KW RiskDetector

Smart Claims Automation & Fraud Detection

What Problems Are We Solving?

- Manual document processing is slow and inconsistent
- High volume of unstructured documents (PDFs, receipts, invoices)
- Inefficient fraud detection methods
- Lack of real-time automation metrics

Our Solution

- OCR and NLP-powered document triage
- Confidence-based claim classification
- Fraud score generated by business rules + ML anomaly detection
- Automation Rate tracking with dashboard visibility

How It Works: Document Triage

- OCR extracts text from uploads
- NLP classifies: receipt, invoice, diagnostic, etc.
- Confidence score assigned using ML model
- Outcome: Auto-Approved or Manual Review based on rules

How It Works: Fraud Detection

- Rule engine flags issues (duplicates, date/amount mismatch)
- ML detects anomalies using clustering/isolation
- Final Fraud Score = rule weight + anomaly score
- Claims flagged above threshold are manually reviewed

Automation Rate Calculation

- Automation Rate = (Auto-Approved Claims / Total Claims) * 100
- Requirements for auto-approval:
  - OCR confidence ≥ 90%
  - Classification confidence ≥ 90%
  - No fraud flags
  - All key data extracted successfully

Expected Business Impact

- Reduce claim processing time from 5 days to < 24 hours
- Detect 20% more fraud from day one
- Achieve >75% Automation Rate
- Lower operational costs by 30%
- Improve NPS and customer retention

Technology Stack

- React + Tailwind for frontend
- FastAPI for backend
- OCR (AWS Textract), NLP (spaCy)
- ML models for classification & anomaly detection
- PostgreSQL + S3 for data storage

Deploy to Vercel

- Export full React app with Next.js or Vite
- Connect GitHub repo to Vercel
- Set project build as 'npm run build'
- Add API or mock endpoints if needed
- Publish and monitor performance from dashboard