Alexandré Zii Miller

M.A.P.P.I.E.

Medical Archive Processor for Personal Information Extraction

Design Process Documentation

Table of Contents

---

1. Problem Definition & Tool Introduction

The Problem

As someone with a complex medical history spanning several conditions, several providers, and throughout several decades, I have found it impossible to personally keep my own health record organized in any cohesive fashion.

For example, presenting with a specific set of symptoms (joint hypermobility, chronic pain, frequent bruising, autonomic dysfunction, none of which fit neatly into any single specialty), I was bounced around between a dozen specialists over the course of

many years, (given that no specialist in particular was able to identify the root cause of my complaints).

Had this pattern been noticed sooner, I could have received adequate treatment for my hEDS and potentially prevented several long-term detriments to my health.

I have also noticed given the current landscape of unstructured medical records, with no single industry standard pipeline -- that critical health information can be overlooked, leading to undesirable health complications or delays.

Furthermore, many health records are not easily accessible to patients in both readability nor portability. Data is inconclusive, sparse, too technical, or presented in non-standard ways.

I don’t have a full list of all medications I’ve ever taken, yet this question is frequently asked when seeing a new provider. I don’t have a list of all my symptoms and side effects, or exact dates of operations and therapies, yet this information could be crucial when exploring other health avenues.

Key Points of Friction:

- Fragmentation of health records
- Inaccessibility of obtaining those records
- Obtained records may not be digitized
- Information overload, hard to find key points
- Challenging use of “medical jargon”
- Inconsistency across records
- Communication barriers (short apts, memory recall issues)
- Big picture patterns (no single provider sees-it-all)

M.A.P.P.I.E. intends to address the above scenarios, by providing patients with a tool to:

- Consolidate their fragmented medical records,
- Extract structured health data,
- Identify potential concerns through AI analysis,
- Generate “professional” summaries they can share with providers

(Transforming scattered chaos into organized, actionable information!)

2. Market Research & Context

Why Existing Solutions Fail

I evaluated existing tools and found significant gaps:

Patient Portals (MyChart, Epic, etc.)

- They show you your data if it is already digital and structured.
- They show a list of meds and a list of vaccines.
- They are dumb. They just store/display data.
- They do not read PDFs. They cannot process external data.

Manual Aggregators (PicnicHealth, Ciitizen, etc.)

- You must sign a power of attorney.
- They must fetch your records from all your doctors.
- Still dependent on you telling them where/what to look for.
- HUMANS manually sort all your data.
- Speed sucks. Takes weeks and months to process.

Provider AI Tools (Ambience, Abridge, Nuance DAX, etc.)

- Expensive tools.
- Can listen to conversation and take notes.
- They are not cross-platform, other providers can use other tools.
- These are for providers only. No insights to the patient.

General AI's (ChatGPT, Gemini, Claude, others)

- Free/Affordable.
- Can upload files, ask for a summary.
- Results are inconsistent, and need careful prompt engineering.
- You just get a wall of text or graphs that hallucinate data.
- File upload and context limits.
- No simple "drag & drop & done" functionality.

Initial Hypothesis…

What if there existed a tool that:

1. Is patient-centered/patient-first.
2. No registration, no app download, no subscription or paywall.
3. Considers complex/messy health records.
4. Support casual health data (fitness trackers, logging apps)
5. Identified points of friction, contradictions, & other potential issues.
6. Organizes everything by timeline and visuals.
7. Easy to understand terminology for those not super health literate.
8. Ability to reference and research more information.
9. Export results for a provider-friendly analysis to-go.
10. (Nearly) instant gratification.

...this would significantly improve the patient/provider experience.

User Research Approach

Primary Research Method: Autoethnography + Informal Interviews

[Describe what you actually did - even if informal]

- Analyzed my own experience with nearly 30 years of medical records.
- Spoke with friends & strangers about their experience managing their own health records, and friction points in the health industry.
- Observed patterns in effectively communicating health across providers. (Biggest point of friction across multiple users).

Key Insights:

1. People hate repeating themselves and sounding like broken records.
2. Both patients and providers complain how SLOW the healthcare industry is (seriously, FAX is still industry standard???).
3. So much data, so all over the place. A massive undertaking for any single individual to own their OWN HEALTH.
4. Even if health is in one place, it is challenging to organize a cohesive timeline or even understand medical terminology.
5. Because of all this scattering, there are too many nuanced health details that are missed, and a high risk of data being incorrect in places, which could be detrimental to future health outcomes.

6. Some existing patient-forward health tools are still too much friction to use. Too many requirements, too many standards, not actually UX friendly.

---

3. User Personas, Stories & Journeys

Persona 1: The Complex Case Patient

Background

Name: Alex Miller
Age: Almost 30 but feels 60
Occupation: Student
Medical Background: Complex "rare" genetic history and undiagnosed issues with overlapping symptoms between health concerns.
Tech Proficiency: Spent more time on the internet than with his IRL parents.

Current Situation

Healthy as a child with minimum complaints, health significantly deteriorated around adolescence and only has snowballed since then. Many ailments were unnoticed, not taken seriously, misdiagnosed or not diagnosed at all until permanent damage occurred.

Goals

- Quickly onboard new specialists without spending 45 minutes explaining entire medical history from scratch
- Identify medication interactions before they cause problems (has happened 400 times already)
- Generate a clean, professional summary to submit with insurance claims or bring to one-off providers/institutions without revealing unnecessary personal details

Frustrations

- I have rare/undiagnosed illnesses that most people do not fully understand. HEDS as an example, is notable in how it intersects with any other health issue.
- I cannot summarize 30 years of my health history in under 30 minutes to anyone, let alone a medical professional that needs a comprehensive picture.
- The above issues repeat any time I travel, switch insurance, or get ping-ponged over to another provider because the current one doesn't want to deal with my complex case anymore.

A Day in the Life

"I was in Poland and ran out of my medicine. I bounced around 11 specialists before ending up in the ER because no one could fill my meds as a foreigner. I had a high fever and waited in the emergency room for 14 hours. I slept on the floor because I was in so much pain and couldn't sit in my wheelchair waiting anymore."

