Full-Stack Developer Technical Assessment

MVP: “Vibe Search” (Inspired by *Shoppin :* https://apps.apple.com/in/app/shoppin/id6738202299)

Objective

Build an **MVP version of Shoppin's "Vibe Search"** featuring multimodal search capabilities that combine visual similarity with contextual understanding.

The system should:

1. **Scrape fashion and lifestyle images** from Pinterest and Instagram
2. **Process and understand** both visual and textual contexts
3. **Cross-reference** against internal product database using hybrid search
4. Return **intelligently ranked results** based on visual similarity, semantic context, and user intent

---

What You'll Build

A sophisticated proof-of-concept pipeline with multimodal search capabilities, combining computer vision with natural language understanding.

Core Deliverables

1. Scraper

- Scrape images from Pinterest boards and Instagram pages (listed below)
- Extract **minimum 50–100 images** with metadata
- Capture:

    - Image URLs and source links
    - Post captions/descriptions
    - Hashtags and tags
    - Engagement metrics (likes, comments) - if available
    - Posted date
    - User/brand information
- Store in structured format (PostgreSQL/MongoDB)
- **Error handling**: Implement retry logic, rate limiting, rotating proxies/sessions
- **Bonus**: Scrape comments for additional context

2. Internal Catalog (Minimal Schema)

Product database with basic information:

```
product_id, title,category,brand_name, image_url, price
```

Example:

```
{
  "product_id": "NK-001",
  "title": "Nike Dunk Low Panda Black White Sneakers",
  "brand_name":"Nike",
  "category":"Shoes",
  "image_url": "https://...",
  "price": 110.00
}
```

Metadata Extraction Strategy:
Since you only have name/title, the system will **automatically extract** rich metadata:

A. From Product Titles (NLP Extraction)

Use NLP to parse titles and extract:

- **Brand**: "Nike", "Adidas", "Prada", etc.
- **Category**: "Sneakers", "Hoodie", "Shorts", "Watch", "Bag"
- **Style indicators**: "Minimal", "Luxury", "Streetwear", "Classic"
- **Product type**: "Low top", "High top", "Oversized", "Slim fit"

Example extractions:

```
"Nike Dunk Low Panda Black White Sneakers"
→ brand: Nike, category: Footwear/Sneakers, colors: [Black, White],
  style: [classic, streetwear], type: Low top

"Oversized Black Hoodie Streetwear"
→ brand: Unknown, category: Tops/Hoodie, colors: [Black],
  style: [streetwear, oversized], type: Hoodie

"Patagonia Baggies Shorts Blue"
→ brand: Patagonia, category: Bottoms/Shorts, colors: [Blue],
  style: [casual, outdoor], type: Shorts
```

B. From Product Images (Vision Model)

Extract visual features automatically:

- **Dominant colors**: RGB analysis + clustering
- **Item category**: Classify using CLIP zero-shot
- **Style attributes**: "formal", "casual", "sporty", "minimal"
- **Visual patterns**: solid, striped, patterned, logo-heavy

C. Generated Embeddings Store Everything

The CLIP embeddings inherently capture:

- Visual style and aesthetics
- Item type and category
- Color schemes
- Brand visual identity
- Use context (formal/casual/sports)

No manual tagging needed - the model learns these implicitly!

3. Multimodal Embedding & Hybrid Search

A. Visual Embeddings

- Generate embeddings using **CLIP, SigLIP, or OpenCLIP**
- Dimensions: 512 or 768
- Store in vector database (pgvector/Qdrant/Pinecone)

B. Text Embeddings

- Generate semantic embeddings for:
  - Product descriptions
  - Scraped captions
  - User search queries
- Use **Sentence-Transformers** (e.g., `all-MiniLM-L6-v2` or `multi-qa-mpnet-base`)
- Store alongside visual embeddings

C. Hybrid Search Implementation

Combine multiple search strategies:

1. **Visual-only search** (traditional image similarity)
2. **Text-contextual search** (semantic understanding)

Search modes:

```
{
  "mode": "visual",       # Image-to-image similarity only
  "mode": "contextual",   # Text query to product matching
}
```

4. Text Contextual Search System

Query Understanding & Auto-Enrichment

User Query: `"beach shorts"`

System Processing:

