Altered Health

Requirements Document Final

Team Members:

Jasmine Flowers
Ian Nieto
John Gornick
Jerry “Tre” Kelley

Client: Jesslynn Stull
Mentor: Bailey Hall

Team Signatures:__Ian Nieto_____
__John Gornick_____
__Jerry ‘Tre’ Kelley_____
__Jasmine Flowers_____

Client Signature:__Kelsey Denham_____
Date: 11/26/25 - **Version**: 2.0

Table of Contents

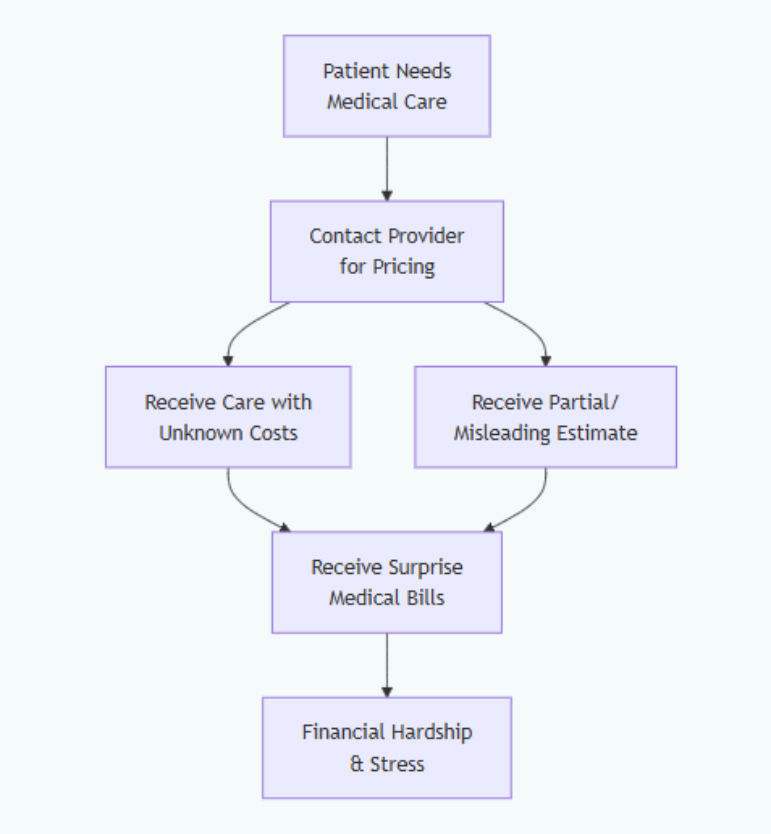
1. Introduction

The United States healthcare system is one of the largest and most critical sectors of the national economy, representing more than $4 trillion in annual expenditures. It is a system filled with multiple providers, insurers, and patients, and one in which pricing for the services provided is usually not very straightforward or consistent. It is based on this vast and complex industry that an important disconnection between the consumers of care and the cost of that care exists. Too often, patients are asked to make major healthcare decisions without access to what many consider a basic consumer right: knowing the price of a service before the service is performed. The result is a process through which individuals receive care only to be confronted with confusing and unexpected bills months later, with little recourse. This lack of transparency results in widespread confusion and surprise financial burdens, and it contributes to an overall erosion of trust in the healthcare ecosystem.

Our project, undertaken in partnership with our client, who brings deep personal conviction to this problem, addresses this critical market failure head-on. Though new to entrepreneurship in the healthcare technology sector, she has first-hand experience with pricing opacity and has thus embarked on a highly dedicated mission to create a solution. The purpose of this document is to outline the requirements for building a web-based application whose purpose would be to empower patients by creating unprecedented transparency in healthcare pricing. This application provides an easy-to-access source that patients can use to compare procedure costs and insurance copays among different providers, prioritizing the Arizona market, so any launch will be feasible and controlled. By transforming opaque and fragmented data into clear,

actionable information, we aim to reduce patient stress and financial hardship, eventually fostering a more informed and equitable health marketplace.

2. The Problem



Fundamental to the U.S. healthcare system is a lack of pricing transparency, leaving patients vulnerable to financial strain. Today's patient journey reflects a broken workflow: people usually schedule care without knowing the cost upfront. The provider delivers the service, bills the insurance, and only later does the patient receive an Explanation of Benefits (EOB) followed by the actual bill, often with unexpected charges. This delay creates surprise medical costs weeks or even months after care, adding unnecessary financial stress and leaving patients blindsided.

This workflow is riddled with several key flaws: A complete information blackout for patients seeking to receive care, at a time when providers simply do not have accurate cost information available. These exist in fragmented pieces in disconnected systems, from provider networks to insurance portals, with no way to aggregate, compare, or cross-check the data. Community knowledge is not being shared so that others might benefit from the experiences of similar procedures.

