**1. Mobile App Development Framework**

**Option 1: Flutter (Dart)**

* **Why?** Single codebase for iOS and Android, high-performance UI, fast development, and rich UI components.
* **Libraries:**
  + camera (for capturing wardrobe photos)
  + image\_picker (for selecting photos from gallery)
  + tflite\_flutter (for running AI models on the device)
  + firebase\_ml\_vision (for image recognition)

**Option 2: React Native (JavaScript/TypeScript)**

* **Why?** Popular among startups, rich ecosystem, and JavaScript-friendly.
* **Libraries:**
  + react-native-camera
  + react-native-vision-camera (for photo scanning)
  + TensorFlow.js (for AI-based image analysis)

**2. AI and Machine Learning (Wardrobe Recognition & Styling Suggestions)**

**Computer Vision for Wardrobe Analysis**

* **Google ML Kit** – Good for basic image recognition (Android/iOS)
* **OpenCV** – Helps with image processing (background removal, item cropping)
* **TensorFlow Lite / ONNX** – Custom AI model for clothing classification
* **YOLO (You Only Look Once) / ResNet** – Pre-trained models for fashion recognition

**AI-based Styling Suggestions**

* **Recommendation Engine:**
  + Collaborative filtering using **Amazon Personalize / TensorFlow Recommenders**
  + **Hugging Face models** for personalized fashion trends
* **GPT-powered Chatbot** for style tips

**3. Cloud Storage & Database**

* **Google Firebase (Firestore & Storage):** Stores user wardrobe data and images efficiently.
* **AWS S3 + DynamoDB:** Alternative for scalable wardrobe storage.
* **SQLite / Room DB (For Local Storage):** Stores wardrobe data offline.

**4. Image Processing & Background Removal**

* **Remove.bg API** – Automatically removes backgrounds for better wardrobe item recognition.
* **Adobe Sensei API** – For advanced fashion and texture recognition.
* **DeepLab (TensorFlow)** – For semantic segmentation of clothes.

**5. Backend for Data Processing & AI Model Training**

* **FastAPI (Python) / Node.js (Express.js)** – Backend for managing data and AI requests.
* **Django (Python)** – If you need a web interface with admin panel.
* **Flask (Python)** – Lightweight API for AI processing.

**6. User Preferences & Smart Suggestions**

* **Google Dialogflow / OpenAI GPT** – Chat-based outfit suggestions.
* **Rule-Based System:** If AI is not feasible initially, a basic rule-based suggestion system using metadata (e.g., category, weather, occasion) can be implemented.

**7. UX/UI Technologies**

* **Figma / Adobe XD** – UI/UX design.
* **Lottie (for animations)** – Used in Flutter/React Native.

**Example Tech Stack Combination**

**Frontend:** Flutter (Dart) / React Native  
**Backend:** FastAPI (Python) + Firebase Firestore  
**AI & Image Processing:** TensorFlow Lite, YOLO, OpenCV  
**Cloud Storage:** Google Firebase Storage / AWS S3  
**Authentication:** Firebase Auth (Google, Apple, Facebook login)

**Future Enhancements**

* **Augmented Reality (AR) Try-On:** Using ARKit (iOS) and ARCore (Android) for virtual dressing.
* **Blockchain (NFTs for Wardrobe Items):** Users can own and trade digital outfits.

1. **User Interface (Mobile App - Flutter/React Native)**
   * Photo capture & upload
   * Image gallery scanning
   * Virtual wardrobe management
   * Outfit recommendation UI
2. **Backend (FastAPI / Node.js)**
   * API for wardrobe item management
   * AI-based style suggestions
   * User preference storage
3. **AI/ML Layer**
   * Image recognition (YOLO, TensorFlow Lite)
   * Outfit recommendation engine (Deep Learning)
   * Fashion trend analysis (GPT-based assistant)
4. **Database & Cloud Storage**
   * Firebase Firestore / DynamoDB (user wardrobe metadata)
   * Firebase Storage / AWS S3 (image storage)
5. **Authentication & Security**
   * Firebase Auth (Google, Apple, Facebook)
   * Secure API calls with JWT tokens

Virtual Wardrobe App Architecture