1. **Extract intent**: Looking for shorts suitable for beach
2. **Expand semantically**:
   - Synonyms: ["swim trunks", "board shorts", "summer shorts"]
   - Related: ["swimwear", "casual bottoms"]
3. **Search across**:
   - Product titles containing: "shorts", "swim", "beach", "trunk"
   - Image embeddings matching: beach/summer/casual vibes
   - Visual features: lightweight fabric appearance, bright colors, casual style

User Query: `"tracking shoes"` / `"trekking shoes"`

System Processing:

1. **Understand context**: Outdoor/hiking footwear
2. **Expand query**: ["hiking boots", "trail runners", "outdoor shoes", "trek shoes"]
3. **Match against**:
   - Titles with: "hike", "trail", "outdoor", "trek", "mountain"
   - Visual features: rugged design, outdoor aesthetic
   - Style embeddings: athletic, outdoor, functional

5. API Endpoints

A. Image-based Search

```
POST /api/search/image
{
  "external_image_url": "<https://image-link.jpg>",
  "top_k": 10,
  "filters": {
    "category": ["Footwear", "Accessories"],
    "price_range": [50, 300],
    "brands": ["Nike", "Adidas"],
    "colors": ["Black", "White"]
  },
  "rerank": true
}
```

Response:

```
{
  "query_analysis": {
    "detected_items": ["sneakers", "sunglasses"],
    "extracted_from_image": {
      "dominant_colors": ["white", "black"],
      "inferred_style": ["streetwear", "minimal"],
      "detected_category": "Footwear"
    }
  },
  "matches": [
    {
      "product_id": "NK-001",
      "name": "Nike Dunk Low Panda",
      "title": "Nike Dunk Low Panda Black White Sneakers",
      "extracted_metadata": {
        "brand": "Nike",
```

```
      "category": "Footwear/Sneakers",
      "colors": ["Black", "White"]
    },
    "visual_score": 0.89,
    "semantic_score": 0.82,
    "combined_score": 0.86,
    "price": 110.00,
    "image_url": "...",
    "match_reasons": [
      "Similar silhouette and shape",
      "Matching black/white colorway",
      "Similar streetwear aesthetic"
    ]
   }
  ],
  "total_results": 45,
  "search_time_ms": 234
}
```

B. Text Contextual Search

```
POST /api/search/text
{
  "query": "beach shorts for summer vacation",
  "top_k": 10,
  "filters": {
    "gender": "Male",
    "price_range": [20, 100]
  }
}
```

Response:

```
{
  "query_understanding": {
    "original_query": "beach shorts for summer vacation",
    "intent": "find casual summer bottoms",
    "extracted_keywords": ["beach", "shorts", "summer", "vacation"],
    "expanded_terms": ["swim trunks", "board shorts", "summer shorts", "swim
shorts", "beach wear"],
    "inferred_context": {
      "category": "Bottoms/Shorts",
      "use_case": "beach/vacation/summer",
      "style": "casual/relaxed"
    }
  },
  "matches": [
    {
      "product_id": "BR-045",
      "name": "Patagonia Baggies Shorts",
      "title": "Patagonia Baggies 5\\" Shorts Blue",
      "extracted_metadata": {
        "brand": "Patagonia",
        "category": "Bottoms/Shorts",
        "colors": ["Blue"]
      },
      "semantic_score": 0.94,
      "title_match_score": 0.87,
      "relevance_reasons": [
        "Title contains 'shorts' - exact category match",
        "Patagonia known for casual outdoor wear",
        "Visual style matches beach/vacation context",
        "Blue color common in summer/beach wear"
      ],
      "matched_terms": ["shorts", "baggies (casual style)"],
      "image_url": "...",
      "price": 65.00
    },
```

```
    {
      "product_id": "RL-089",
      "name": "Ralph Lauren Swim Trunks",
      "title": "Ralph Lauren Classic Fit Swim Trunks Navy",
      "extracted_metadata": {
        "brand": "Ralph Lauren",
        "category": "Bottoms/Swimwear",
        "colors": ["Navy"]
      },
      "semantic_score": 0.91,
      "title_match_score": 0.95,
      "relevance_reasons": [
        "Contains 'swim trunks' - direct synonym match",
        "Explicitly designed for beach/water",
        "Navy color appropriate for vacation wear"
      ],
      "matched_terms": ["swim trunks", "swim"],
      "image_url": "...",
      "price": 75.00
    }
  ],
  "search_strategy": "Hybrid: BM25 text search + semantic embeddings",
  "total_results": 23
}
```

6. Re-ranking & Intelligence Layer

A. Multi-stage Ranking

1. **First-stage retrieval**: Vector similarity (top 100)
2. **Second-stage reranking**:
    - Cross-encoder model for better relevance
    - Business rules (popularity, inventory, margins)

- Personalization signals (if applicable)

B. Score Fusion

```
final_score = (
    α * visual_similarity +
    β * semantic_similarity +
    γ * popularity_score +
    δ * price_relevance
)
```

Adjustable weights based on query type.