Persona 2: Health Nut

Background

Name: Jack Jacked
Age: 34 but feels 24
Occupation: Triathlete
Medical Background: No health concerns, minor cardiovascular family history.
Tech Proficiency: Early adopter, owns Oura ring, Apple Watch, Whoop strap, uses 6 health apps

Current Situation

Tracks everything; sleep, HRV, VO2 max, resting heart rate, recovery scores, nutrition macros. Has health data scattered across Oura, Apple Health, Strava, MyFitnessPal, and annual physical results in the patient portal. Wants to see the big picture but can't connect wearable data with clinical lab results.

Goals

- Correlate clinical lab results (cholesterol, testosterone, vitamin D) with wearable biometrics to optimize performance

- Track trends over multiple years to see if training changes actually improve health markers
- Share comprehensive health data with sports medicine doctor without manually combining sources

Frustrations

- *"My doctor ordered labs in March and again in September. I know my training changed between those dates, but I can't see my resting heart rate trend from my Oura ring alongside my cholesterol levels. They're in completely different systems."*
- *"I have five years of annual physical results showing my HDL improving, but it's just PDFs in a portal. I want to see it graphed against my training volume from Strava. That's the actual story."*
- *"When I switched to a new sports medicine clinic, they asked about previous injuries. I have all the MRI reports and PT notes, but they're scattered across three different orthopedic offices. I spent an hour digging through emails to find them all."*

A Day in the Life

"I got my annual physical results back last week—everything looked good, but my ferritin was borderline low. I wanted to see if this was a trend or a one-time thing. I logged into four different patient portals going back three years, downloaded PDFs, and tried to find ferritin in each report. Some called it 'serum ferritin,' others just 'ferritin,' one abbreviated it. I made a spreadsheet manually. Took me 90 minutes. Then I wanted to see if it correlated with my training load from Strava during those periods—gave up. If I could dump all my labs and wearable data into one system and just ask: show me ferritin trends vs training volume, that would save me hours every time I want to optimize something."

Persona 3: Data Enthusiast

Background

Name: Name: Dr. Sarah Kim (PhD, not MD)
Age: 38
Occupation: Biostatistician
Medical Background: Healthy, type 1 diabetes (well-managed), regular

checkups
Tech Proficiency: Builds her own data visualization tools, codes in R and Python daily

Current Situation

Fascinated by her own health data as a dataset. Tracks blood glucose with CGM, but is frustrated that her endocrinologist appointments feel like "vibes-based medicine" instead of data-driven decisions. Has 5 years of A1C results, CGM exports, and diet logs but can't easily combine them.

Goals

- Extract structured data from clinical notes to analyze treatment effectiveness over time
- Build visualizations showing A1C trends, insulin dosage changes, and diet patterns together
- Export clean CSV files from medical records to feed into her own statistical models

Frustrations

- *"My doctor said 'your control looks good' but I want to quantify that. My A1C dropped from 7.1 to 6.4—was that the new insulin regimen or my dietary changes? The data is trapped in clinical notes written in prose."*
- *"I have years of CGM data from Dexcom, but my lab A1C results are PDFs in a patient portal. I can't programmatically compare them. This should be trivial."*
- *"Every time I try to extract data from medical records, I'm fighting with PDFs, OCR errors, and inconsistent formatting. If I had structured JSON output, I could do real analysis in minutes."*

A Day in the Life

"I had my quarterly endo appointment yesterday. My doctor reviewed my A1C and said it looked stable. I asked if we could look at the trend over the last two years alongside my insulin dosage changes and diet logs. She said the EHR doesn't let her view it that way—everything is in separate tabs. I went home and spent three hours manually entering data from five PDFs into a spreadsheet so I could make a graph. I'm a statistician—I should be able to feed my medical records into R and

get publication-quality visualizations instantly. Instead, I'm doing data entry like it's 1995."

Persona 4: The Privacy-Conscious Professional

Background

Name: David Chang
Age: 45
Occupation: Corporate Attorney
Medical Background: Healthy, history of mental health treatment (depression), regular preventive care
Tech Proficiency: Very tech-savvy, runs VPN 24/7, uses encrypted email

Current Situation

Sees a therapist, takes antidepressants, had inpatient psychiatric treatment 10 years ago. Deeply concerned about stigma if mental health history leaks. Works in conservative industries where this could affect partnership track. Terrified of cloud-based patient portals being breached. Wants control over his own records.

Goals

- Store complete medical records locally (not in cloud) with encryption
- Selectively share only relevant portions of history (e.g., only surgical history for anesthesia consult, hiding psychiatric records)
- Generate clean summaries that omit sensitive information without lying or creating gaps

Frustrations

- *"Every patient portal has my psychiatric history sitting in the cloud. I don't trust these systems—healthcare has major data breaches constantly. I want my records on my own encrypted drive."*
- *"When I had a colonoscopy, the anesthesiologist asked about my medications. I take an SSRI. I had to disclose it, but then it's*

in those records forever. I wish I could provide medication info without it being permanently associated with 'depression.'"

- *"I requested my complete medical records from my old psychiatrist. They charged me $200 for 400 pages of photocopies. I can't search for them, can't edit them, can't selectively share parts. They're just paper in a file cabinet. This should be digital and under my control."*

A Day in the Life

"I'm interviewing for partner at a new firm. They require a pre-employment physical and drug screen. Standard stuff. But the intake form asks 'have you ever been hospitalized for mental health treatment?' I have—10 years ago for depression, fully recovered, on maintenance meds. If I say yes, do I lose the job? If I say no, am I lying on official paperwork? I called the occupational health clinic to ask if I could just provide relevant medical records (proving I'm healthy now) without disclosing the hospitalization. They said they need 'complete history.' I'm stuck between honesty and career suicide. If I had a tool that let me generate a verified health summary showing I'm currently healthy, stable on medications, and cleared for full-time work—without including the hospitalization—I could satisfy their requirements without disclosing something that's nobody's business."