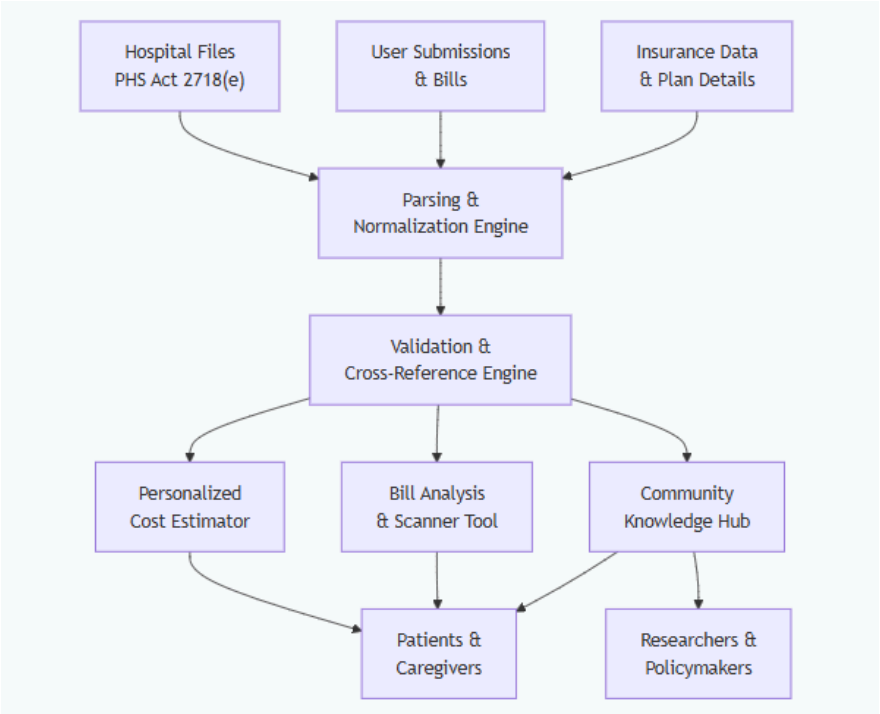
These systemic failures create a significant power imbalance, whereby patients make critical decisions without basic price information available to them in other economic sectors. The resulting surprise bills contribute substantially to medical debt and create barriers to needed care. Our solution must address these particular deficiencies by creating a centralized platform that transforms the historically opaque pricing of healthcare into transparent, comparable data.

3. The Solution

We are creating a centralized web application that will revolutionize healthcare pricing with transparency by providing the patient with the ability to make informed decisions based on accurate, upfront cost information. Our platform directly addresses the documented problems by aggregating and standardizing data from multiple sources required by section 2718(e) of the Public Health Service (PHS) Act into an accessible interface that enables informed decision-making. It eliminates information blackouts through comprehensive procedure pricing, prevents misleading estimates via integration of insurance copays, combats surprise billing with future bill analysis tools, and builds missing community knowledge through sharing user experiences.

The system leverages proprietary health-care databases, public data sets, and user submissions, all processed through parsing engines that scrape and standardize disparate information. This data goes through computational normalization across billing formats and community validation, cross-referencing official data with real patient experiences. Personalized cost estimates and community-driven provider evaluations create a powerful knowledge resource that benefits everyone.

To visually articulate how these components work together, view the following diagram:



Simpler alternatives, like linking to existing hospital files or manually entering data, would suffer from fragmentation and scalability issues; thus, we rejected them. Our automated aggregation, combined with community validation, is optimally scalable and accurate, saving on technical investment in data processing but offering a superior user experience. This

fundamentally changes how consumers engage with healthcare, placing downward pressure on prices via the market while reducing medical debt through transparent, comparable cost information that allows true value-based decisions.

Project Requirements

1. Domain-Level Requirements

Domain-level requirements are the high-level requirements that our software will be required to perform. These will include making our healthcare data not only searchable, but also accessible. We will also need to implement a user-friendly login/signup page. Our domain-level requirements are as follows:

D1. Searchable healthcare data (pricing, provider, procedure, etc.): The software needs to have searchable healthcare data for customers to view and look into.

D2. Accessible Healthcare data: After the search, and a service is selected, there should be clear healthcare pricing data shown about the given search result.

D3. User-friendly login/signup: There should be login and signup pages for users to access their accounts, which may store personal information.

2. Functional-Level Requirements

Functional level requirements are the functions our software must perform to achieve the domain-level requirements. On the surface level, these will include the searchable

healthcare data and the accessible healthcare data. For the following sub sections we have implemented “F” tags, which are MUST-haves, and “S” tags, which are SHOULD-haves.

F1. Searchable healthcare data:

F1.1: Search bar: The search bar should be in the center of the homepage, similar to many chat-based LLMs. When the search is activated, it moves the user to the search page. The search should find relevant procedures at nearby clinics to select from.

F1.2: Search backend: Search should search containers in Azure to categorize based on procedure and location. Additionally, the search should take the user input and interpret what procedures may be similar to the search query.

F2. Accessible Healthcare data:

F2.1: Selection-based healthcare data: After selecting an option through the search, all information about the item should be displayed in a user-friendly way. Relevant data would include procedure information and pricing.

F3. User-friendly login/signup:

F3.1: User Login Page: The user login page should have a secure sign-in section that also includes a forgot password button. This may be secured via an Azure authentication token and HTTPS end-to-end encryption.

