## ****TECH STACK****

## ****Frontend**** (Seller + Customer Interface)

**Framework:** React.js (with Create React App or Vite)

**UI Components:** Tailwind CSS for styling

Recharts for charts/analytics (Charts.js)

**Image Compression:** browser-image-compression

**Networking:** Axios for API calls

**Routing:** React Router DOM

**Extras:** QR code generation (if needed for simulations or links)

## ****Backend**** (App Logic + APIs)

**Platform:** Firebase

* **Firebase Authentication**
* **Firestore Database**
* **Firebase Storage**
* Firebase Cloud Functions

## ****AI Layer (Smart Returns Engine)****

**Framework:** Python + Flask API (deployed on Render / Railway)

**Modules:**

**Return Validation:** Uses Pillow / Google Cloud Vision API to detect damage, mismatched items, or fake claims.

**Return Prediction:** Scikit-learn (Random Forest) predicts likelihood of a return based on product type, price, customer history, etc.

**API Flow:**

Frontend sends order + image → Flask API → returns AI result { approved: true, confidence: 0.87, notes: "..." }

→ Result stored in Firestore under returns/{returnID}/aiValidation

**Storage & File Handling**

**Service:** Firebase Storage

**Policy / Optimizations:**

Compress images before upload (max width 1080px, quality 0.7).

Generate 200×200 thumbnails for quick display.

Limit videos to 15 seconds for MVP.

Automatically delete old files using Firebase lifecycle rules.

## ****Deployment****

**Frontend:** Vercel

**Backend:** Firebase

AI Microservice: Render or Railway