Project Assignment: Visual Similarity Search for Eyewear

1. Problem Statement

Traditional text search for fashion and accessories like eyewear often fails because users struggle to describe specific styles, frame shapes, or subtle textures in words. A user might have a picture of a celebrity wearing glasses or a photo of their old pair and want to find "something similar."

Your task is to design and implement an **AI-powered Visual Search Platform** that:

- Processes and indexes a catalog of eyewear images. Take images from the internet or lenskart for different frames and brands.
- Allows users to upload an image to find visually similar products.
- Identifies similarity based on attributes like color, frame style, material, and brand.
- Demonstrates production-grade system design, including image processing pipelines and vector search.

2. System Overview

The system must support:

- **Image Ingestion & Feature Extraction:** Extracting mathematical representations (embeddings) from catalog images.
- **Visual Query Processing:** Handling user-uploaded images and converting them into searchable vectors.
- **Multi-Attribute Similarity:** Ranking results based on style, color, and shape.
- **Scalable Vector Retrieval:** Efficiently searching through thousands of images.

3. Functional Requirements

3.1 Image Ingestion Pipeline

You must build a reusable pipeline that:

- Accepts images in standard formats (JPG, PNG).
- **Preprocessing:** Normalizes images (resizing, cropping, color correction).
- **Feature Extraction:** Uses a Deep Learning model (e.g., ResNet, Vision Transformer) to generate embeddings.
- **Storage:** * Stores metadata (brand, price, material) in a **Structured Database**.
    - Stores high-dimensional vectors in a **Vector Database** (e.g., Pinecone, Milvus, or FAISS).

3.2 Visual Search Engine

The search system must:

- Accept an image upload via easy ui or any other interface. Interface will not have more weightage in evaluation. Logic to extract similar specs will have more weightage.
- Perform a **Nearest Neighbor Search** to find the top-K most similar items. Or any other relevant logic.
- **Filter Logic:** Allow users to narrow down visual results by structured filters:
    - Price Range
    - Brand
    - Material (Acetate, Metal etc)
- Return a ranked list of products with a "Similarity Score."

3.3 Attribute Recognition (Mandatory AI Integration)

The system must demonstrate deeper AI understanding by implementing at least one of the following:

- **Automatic Tagging:** Using a classifier to label the uploaded image (e.g., "Aviator," "Wayfarer," "Transparent Frame", “Rim less”)

3.4 Feedback Loop (Mandatory)

The system must show how it could improve from user interaction:

- Track "Relevant/Not Relevant" clicks on search results.
- Implement a simple logic to "boost" products that are frequently clicked for specific visual styles.

4. Non-Functional Requirements

- **Architecture:** Clear separation between the AI Inference layer and the Data Storage layer.
- **Performance:** Search results should ideally be taking too long to give result.
- **Observability:** Basic logging for failed image uploads or high-latency queries.

5. Evaluation Criteria

Category	Weight
Search Accuracy & Visual Relevance	30%
System Architecture & Vector DB Usage	20%
AI Model Implementation (CNN/ViT)	20%
Code Quality & Modularity	15%
API Design & Documentation	15%

6. Deliverables

Each candidate must submit:

1. **Source Code:** A clean, documented repository.
2. **Architecture Diagram:** Showing the flow from image upload to vector retrieval.
3. **README:** Explaining the choice of model, the vector distance metric used (Cosine, Euclidean), and how to run the pipeline.
4. **Sample Dataset:** A small collection of images used for the demo.
5. Video explanation of the Demo (5-10 mins). Explain how you use AI to solve this.

7. Bonus (Optional)

- **Smart Cropping:** Automatically detecting the eyewear within a busy photo (e.g., a person's face) before searching.
- **Multi-Modal Search:** Allowing a user to upload an image *and* add a text modifier (e.g., Uploads a black frame + types "but in tortoise shell color")