Persona 5: The Hypochondriac

Background

Name: Jessica Torres
Age: 21
Occupation: Marketing
Medical Background: Generally healthy, anxiety disorder, frequent urgent care visits
Tech Proficiency: Comfortable with apps, googles symptoms extensively (too much)

Current Situation

High health anxiety. Every symptom feels catastrophic. Visits urgent care 8-10 times per year for chest pain (always anxiety), abdominal pain (always IBS), headaches (always tension). Has seen 12 different

providers in the last 18 months. Each time feels like starting over. Wants to show providers her history so they take her seriously, but also fears being labeled as "that patient."

Goals

- Consolidate all past visits to show providers she's been worked up thoroughly (not just anxious and untested)
- Quickly share test results proving symptoms aren't new or unexplained when anxiety strikes
- Reduce repeat testing by showing recent labs/imaging were already normal

Frustrations

- *"I went to urgent care last month with chest pain. They treated me like I was wasting their time until I mentioned I'd had an EKG three weeks ago that was normal. Then they actually listened. But I didn't have proof with me—it was at a different clinic."*
- *"Every new doctor asks about family history and medication allergies. I've answered these questions 50 times. I just want to hand them a document and say 'here's everything.'"*
- *"My therapist says I need to stop going to urgent care for every symptom, but what if it's real this time? If I had all my records showing consistent patterns, maybe I'd feel more confident distinguishing anxiety from actual problems."*

A Day in the Life

"I woke up with my left arm tingling and immediately panicked—heart attack? I drove to urgent care at 6am. The doctor asked if I'd had cardiac workup before. I said yes—EKG, stress test, echocardiogram, all normal, but I couldn't remember when or where. They repeated the EKG (normal, $500). Later I checked my records. I'd had one five weeks ago at a different urgent care, also normal. If I'd had all my cardiac workup history on my phone, I could have shown them and maybe avoided the repeat test. Or at least felt reassured without the $500 bill."

Persona 6: The Caretaker

Background

Name: Linda W.
Age: 57
Occupation: High School Teacher
Medical Background: Healthy personally, managing care for mother (83) with dementia and father (85) with heart failure
Tech Proficiency: Uses basics (email, texting), overwhelmed by too many portals

Current Situation

Primary caregiver for both parents. Dad sees cardiologist, primary care, nephrologist. Mom sees neurologist, psychiatrist, primary care. Between them: 14 medications, 8 specialists, appointments twice a week. Linda has login credentials for 6 different patient portals. Constantly asked to provide a "complete medication list" at appointments and can never remember all 14.

Goals

- Maintain single source of truth for both parents' medications (keeps preventing dangerous interactions)
- Quickly generate summaries when taking parents to new specialists or ER
- Track which parent saw which doctor when (appointments blur together)

Frustrations

- *"Last month Dad went to the ER for shortness of breath. They asked what medications he takes. I knew most of them but blanked on two. The ER doctor said one of the ones I forgot was probably causing the problem. If I'd had a complete list on my phone, we could have figured it out hours earlier."*
- *"I have six different patient portal logins for my parents. Half the time I can't remember which parent sees which doctor at which hospital. I've shown up to the wrong building twice because I got their appointments confused."*
- *"When Mom's neurologist asked if her confusion was worse since starting the new medication, I honestly didn't know. That was three months ago and I see her every day. It's hard to notice gradual changes. If I had a timeline of medication changes and doctor notes, I could actually answer those questions."*

A Day in the Life

"I took Mom to a new psychiatrist yesterday because her old one retired. The intake form asked for complete medical history, current medications, and previous psychiatric treatments. I had some of it in my head, some written on scraps of paper in my purse, and some buried in three different patient portals I couldn't access from the waiting room. The doctor kept asking followup questions—'When did she start the Aricept? What dose? Why did they discontinue the other one?' and I didn't have answers. I felt like a bad daughter. I'm doing my best, but I can't keep track of everything for two people across a dozen doctors. If I could just pull up a single document with both parents' complete histories, I wouldn't feel like I'm constantly failing them."

Persona 7: The Frequent Flier

[This can be more of a composite, but still grounded in real observations]

Background

Name: Raj Patel
Age: 42
Occupation: International Business Consultant
Medical Background: CControlled hypertension, chronic back pain from old injury, travels 200+ days/year
Tech Proficiency: Uses apps for everything, lives out of cloud storage

Current Situation

Spends 8 months per year abroad (Asia, Europe, South America). Takes daily medications, occasional flare-ups require urgent care in foreign countries. Has had emergencies in Tokyo, Berlin, and São Paulo where language barriers and lack of medical records complicated care.

Goals

- Carry complete medical history on phone in case of emergency abroad (allergies, medications, past surgeries)

- Share records with international providers who don't have access to US healthcare systems
- Generate translated summaries for non-English-speaking providers

Frustrations

- *"I had a back spasm in Tokyo and went to a clinic. They asked about previous imaging and treatment. Everything was in English, in a US patient portal, and the clinic had no way to access it. I ended up getting repeat X-rays that I'd had six months earlier."*
- *"I carry a paper list of my medications and allergies in my wallet, but it's not enough. When I had chest pain in Berlin, the ER doctor wanted my last EKG and echocardiogram. I couldn't get them—my US hospital doesn't have international fax."*
- *"Every time I travel, I worry: what if I need surgery abroad? How do I give them my medical history? Will they even take a printed PDF seriously, or will they need official records they can't access?"*

A Day in the Life

"I was in Mumbai for meetings when I woke up with severe abdominal pain. I went to a private hospital. Fortunately they spoke English. They asked about previous abdominal surgeries and allergies. I told them I'd had an appendectomy in 2018 and I'm allergic to penicillin, but I had no proof. They didn't want to give me antibiotics without knowing my allergy history was documented somewhere. I spent an hour on the phone with my US primary care office trying to get records faxed internationally. It eventually worked, but the 6-hour time difference made everything worse. If I'd had my complete surgical history and allergy list on my phone as a verified document, the whole thing would have taken 5 minutes."

---

4. Design Components

Journey 1: Current State (Before Mappie)

Trimmed for this web version.

5. Design Decisions & Trade-offs

Decision Framework

For each major design decision, I evaluated:

1. **Problem:** What problem am I solving?
2. **Options:** What approaches did I consider?
3. **Decision:** What did I choose?
4. **Rationale:** Why did I make this choice?