C. Diversity & Deduplication

- Ensure diverse results (avoid showing 10 similar items)
- Group by brand/style and show representative samples
- Remove near-duplicates

7. Frontend

Features:

- **Gallery View**: All scraped images in grid
- **Click to Search**: Click any image → show matches
- **Text Search Bar**: Natural language queries

UI Sections:

1. **Explore Feed**: Scraped images with metadata
2. **Search Interface**: Text + image upload
3. **Results Grid**: Products with match scores
4. **Detail Modal**: Product information + similar items

Enhanced Tech Stack

Backend

- **Framework**: FastAPI / Django (async support)
- **Database**: PostgreSQL 15+ with pgvector / pinecone

Frontend

- **Framework**: Next.js 14+ (App Router)
- **UI**: Tailwind CSS + shadcn/ui
- **State**: Zustand or React Query

Evaluation Criteria

Category	Points
Multimodal Search Quality	35
- Visual matching accuracy	15
- Text contextual understanding	20
Architecture & Code Quality	25
- System design	10
- Code clarity & organization	8
- Scalability considerations	7
API Design	15
Data Pipeline Robustness	15
Documentation	10
UI/UX & Demo	10
BONUS (+10 each)	
- Re-ranking implementation	+10
- Query expansion/understanding	+10
- Batch processing	+10
- Analytics dashboard	+10

Category	Points
- Deployment (Docker/cloud)	+10

Deliverables

Required

1. **GitHub Repository** with:
    - Clean, documented code
    - `docker-compose.yml` for easy setup
    - Environment configuration templates
2. **Comprehensive README**:
    - Architecture diagram
    - Setup instructions (< 5 commands)
    - API documentation
    - Model choices & rationale
    - Scraping strategy & challenges
    - Performance benchmarks
3. **Demo Video** (5 minutes):
    - Scraping process
    - Visual search demo
    - Text contextual search demo
    - Results explanation

Features (Stretch Goals)

1. Visual Question Answering

Query: "Show me shoes that would match this outfit"
+ Upload image of outfit
→ Return matching footwear

2. Style Transfer Search

"Find items with the same vibe but in different colors"

3. Negative Search

"Similar to this but NOT sneakers"

4. Budget Alternatives

"Show me similar items under $50"

5. Complete the Look

Input: Shoes
Output: Matching tops, bottoms, accessories

Timeline

7 days from task receipt

Reference Sources

Pinterest Boards

1. Minimal Streetwear
2. Men's Streetwear Outfit Ideas
3. Streetwear Outfit Ideas
4. Streetwear Fashion Instagram
5. Luxury Fashion – Roxx Inspire
6. Luxury Classy Outfits
7. Luxury Streetwear Brands

Instagram Pages

1. @minimalstreetstyle
2. @outfitgrid
3. @outfitpage
4. @mensfashionpost
5. @stadiumgoods
6. @flightclub
7. @hodinkee
8. @wristcheck
9. @purseblog
10. @sunglasshut
11. @rayban
12. @prada
13. @cartier
14. @thesolesupplier

Success Metrics

Technical KPIs

- Visual search accuracy: >80% top-5 hit rate
- Text search relevance: >85% user satisfaction
- API response time: <500ms (p95)
- Embedding generation: <100ms per image

Quality Metrics

- Semantic understanding accuracy
- Multi-attribute query handling
- Cross-category search capability
- Ranking diversity score

Submission Checklist

- [ ] Code repository with clear structure
- [ ] Docker setup working (one command to run)
- [ ] All 3 search modes implemented
- [ ] At least 50 scraped images with metadata
- [ ] API endpoints functional
- [ ] Frontend with all core features
- [ ] README with setup instructions
- [ ] Demo video showcasing key features
- [ ] Bonus features (if any) clearly marked
