database

📌 Request:

```
POST /api/items
Content-Type: application/json

{
  "name": "Mouse",
  "description": "Mouse gaming dengan DPI tinggi"
}
```

📌 Response:

```
{
  "id": 3,
  "name": "Mouse",
```

```
  "description": "Mouse gaming dengan DPI tinggi"
}
```

3 PUT /api/items/{id}

Mengupdate item berdasarkan ID

📌 Request:

```
PUT /api/items/1
Content-Type: application/json
{
  "name": "Laptop Updated",
  "description": "Laptop dengan spesifikasi terbaru"
}
```

📌 Response:

```
{
  "id": 1,
  "name": "Laptop Updated",
  "description": "Laptop dengan spesifikasi terbaru"
}
```

4 DELETE /api/items/{id}

Menghapus item berdasarkan ID

📌 Request:

```
DELETE /api/items/1
```

📌 Response:

```
{
  "message": "Item deleted successfully"
}
```

🎨 Frontend React

1. Install Dependencies

```
npm install
```

2. Jalankan Aplikasi

```
npm run dev
```

Dokumentasi Dockerfile

Docker digunakan untuk mengemas aplikasi agar dapat berjalan di berbagai lingkungan dengan konsistensi tinggi. Berikut adalah langkah-langkah untuk menggunakan Docker dalam proyek ini.

Dockerfile

Docker digunakan untuk mengemas aplikasi agar dapat berjalan di berbagai lingkungan dengan konsistensi tinggi. Berikut adalah langkah-langkah untuk menggunakan Docker dalam proyek ini.

Dockerfile Backend (Flask)

```
# Menggunakan base image Python
FROM python:3.9

# Menetapkan working directory dalam container
WORKDIR /app

# Menyalin file requirement.txt ke dalam container
COPY requirements.txt ./

# Menginstal dependensi
RUN pip install --no-cache-dir -r requirements.txt

# Menyalin semua file dari proyek ke dalam container
COPY . .

# Mengekspos port yang digunakan oleh aplikasi
EXPOSE 5000

# Menjalankan aplikasi Flask
CMD ["python", "app.py"]
```

Dockerfile Frontend (React + Vite)

```
# frontend/my-react-app/Dockerfile
FROM node:16-alpine

WORKDIR /app

COPY package*.json ./
RUN npm install

COPY . .

# Build untuk production menggunakan Vite
RUN npm run build

# Gunakan Nginx untuk serve static file
FROM nginx:stable-alpine
COPY --from=0 /app/dist /usr/share/nginx/html

EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

📌 Cara menggunakan Docker

1. Membangun image Docker

Fask

```
docker build -t flask_app .
```

React

```
docker build -t react-frontend-vite:1.0 .
```

2. Menjalankan container Docker

Flask

```
docker run -p 5000:5000 flask_app
```

React

```
docker run -d -p 3000:80 --name react-container-vite react-frontend-vite:1.0
```

3. Memeriksa status container Docker

```
docker ps
```

4. Menghentikan container Docker

```
docker stop flask_app react-container-vite
```

☑ Kesimpulan

1. Backend Flask digunakan untuk menyediakan API CRUD dengan database PostgreSQL.
2. Frontend React digunakan untuk menampilkan data dari API.
3. CORS sudah diaktifkan agar React bisa berkomunikasi dengan Flask.
4. API bisa diuji dengan Postman atau REST Client untuk memastikan fungsionalitas berjalan dengan baik.
5. Docker digunakan untuk memastikan aplikasi berjalan dalam lingkungan yang terisolasi.