Decision 1: Tool Selection (Notion/Framer vs. Custom Code)

Problem

This course recommends using Notion for databases and Framer for interactive prototyping. Should I use them or build with custom code?

Options Considered

Option A: Notion/Framer

- **Pros:** Visual interface, built-in components, animations, easy libraries, lots of class support, etc.
- **Cons:** Great as a design tool but falls flat very quickly once technical execution exceeds what is available in the libraries or components of the site. Less flexibility.

Option B: Custom React/TypeScript

- **Pros:** I control the hosting, security, API calls, and can write my own very tailored functionality.

- **Cons:** A ton more manual work, no instructor/TA support if things go wrong.

Decision & Rationale

Custom React/TypeScript implementation with Vite build system will give me the most flexibility when it comes to quick design and custom control of real-world implementation. Instead of taking additional time to learn new tools (Framer, Notion (the AI of which looks AMAZING)) I can focus solely on spending it for design & iteration.

Implementing the features I am thinking to add, would require writing most of the application in custom code anyway. Starting custom removes significant initial friction.

Decision 2: Data Processing (Client-side vs. Server-side)

Problem

Where should medical document processing happen; user's browser or a backend server?

Options Considered

Option A: Server-side Processing

- User uploads files to server
- Server processes PDFs, runs OCR, calls AI
- Server sends back structured data
- **Pros:** More processing power, & persistent storage need
- **Cons:** Privacy concerns, & infrastructure cost

Option B: Client-side Processing

- All processing happens in the user's browser
- Only structured JSON sent to API
- No file uploads to server
- **Pros:** Privacy, speed
- **Cons:** Browser limitations, uncontrollable variables

Decision

A Hybrid approach: Client-side parsing (PDF.js, Tesseract), server-side AI analysis (Cloudflare Function)

Rationale

Medical records are highly sensitive personal information. Trust and privacy are #1.

Client-side processing addresses:

- **Privacy:** Raw medical documents never leave the user's device
- **Trust:** Users can verify no uploads by checking network tab
- **Compliance:** Reduces HIPAA compliance burden (though still not HIPAA compliant in current form)
- **Control:** User owns their data completely

Server-side AI analysis is necessary because:

- OpenAI API requires server-side key management (can't expose key to client)
- AI processing requires significant compute beyond browser capabilities
- Structured JSON is much smaller and less sensitive than raw documents

---

Decision 3: User Annotation/Notes

Problem

Should users be able to add their own notes and annotations to the AI-generated summary?

Why I changed course: During my demo meeting, Professor Mikhak asked: "Can users annotate what the AI generated?"

This made me realize annotation isn't a "nice-to-have". It's quite fundamental to user agency. Medical records contain objective (and wrong) data, but patients also have subjective (and sure, maybe wrong) experiences:

- "My rash got worse even though I said it got better."

- "This medication made me gain weight and triggered self-esteem issues."
- "I don’t feel comfortable sharing x with this provider."
- “This medication was prescribed, but I never actually took it.”
- “I don’t remember this health incident on my records/it’s a mistake.”

Without annotation, the tool is read-only and doesn't empower users to add their own voice. Sure they can speak at their appointments. But health records are complex, wait times are long, face-to-face apts are short, and no human has a perfect memory.

Options Considered

Option A: Include annotation with AI processing

- Users have a textbox to add notes to the records before pre-processing their medical files
- Notes NOT included in PDF export directly
- **Pros:** Their notes are integrated into their health record directly
- **Cons:** AI can abuse this info, spit out wrong information, and doesn’t call out specific patient concerns in any obvious way

Option B: Annotations should NOT be processed through AI

- Users add notes only after the processing is completed, at the results page
- Notes included in PDF export directly
- **Pros:** Notes are more concise and detailed to specific concerns
- **Cons:** More directly shared with providers as they are clearly notated in the exportable report as user added notes (for sensitive info, this could be difficult for some users)

Decision & Rationale

Initially my thinking was to have option A, but this would introduce too many uncontrollable variables when it comes to AI-processing, so… I then implemented the “basic” raw annotations after professor feedback and open discussion on the topic.

---

6. Technical Architecture

Data Flow

1. Upload & Parsing (Client-side)

Input: User drags/drops medical documents
Processing:

- File type detection (PDF, Text. Images, & CSV)
- PDF text extraction using PDF.js (loaded via CDN)
- Image OCR using Tesseract.js (npm package)
- CSV parsing for wearable device data

Output: Raw text strings per document (JSON)

2. Structuring (Client-side)

Input: Raw text from all documents
Processing:

- Concatenate all text with document boundaries
- Create JSON payload with metadata

Output: Structured JSON

```
{
  "documents": [
    {"name": "file1.pdf", "text": "extracted content..."},
    {"name": "file2.jpg", "text": "OCR content..."}
  ]
}
```

3. AI Analysis (Server-side - Cloudflare Function)

Input: JSON payload from browser
Processing:

- Cloudflare Pages Function receives JSON
- Constructs OpenAI API request with:
    - System prompt (medical extraction instructions)
    - User message (the JSON payload)
    - Response format: JSON mode
    - Temperature: 0.2 (conservative)
- Sends to OpenAI GPT-4o
- Handles rate limiting with exponential backoff
- Returns structured health data

Output: Health report JSON

*****EASTER EGG: Type the Konami code anywhere while on the results page and you will see a secret.***

Example structure:

```
{
  "patient_info": {...},
  "summary": "...",
  "alerts": [...],
  "vitals": [...],
  "dynamic_sections": [...]
}
```

4. Validation & Display (Client-side)

Input: Health report JSON from API
Processing:

- Sanitize and validate response (sanitize.ts)
- Fixes common AI formatting issues
- Normalize dates, values, & severity levels
- Render with React components
- Generate charts with Recharts

Output: Interactive health summary UI

5. Export (Client-side)

Input: Displayed health summary + user preferences
Processing:

- User selects which sections to include
- jsPDF generates PDF in browser
- Includes charts, alerts, structured data
- Adds security placeholders and disclaimers

Output: Downloadable PDF file

The Tech Stack

Frontend

- **React 18:** Component-based UI
- **TypeScript:** Type safety
- **Tailwind CSS:** Utility-first styling
- **Vite:** Build tool and dev server (super quick iterative testing)

