Product Requirements Document (PRD)

1. Project Overview

This web application is designed to streamline the process of collecting insurance fact-find information from clients. It uses AI to guide users through a set of dynamic, admin-configurable questions, collects responses, generates a signed PDF, and integrates with email and Excel-based systems.

2. Goals & Objectives

- Digitize and automate the insurance fact-finding process.
- Enable dynamic, step-by-step data collection via a conversational AI assistant.
- Allow admins to customize questions and AI behavior without code changes.
- Generate a signed PDF summary of responses.
- Push structured data to email and Excel destinations.
- Ensure secure and mobile-first experience for customers.

3. User Roles

3.1 Admin

- Manage list and order of questions.
- Define question types (text, yes/no, multiple-choice).
- Set conditional logic for questions.
- Configure OpenAI prompt settings.
- Manage email and Excel integration.
- Monitor session submissions.

3.2 Customer

- Login via Clerk (email or magic link).
- Engage in a step-by-step question journey.
- Input responses via dynamic UI (text, toggle, etc.).
- Review and sign the final form.
- Receive confirmation or next-step guidance.

4. Core Features

4.1 Admin Dashboard

- Login (Clerk or separate admin access)
- CRUD operations for:
  - Questions
  - Answer types
  - Conditional logic
- Drag-and-drop reorder of questions
- Set system instructions for OpenAI prompt
- Configure:
  - Email template + recipient(s)
  - Excel sheet structure/mapping
- Monitor customer submissions (with timestamps)

4.2 Customer AI Dashboard

- Mobile-first login via Clerk
- Step-by-step assistant-style question interface
- Save/resume session
- Conditional branching based on prior answers
- Inline validations for each input

- Real-time autosave

4.3 Signature & Submission

- Final review screen
- Signature capture (canvas pad)
- Submit button:
  - Triggers PDF generation
  - Sends PDF to configured email(s)
  - Saves data to Excel-compatible format (local or cloud)

4.4 AI Assistant

- Integrate OpenAI GPT API (via system prompt + question context)
- Use prior answers to inform phrasing
- Ensure structured and predictable output
- Handle rephrasing, clarification requests

5. Technical Requirements

Frontend

- Framework: React + TypeScript
- Styling: TailwindCSS
- Auth: Clerk
- Components: shadcn/ui or Radix
- Signature: react-signature-canvas or HTML5 Canvas

Backend

- Runtime: Node.js (Express or tRPC)

- Database: PostgreSQL (Supabase or hosted)

- PDF Gen: Puppeteer / PDFKit

- Excel Gen: exceljs / xlsx

- Email: SendGrid or Postmark

- AI API: OpenAI GPT-4 or GPT-3.5

6. Integrations

- OpenAI (chat-like interface)

- Clerk (auth and user management)

- Email delivery (SendGrid, Postmark, SES)

- Excel (download or cloud push, e.g., Google Sheets API)

- Optional: Zapier/webhooks for additional workflows

7. Data Model Outline

Tables

- users (id, name, email, created_at)

- sessions (id, user_id, started_at, completed_at)

- questions (id, text, type, order, condition_logic)

- answers (id, session_id, question_id, value)

- config (ai_prompt, email_target, excel_template_url, etc.)

8. Security & Compliance

- Clerk handles secure login and session management.

- Data encrypted at rest and in transit.

- PDF & signature stored securely.

- GDPR and data retention support (e.g., auto-deletion after X days).

9. Risks & Assumptions

- AI output must remain structured - guardrails required.

- Users may abandon mid-flow - need auto-save and resume.

- PDF and Excel generation needs accurate formatting.

- OpenAI latency or rate-limits must be handled gracefully.