Position: Full-Stack Engineer

Develop a scalable, AI-powered product listing platform using Next.js 15+, FastAPI, and Firebase, with a strong emphasis on real-time features and robust data management. If you have any questions email **nish@boxsy.io**

Project Overview

You'll build a full-stack application that enables users to:

1. **Securely upload** product images to Firebase Storage.
2. **Trigger AI processing** (via FastAPI) to generate product titles, descriptions, and potentially keywords based on the image.
3. **Implement comprehensive user authentication** and authorization with Firebase Authentication.
4. **Display product listings** in a rich, responsive UI.

Mandatory Requirements

1. **Authentication**: Firebase Authentication (Email/Password, Google Sign-In).
2. **Image Upload & Storage**: Firebase Storage with secure access rules.
3. **AI Integration**:
    - Backend (FastAPI) responsible for calling an AI API (e.g., OpenAI Vision API, or a similar image-to-text model).
    - FastAPI endpoint to receive image data, call AI, and return generated text.
4. **Database Integration**: Firebase Firestore for all product metadata and user data.
    - Utilize Firestore security rules for robust data access control.
    - Implement real-time listeners for updates (e.g., on the dashboard for listing status changes).
5. **Backend (FastAPI)**:
    - Handle image upload requests from Next.js, process with AI, and store metadata in Firestore.
    - Endpoints for creating, reading, updating, and (soft) deleting product listings.
    - Endpoints for admin actions (approving/rejecting listings).
    - Implement basic API key authentication/authorization for backend endpoints.
6. **Frontend (Next.js 15+, App Router)**:
    - Tailwind CSS, ShadCN for all styling, ensuring a clean and mobile-friendly UI.
    - Product Upload Form: Upload image, trigger AI generation, and save.
    - User Dashboard: Display only the authenticated user's listings with status (pending, approved, rejected).
    - Admin Dashboard: A separate view (or integrated with permissions) to see all pending listings and approve/reject them.
7. **Real-time Features**:

- Use Firestore real-time listeners for immediate updates on the user's dashboard when a listing's status changes.
- Possibly a "live" status indicator during AI processing.

Suggested Project Structure (Monorepo)

```
None
/backend (FastAPI app)
  /app
    main.py
    /routes
      auth.py
      products.py
      admin.py
    /services
      firebase_service.py
      ai_service.py
    /schemas
      product.py
      user.py
/frontend (Next.js 15+ app)
  /app
    /auth
      login.tsx
      register.tsx
    /dashboard
      page.tsx (User Listings)
    /upload
      page.tsx
    /admin
      page.tsx (Admin Listing Review)
  /components
    UploadForm.tsx
    ListingCard.tsx
    AdminReviewCard.tsx
  /lib
    firebase.ts
    api.ts (for FastAPI calls)
  /types
    index.ts
```

Submission Instructions

1. **Deploy the entire application**:

    - Frontend (Next.js): Vercel, Netlify, etc. Share a live link.
    - Backend (FastAPI): Render, Google Cloud Run, Vercel (for serverless functions if structured that way), etc. Share a live API endpoint URL.
2. **Host code on a public GitHub repo**:
    - Separate directories for `frontend` and `backend`.
    - **Do not commit `.env` values.** Use environment variables for API keys, Firebase config, etc.
3. **Record a Loom video (5-7 minutes, free plan)**:
    - Show user registration and login.
    - Demonstrate product image upload and AI generation.
    - Show the user's dashboard with their listings and real-time updates.
    - Log in as an admin user (explain how you set up admin role) and demonstrate approving/rejecting a listing.
    - Brief walkthrough of your codebase, highlighting key FastAPI services, Next.js components, and Firebase interactions (security rules, real-time listeners).
4. **Email all three links to**:
    - elisabeth@boxsy.io
    - nish@boxsy.io

Include:

- Loom video link
- GitHub repo link
- Deployed app link (Frontend)
- Deployed API link (Backend)

Evaluation Criteria

- **Comprehensive Firebase Usage**: Authentication, Firestore (real-time, security rules), Storage.
- **Robust FastAPI Backend**: API design, AI integration, data handling, error handling, authentication.
- **Next.js 15+ Frontend**: App Router, Tailwind CSS, ShadCN, clean UI/UX, responsive design, efficient data fetching/real-time updates.
- **AI Integration**: Seamless interaction between frontend, backend, and AI service.
- **Admin Functionality**: Clear implementation of admin role and listing review process.
- **Code Quality**: Readability, maintainability, proper use of TypeScript (if applicable), error handling, environment variables.
- **Deployment**: Successful deployment of both frontend and backend.
- **Video Demo**: Clear, concise, and thorough demonstration of all features.
- **Bonus**:
    - Error handling and loading states in UI.
    - More advanced Firestore queries/indexing.
    - Pagination for listings.

    - Input validation on both frontend and backend.

Final Checklist

- Next.js 15+ (App Router)
- FastAPI backend
- Firebase (Auth, Firestore, Storage)
- Tailwind CSS, ShadCN
- AI API (OpenAI Vision or similar)
- Public GitHub repo (no `.env` files)
- Public deployed app (Frontend)
- Public deployed API (Backend)
- Loom video demo (5 Mins)
- Email to elisabeth@boxsy.io + nish@boxsy.io