# Visual Product Matcher — Assigment Documentation

## Project Overview

**Visual Product Matcher** is a full-stack application that allows users to upload an image and find visually similar products from a dataset using a deep learning model.
The system is built with:

- **Frontend:** React + Vite + TailwindCSS
- **Backend:** FastAPI (Python) with TensorFlow (MobileNetV2)
- **Database:** MongoDB Atlas
- **Deployment:** Render (Backend) + Vercel (Frontend)

## Objective

To create an AI-powered image similarity application where users can:

- Upload product images
- Automatically extract embeddings using MobileNetV2
- Retrieve visually similar images based on computed embeddings
- Use secure authentication (Signup/Login) with JWT

## Technologies Used

| Layer | Tools & Frameworks |
|---|---|
| **Frontend** | **React.js, Vite, TailwindCSS, Axios, Framer Motion, React Hot Toast** |
| **Backend** | **FastAPI, TensorFlow (MobileNetV2), Pydantic, JWT, Cloudinary** |
| **Database** | **MongoDB Atlas** |
| **Deployment** | **Render (Backend) + Vercel (Frontend)** |
| **Version Control** | **GitHub** |

## Project Architecture

Workflow:

1. User logs in via the frontend (React)
2. Frontend communicates with the FastAPI backend via REST APIs
3. Images are uploaded to Cloudinary
4. The backend uses MobileNetV2 to extract embeddings
5. MongoDB stores image metadata and user details
6. The backend returns visually similar products to the frontend

## Project Architecture

```
visual-product-matcher/
│
├── backend/
│   ├── __pycache__/
│   ├── images/
│   ├── .env
│   ├── .gitignore
│   ├── auth.py
│   ├── auto_seed.py
│   ├── db.py
│   ├── main.py
│   ├── model.py
│   ├── rebuild_embeddings.py
│   ├── requirements.txt
│
├── frontend/
│   ├── node_modules/
│   ├── public/
│   ├── src/
│   │   ├── assets/
│   │   ├── components/
│   │   │   └── ProductCard.jsx
│   │   ├── pages/
│   │   │   ├── Home.jsx
│   │   │   ├── Landing.jsx
│   │   │   ├── Login.jsx
│   │   │   └── Signup.jsx
│   │   ├── api.js
│   │   ├── App.jsx
│   │   ├── main.jsx
│   ├── .env
│   ├── .gitignore
│   ├── eslint.config.js
│   ├── index.html
│   ├── package.json
│   ├── postcss.config.js
│   ├── tailwind.config.js
│   ├── vite.config.js
│   ├── vercel.json
│   └── README.md
│
└── README.md
```

## Backend Deployment

**Platform: Render**

**URL:**
https://visual-products-matcher.onrender.com/

**API Docs:**
https://visual-products-matcher.onrender.com/docs

**Command:**

*uvicorn main:app --host 0.0.0.0 --port {$(port)}*

## Frontend Deployment

**Platform: Vercel**

**URL:**
https://visuals-product-matcher.vercel.app/

**Build Command:**

*npm run build*

**Output Directory: dist**

**Environment Variable:**

*VITE_API_BASE_URL=https://visual-product-matcher-backend-33ks.onrender.com*

## Environmental Variables

**Backend .env**

*MONGO_URI=your_mongo_uri*
*JWT_SECRET=your_jwt_secret*
*CLOUDINARY_URL=your_cloudinary_api_url*

**Frontend .env**

*VITE_API_BASE_URL=*https://visual-products-matcher.onrender.com/

## Features

- ✅ User Authentication (JWT)
- ✅ Image Upload via Cloudinary
- ✅ Deep Learning-based Image Matching
- ✅ Responsive UI with TailwindCSS
- ✅ Toast Notifications and Animations
- ✅ Fully Deployed (Vercel + Render)

## Testing & Validation

- Verified /auth/signup and /auth/login endpoints using Postman.
- Confirmed backend availability via /docs.
- Checked frontend API integration with correct .env variables.
- Tested end-to-end image upload and retrieval process.

## Conclusion

The **Visual Product Matcher** successfully integrates AI-powered visual similarity detection with a modern web interface.
It demonstrates the ability to deploy and integrate **FastAPI, MongoDB, and React** into a cohesive, production-grade application.

## Submission Details

| Item | Description |
|---|---|
| **GitHub Repo Link** | https://github.com/Satyam216/visuals-product-matcher |
| **Branch** | main |
| **Deployment Links** | Backend (Render), Frontend (Vercel) |
| **Excluded Files** | node_modules/, .env, dist/, .vscode/ |
| **Project Size** | Within GitHub limits |