Document Processing

- **PDF.js:** PDF text extraction (CDN)
- **Tesseract.js:** OCR engine (necessary!)
- **Custom parsers:** Wearable device data (also necessary for those huge datasets, ask me how I suffered)

AI Integration

- **OpenAI GPT-4o:** Medical record analysis (good balance of speed, output, and affordability)
- **Cloudflare Functions:** Serverless API endpoint (hiding my API key)
- **Custom prompting:** Citation and evidence requirements in results

Data Visualization

- **Recharts:** Chart library
- **jsPDF:** PDF generation
- Exploring alternatives for PDF exports due to language character limitations in PDFs (Chinese/Arabic, etc)

Deployment

- **Cloudflare Pages**: Static hosting + Functions (FREE!)
- **GitHub**: Version control (Framer does not offer rollback?)
- **Wrangler**: Deployment CLI (to CF/how to handle my functions)

---

7. Implementation & Features

Core Features & User Need Mapping

Feature 1: Multi-Format Document Upload

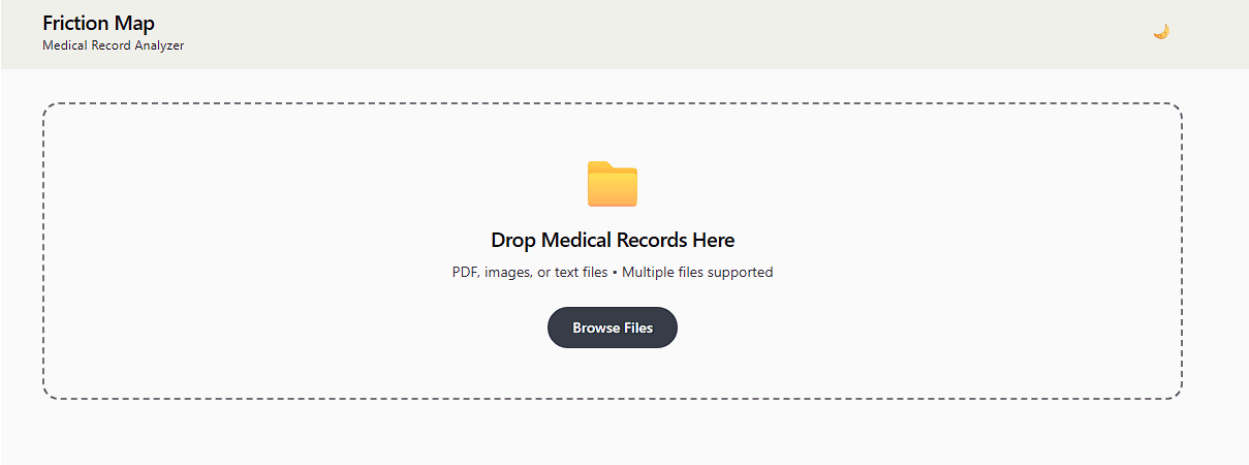
User Need: "I have medical records in different formats. PDFs from patient portals, scanned images from old paper files, CSV exports from my Fitbit."

Implementation:

- Drag-and-drop upload zone
- Support for: PDF, JPG, PNG, CSV, TXT, MD, JSON, XML
- Multiple file handling (process sequentially)
- File type detection and routing to appropriate parser

Design Decision: Made drag-drop the primary interaction (no "browse" button) to reduce friction. Users think of uploading as "dropping files in" not "selecting from dialog."

Screenshot:



Feature 2: OCR for Scanned Documents

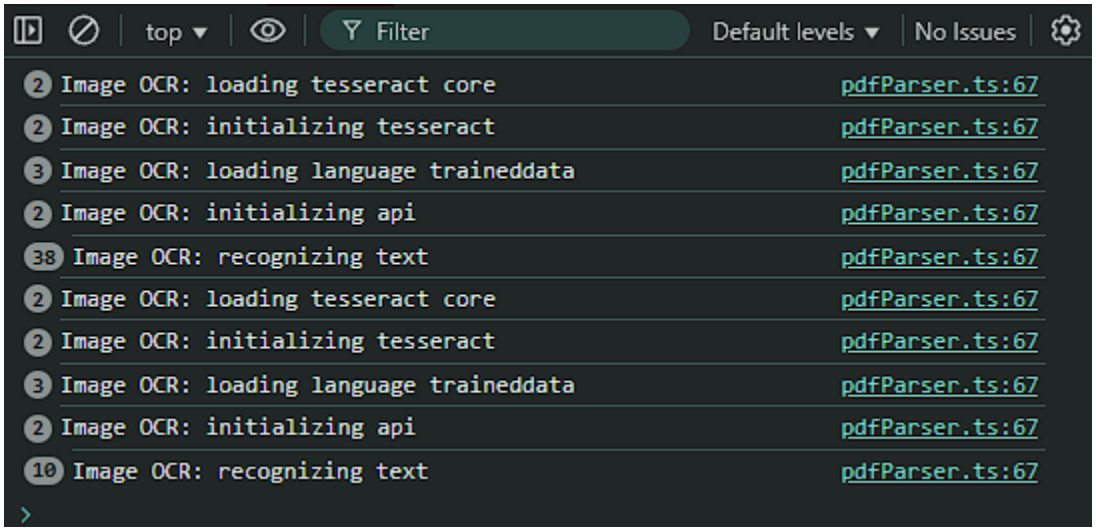
User Need: "My primary care doctor has a huge stack of paper records. Even if I got them, they'd be scanned images, not searchable PDFs."

Implementation:

- Tesseract.js for client-side OCR
- Automatic detection when PDF page has minimal selectable text
- Progress indicator during OCR processing
- English language optimized (could expand to multi-language)

Design Decision: Run OCR automatically rather than making it a manual option. Users shouldn't need to understand what OCR is - the tool should just work regardless of file type.

