# Wardrobe Manager App - PRD (Free Tier Friendly)

## 1. Overview

The Wardrobe Manager App allows users to upload images of their clothing items (shirts, pants, jeans, shorts, etc.) and organize them virtually. When a new clothing image (e.g., a black shirt) is uploaded, the app identifies whether the user already owns a similar item, taking into account features like color, texture, and fabric type. It can also provide suggestions for similar items or complementary combinations.

## 2. Key Goals

1. Provide a free-tier friendly wardrobe management solution.  
2. Use AI (Gemini API) to describe and compare clothes.  
3. Offer user authentication via Google OAuth.  
4. Store images and metadata efficiently.  
5. Enable visual similarity comparison with percentage match.  
6. Host everything on free-tier or low-cost cloud infrastructure.

## 3. Architecture Overview

The system architecture consists of the following components:  
  
- \*\*Frontend\*\*: Next.js (deployed on Vercel free tier)  
- \*\*Backend\*\*: FastAPI (Python) or Node.js Express hosted on Vercel/Render  
- \*\*AI Service\*\*: Gemini API (for image captioning, similarity)  
- \*\*Database\*\*: Supabase (PostgreSQL + free storage)  
- \*\*Auth\*\*: Google OAuth via Supabase Auth  
- \*\*Storage\*\*: Supabase Storage or Firebase Storage (for images)

## 4. Core Features

- Upload and store clothing images with metadata (type, color, texture).  
- Automatically generate image descriptions using Gemini API.  
- Compare new uploads against existing wardrobe images using image embeddings.  
- Display similarity score and nearest match.  
- Provide outfit recommendations (optional future feature).  
- User authentication via Google OAuth.  
- Dashboard view of all wardrobe items.

## 5. Technical Stack

\*\*Frontend:\*\* Next.js + TailwindCSS  
\*\*Backend:\*\* FastAPI (Python) or Node.js Express  
\*\*Database:\*\* Supabase (PostgreSQL)  
\*\*Storage:\*\* Supabase Storage (free 1GB tier)  
\*\*Auth:\*\* Supabase Google OAuth  
\*\*AI Models:\*\* Gemini API (text + image embeddings)  
\*\*Hosting:\*\* Vercel (Frontend + Backend)

## 6. API Endpoints

1. \*\*POST /upload\*\* - Upload an image and metadata.  
2. \*\*GET /items\*\* - Get all wardrobe items for a user.  
3. \*\*GET /similarity/:id\*\* - Compare a new image with existing ones and return similarity scores.  
4. \*\*POST /describe\*\* - Call Gemini API to describe an image.  
5. \*\*GET /recommendations\*\* - Suggest similar or complementary items.

## 7. User Workflow

1. User signs in via Google OAuth.  
2. Uploads a new clothing image.  
3. Backend stores image and requests description from Gemini API.  
4. Image embedding is computed and stored.  
5. When a new image is uploaded, its embedding is compared against the user’s stored wardrobe.  
6. If similar item exists, show match percentage and item info.  
7. Display wardrobe grid view and suggestions.

## 8. Free-Tier Optimization

- Use Supabase for both database, storage, and auth (all free tier supported).  
- Deploy frontend + backend on Vercel.  
- Cache embeddings locally or in Supabase to avoid repeated Gemini API calls.  
- Limit image uploads to ~100 per user for free plan.  
- Compress images before upload.

## 9. Future Enhancements

- Outfit recommendation based on colors and styles.  
- Integration with e-commerce platforms to suggest missing wardrobe items.  
- Allow tagging (e.g., casual, formal, party wear).  
- Option for wardrobe sharing with friends.