Product Search Project Documentation

# 1. Project Overview

This project allows searching products using:  
- Text search (SentenceTransformer all-MiniLM-L6-v2)  
- Image search (CLIP model clip-ViT-B-32)  
- Products embedded & stored in ChromaDB  
- Backend APIs with FastAPI for Insert, Update, Delete, Search

# 2. Project Structure

AI\_PRODUCT\_SEARCH\_MAIN/  
 - backend.py # FastAPI backend (CRUD + Search APIs)  
 - embed\_to\_chroma.py # Embedding script  
 - chroma\_db/ # ChromaDB storage

# 3. Setup Instructions

1. Install Python 3.9+  
2. Create virtual environment:  
 python -m venv venv  
 source venv/bin/activate (Linux/Mac)  
 venv\Scripts\activate (Windows)  
  
3. Install dependencies (single command):  
 pip install fastapi uvicorn chromadb psycopg2 sentence-transformers pillow requests

# 4. Data Embedding

Run: python embed\_to\_chroma.py  
- Fetch products from PostgreSQL  
- Embed text & images  
- Store embeddings in ChromaDB collections

# 5. Backend API (FastAPI)

Run server:  
 uvicorn backend:app --reload  
  
Docs available at:  
 http://127.0.0.1:8000/docs  
 http://127.0.0.1:8000/redoc

# 6. Available APIs

Root: POST /  
Fetch products: POST /products (with offset & limit)

Text search: POST /search (with query payload)

Image search: POST /image-search (upload file)

Insert: POST /insert

Update: POST /update

Delete: POST /delete

## Payload Examples

Insert Payload Example:

{  
 "id": "12345",  
 "oem\_id": "OEM-111",  
 "name": "Test Product",  
 "description": "This is a test insert",  
 "images": "https://example.com/img.jpg",  
 "specifications": "Spec A, Spec B"  
}

Update Payload Example:

{  
 "id": "12345",  
 "name": "Updated Product",  
 "description": "Updated desc",  
 "images": "https://example.com/new.jpg",  
 "specifications": "Spec X, Spec Y"  
}

Delete Payload Example:

{  
 "id": "12345"  
}

# 7. Testing APIs

- Use Postman or curl  
- Insert/Update/Delete: Method POST → Body → raw → JSON  
- Image Search: Method POST → Body → form-data → file

# 8. Project Flow

1. PostgreSQL → embed\_to\_chroma.py → ChromaDB  
2. backend.py → FastAPI → CRUD + search  
3. Frontend (optional) → API calls → UI results