Screenshot:



Feature 3: AI-Powered Health Summary

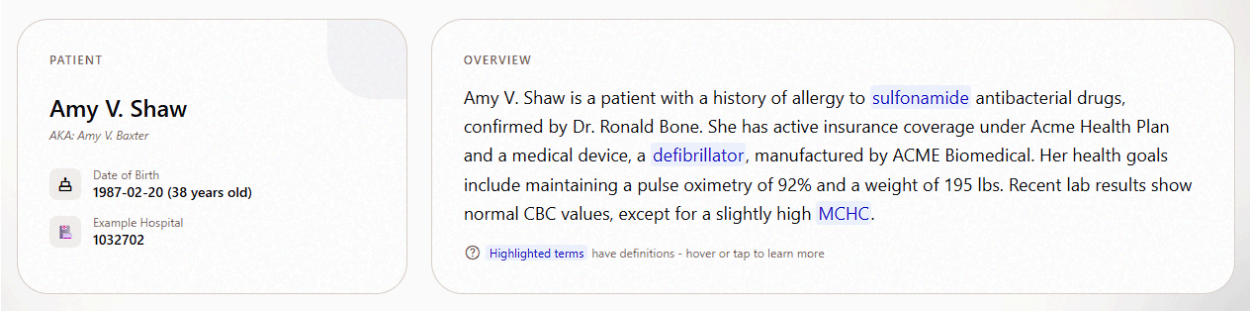
User Need: "I don't want to read through 50 pages of medical jargon. I need the key information organized and easy to understand."

Implementation:

- GPT-4o analyzes all extracted text
- Structured JSON response with consistent schema
- Categories: Medications, Conditions, Procedures, Vitals, Allergies, etc.
- Chronological ordering within categories
- Status tracking (Current vs. Past)

Design Decision: Use dynamic sections instead of fixed categories. This allows the AI to create relevant sections based on what's actually in the records (e.g., "Mental Health" section only appears if relevant).

Screenshot:



Feature 4: Notable Alerts

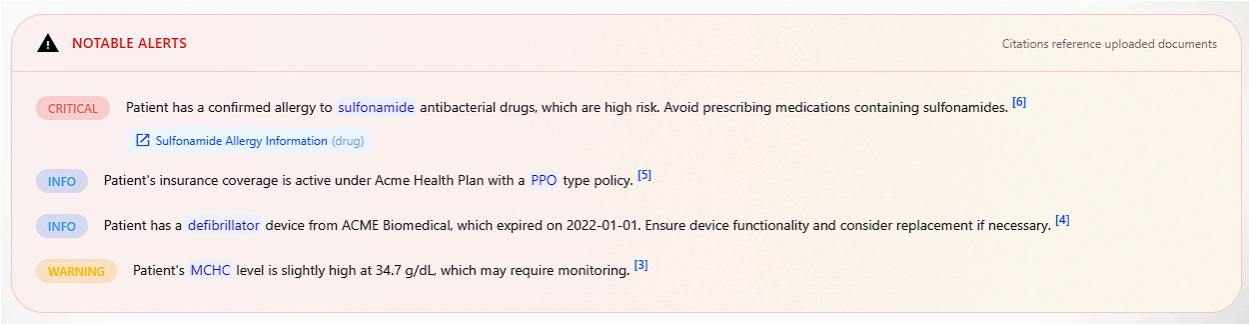
User Need: "I want to know if there are any red flags or contradictions I should discuss with my doctor."

Implementation:

- AI identifies: Drug interactions, contradictory diagnoses, missing follow-ups, concerning trends
- Severity levels: Critical, Warning, Info
- Evidence-based: Citations to specific documents
- Reasoning explained in plain language
- Recommendations for action

Design Decision: Display alerts prominently at top of summary (not buried). Use color coding (red/yellow/blue) but don't rely solely on color (also use icons and text labels for accessibility).

Screenshot:



Feature 5: Interactive Charts

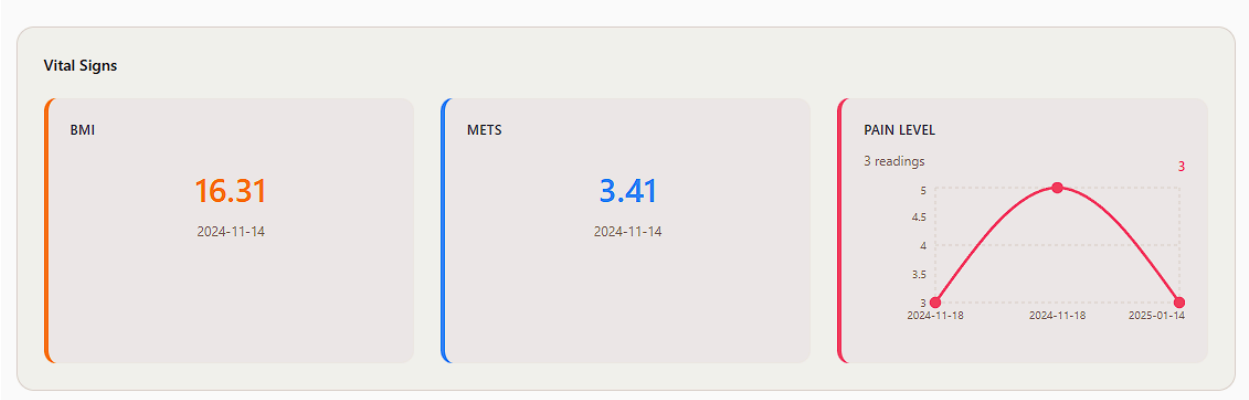
User Need: "I want to see how my weight/blood pressure/lab values have changed over time."

Implementation:

- Recharts line charts for time-series data
- Support for: Weight, BP (systolic & diastolic), Heart Rate, Labs, Glucose, etc.
- Automatic chart generation when sufficient data points exist
- Hover tooltips for precise values
- Responsive sizing

Design Decision: Only show charts when there are 2+ data points with dates. A single measurement isn't a "trend" so shouldn't be visualized as a chart.

Screenshot:



Feature 6: Customizable PDF Export

User Need: "I want to share a health summary with my new doctor, but I don't want to include everything - some history isn't relevant to this appointment."

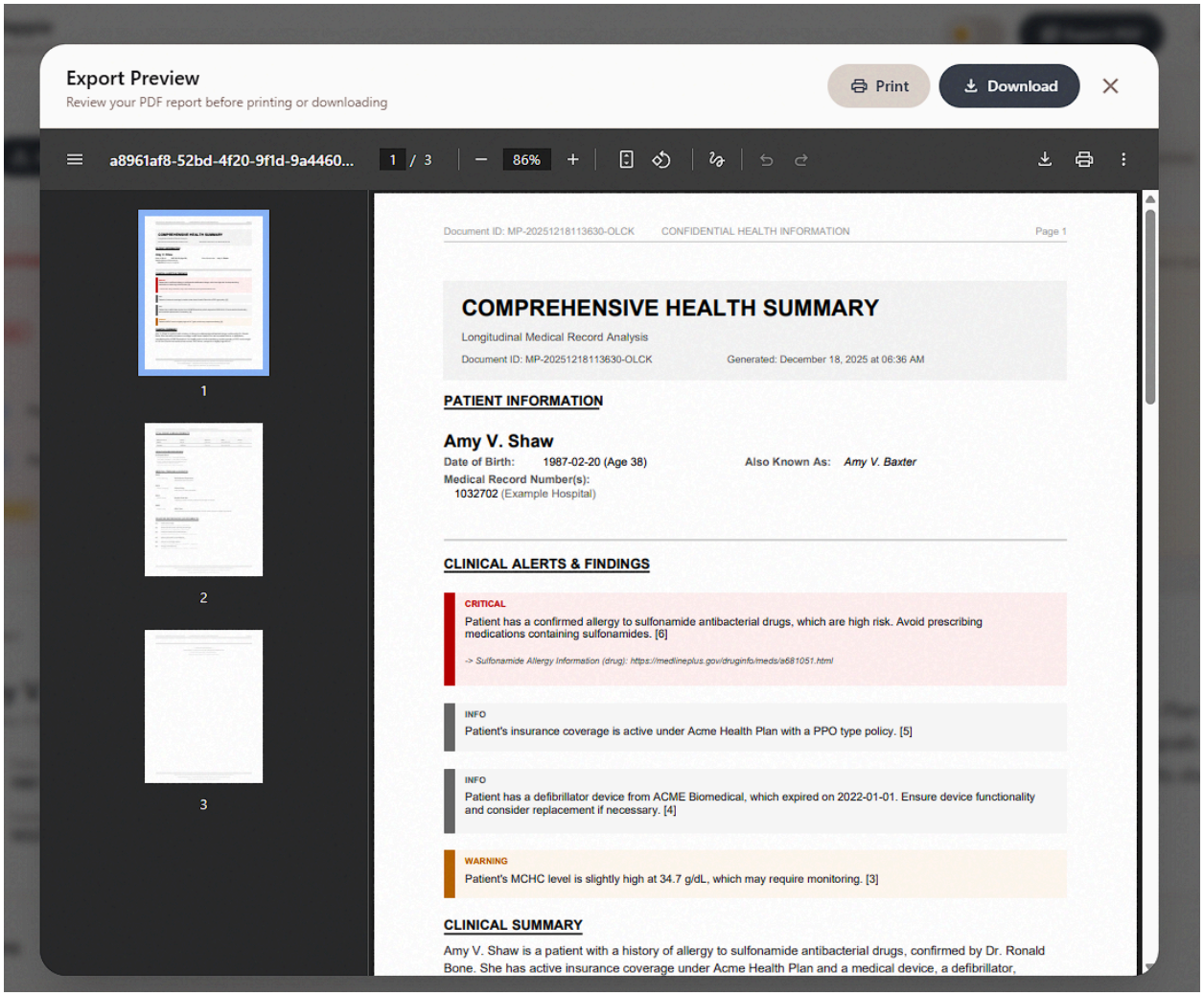
Implementation:

- Filter buttons to show/hide sections
- Export dialog with section checkboxes
- jsPDF generates clean, professional PDF

- Includes: Summary, filtered sections, charts (as images), alerts
- Security placeholders (for production version)

Design Decision: Make filtering interactive and immediate (not just in export dialog). Users can hide/show sections in the main view, and export reflects their customization. This provides clear feedback about what will be in the PDF.

Screenshot:



Feature 7: Privacy-First Architecture

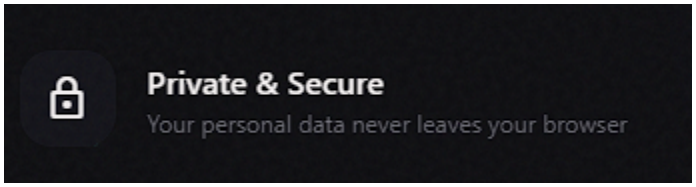
User Need: "I'm not comfortable uploading my medical records to some random website. How do I know they're secure?"

Implementation:

- All document parsing happens in browser (PDF.js, Tesseract in browser)
- No file uploads to server - files never leave user's device
- Only structured JSON (not raw documents) sent to API
- No persistent storage - data exists only in browser memory
- No cookies, no tracking

Design Decision: Transparency about privacy. Add a prominent notice explaining that documents are processed locally. This builds trust.

Screenshot:



Component Architecture

Main App Component (App.tsx)

Responsibilities:

- State management for file processing
- Orchestrating upload → parse → analyze → display flow
- Error handling and user feedback
- Rendering child components

Key State:

```
const [files, setFiles] = useState<ParsedFile[]>([]);
const [analyzing, setAnalyzing] = useState(false);
const [report, setReport] = useState<HealthReport | null>(null);
const [error, setError] = useState<string | null>(null);
```

AI Integration (chat.ts)

Responsibilities:

- Cloudflare Pages Function (server-side)
- OpenAI API calls
- Retry logic with exponential backoff
- Rate limiting handling
- Error handling

Key Features:

- Comprehensive system prompt with medical extraction instructions
- JSON mode for structured output
- Input truncation for very large files
- Citation requirements for transparency

PDF Export (pdfExport.ts)

Responsibilities:

- Generate PDF from health summary
- Include charts as images
- Handle multi-page layouts
- Add headers, footers, disclaimers

Key Challenge: Charts need to be converted to images (canvas → base64) before adding to PDF. This requires rendering the chart, capturing it, then including it in jsPDF.

Accessibility Considerations/Potentials

WCAG 2.1 AA Compliance

Color Contrast:

- All text meets minimum 4.5:1 contrast ratio
- Alert colors tested for color-blind users
- Icons used alongside color for severity (not color alone)

Keyboard Navigation:

- All interactive elements accessible via Tab
- Visible focus indicators
- Logical tab order (top to bottom, left to right)

Screen Reader Support:

- Semantic HTML
- ARIA labels for icon-only buttons
- Alt text for meaningful images
- Live regions for status updates (e.g., "Processing...")

Responsive Design:

- Mobile-first approach
- Touch targets minimum 44x44px
- Text scales without breaking layout
- Works on screens from 320px to 4K

---

8. Iteration & Feedback

Feedback & Other Responses

Methods

Me: Self-testing with my own devices.

Instagram Outreach: Asking strangers (followers) online to upload files and test and provide feedback with screenshots.

Screensharing: A couple instances I watched someone live-explore the site and noted their UX in real time.

Class: Watching outer students navigate the thought-process of their site design gave me some new pointers.

---

9. Reflection

What Worked Well

1. Understanding the Gap

Building a tool I **personally** need, kept me motivated and grounded in real use cases. I didn't have to abstractly imagine what users may want, because I am front-and-center as a user persona.

Evidence: Every feature directly addresses a pain point I've experienced. The tool solves a problem I will actually use after this course. (I'm printing out my records and taking them overseas! Hopefully the doctors take the exported report seriously, wish me luck!!!)

---

2. Technical Risk-Taking

I chose a technically ambitious project (OCR, AI, PDF generation, client-side processing, latest UX design standards, etc.) This pushed my skills and produced something genuinely novel.

Evidence: I maintained engagement and built a working live prototype. I believe had I stuck with Notion/Framer, it would not have been challenging enough for me to execute to the same extent (I would lose interest).

---

3. Responsive to Feedback

When Professor Mikhak raised concerns about AI transparency and user annotation, I didn't defend my choices (okay well I did a little bit) but I still listened and implemented those improvements.

Evidence: Added citations to AI alerts and a basic annotation system post-feedback.

What I'd Do Differently

1. Not Fear Building The Prototype

Even if I built with custom code, I should have:

- Not feared getting too attached to any iteration of the project
- Not feared UX issues or bugs or other functional mistakes
- Been OKAY with not having all the features figured out

Why It Matters: I avoided development because I feared that touching code would sink me into an engineering mindset and I'd lose track of the design process thinking entirely.

---

2. Test With Real Users Earlier

I assumed my experience was quite representative. Should have:

- Shown the super rough early prototypes to people
- Asked 2-3 people with real medical records to try it
- & watch them use it (**!!!without helping!!!**) to find UX issues

Why It Matters: I MAY have discovered feature gaps and other usability issues earlier, or my design could have pivoted very differently.

---

3. Carefully Document Concurrently

Should have maintained a more simple, universal log of:

- Weekly design decisions and rationale
- Screenshots showing UI evolution
- Feedback received and how it shaped work

These existed, but not before capturing each step of the process and many of the notes and feedback were not localized to any one space. My entire design process was scattered and took a long time to assemble in a cohesive fashion for this document.

Why It Matters: Concurrent documentation is snapshot-capture. It's more accurate in the sense that I'm less likely to forget parts of the process retroactively.

Design Thinking Methodology

My Approach: Compressed and non-linear, but covered elements:

- **Problem**: Identified through personal experience
- **User Research**: Autoethnography + informal conversations
- **Personas**: Grounded in real people
- **Journeys**: Mapped late, but based on actual experiences
- **Prototyping**: Built a functional prototype
- **Testing**: Considerable testing (in a short span of time)
- **Iteration**: Responded to professor/user feedback, refined features

Reflection: I completed the design thinking process, but in a more compressed, non-traditionally undocumented way. Extensive thinking happened, however the communication of that thinking is where I struggled to articulate my decisions.

Component-Based Design

Course Emphasis: "Thinking in components"

My Implementation Has:

- React component architecture
- Reusable UI components (alerts, cards, buttons)
- Modern Material Design 3.0 SYSTEM (typography + icons + etc.)
- Very modular code structure (organized as individual .ts files where appropriate)

Alignment: Technical alignment here is good, but without using Notion/Framer, I did not have access to the same component libraries and extensions as other students.

Future Roadmap

V2 Features

User-Facing:

- Data persistence with encrypted local storage option
- FULL Multi-language character export support(Chinese/etc.)
- Export to EHR formats (HL7, FHIR)

Technical:

- HIPAA compliance assessment and implementation
- Clearer encryption
- Audit logging (mini debug mode already exists)

AI Improvements:

- RAG system with approved medical knowledge sources
- Confidence scores for alerts
- Some false positive feedback mechanism
- Integration with medical databases (RxNorm, SNOMED CT)
- Multi-model compatibility, auto-detect based on ingested data

---

Long-Term Vision (1-2 Years?)

Product Evolution:

- Mobile app (React Native?)
- Validation/integrity checks (hashing)
- Integration with major EHR systems
- API for third-party healthcare apps (Epic/MyCHart/Privia/etc.)

Business Model Ideas (if pursuing via iLabs):

- Freemium: Basic features free, advanced features paid
- OR Nonprofit: Grant-funded, free for all patients

- OR Open Source: Community-maintained, monetize through support/hosting

Impact Goal:

- Help users organize & better understand their medical records
- Reduce appointment prep time by X%
- Identify medication safety issues proactively

Reflections

My Most Valuable Lesson: I always assumed that users need to understand HOW something works in order to trust it. I realize this isn't the case, and many testers don't even ask privacy/security/operability questions that I anticipated they would. That doesn't mean it shouldn't be considered and implemented. It's tricky to work with emerging software that is effectively a "black box" in terms of design and engineering, but that is the future users have already come to start accepting.

P.S. I also realized just how much of my "design thinking" was intuitive decision making vs concrete, step-by-step problem solving. I struggled to articulate and justify some of my design decisions as I needed to mentally reverse-engineer my thought process.

You've reached the end of my Design Process Documentation