F3.2: User Signup Page: The user signup page should have a secure sign-up section that also includes several user security login options. This may be secured via an Azure authentication token and HTTPS end-to-end encryption.

S1. Health document scanning:

S1.1: Bill document scanning: Logged-in users with subscriptions should be able to upload a bill and have our software scan through issues with the bill and inform the customer if they have been overcharged for their health service.

S2.2: Health document scanning: The system should be able to process user-submitted medical bills, EOBs, or price sheets and convert them into structured, machine-readable data

S3: Region-based search: The software should be able to search by region to find region-specific information.

S4: Document scanning performance: Scanning documents may be much slower than other functional requirements, as it may end up using a generative AI model that we may need to personally host or find a provider. This will be slower and have a much higher backend CPU/GPU requirement than anything else our software does. New, powerful hardware will be essential for this kind of processing.

3. Performance Requirements

These are requirements of the software to meet certain performance standards, to make sure the user has a good experience and good performance on the backend.

P1: Search Performance: Search should be extremely responsive, giving results in under 5 seconds. The speed of search will be limited by how much data there is to search through, so sorting it is a must. Our decompiler will then sort the results and place the data in specific Azure containers, so when searching, it only searches relevant containers instead of everything. Additionally, each container will be sorted by procedure type, so only specific locations in the container will need to be searched.

P2: Database Decompiler Performance: The decompiler will be the most CPU-intensive portion of the software when running. Ideally, we will only run the decompiler when documents get updated, which may be on a schedule; however, we will only know as we get more data. Minimizing how often we need to run the decompiler will be key to minimizing processing requirements.

P3: Client-side performance: The client's side of the software will need to be very quick and responsive; this will be easily achievable using the React framework within NextJS. If the frontend has to wait on backend data due to an extensive search or a bad connection, it should make it clear to the client that it is processing or that there is a bad connection.

4. Environmental Requirements

Our project will be developed and deployed entirely within a controlled cloud environment using Microsoft Azure. These environmental requirements will define the external constraints under which our application must operate.

E1: Hardware and Platform Constraints

E1.1: The system will be hosted using Azure App Services, with a PostgreSQL Database for storage and Azure Blob Storage for data management.

E1.2: Client access will occur through standard web browsers (Chrome, Edge, Safari, Firefox) on desktop and mobile devices with internet connectivity.

E1.3: No physical servers or on-premise infrastructure will be required beyond developer workstations.

E2: Software and Development Environment

E2.1: Development will utilize Visual Studio Code for code development and GitHub for source control, CI/CD pipelines, and deployment automation.

E2.2: Backend services will be developed using Python (FastAPI) and Node.js, while the frontend will use React.

E2.3: Data will be managed through an Azure PostgreSQL Database with encryption at rest and in transit.

E3: Security and Compliance Environment

E3.1: All development and deployment activities will adhere to HIPAA and HITECH compliance standards to ensure the protection of healthcare-related data.

E3.2: User authentication will leverage Azure Active Directory and token-based authentication for secure role access.

E3.3: Data will be encrypted using TLS 1.2 or higher for transfers and AES-256 for storage.

E4: Testing and Deployment Environment

E4.1: A dedicated staging environment will be maintained separately from production to ensure controlled testing before deployment.

E4.2: Continuous integration and testing pipelines will verify builds through unit and functional testing before merging.

E4.3: Deployment to the production environment will be executed through GitHub Actions, which integrates well with GitHub's version control.

Potential Risks

This is a project that will have to handle large amounts of information, much of which is sensitive. It is highly important that we are able to accurately assess what the largest risks are and the likelihood of them coming to pass so that we do not make mistakes with crucial information. The greatest perceived risks to our development are as follows.

R1: Parsing and Normalization Failures

The system depends heavily on extracting and standardizing pricing files from hospitals and insurers. These files often vary in structure, formatting, naming conventions, and encoding. If the parsing engine cannot correctly interpret these files, the platform will be unable to populate accurate or complete pricing datasets.

Impact: High

Likelihood: Moderate

Mitigation: Design the parser to be modular and adaptable, create schema-flexible ingestion pipelines, and continuously test with sample datasets from multiple providers.

R2: Failure to Interpret Insurance Documents

Insurance documents such as EOBs, prior authorization notices, or coverage summaries may contain inconsistent terminology or ambiguous formatting. The system may misread copay, coinsurance, deductible, or negotiated rate information, leading to inaccurate patient estimates.

Impact: High

Likelihood: Moderate

Mitigation: Develop controlled vocabularies, validation checks, and mapping rules; maintain human-review loops early in the pipeline.

R3: Search Engine Misclassification or Irrelevant Results

The search engine may incorrectly map user queries to the wrong procedures, return incomplete results, or fail to interpret synonyms (e.g., "colonoscopy screening" vs. "colorectal exam CPT 45378"). This would damage user trust and limit the usability of the platform.

Impact: High

Likelihood: Moderate

Mitigation: Implement synonym dictionaries, CPT/HCPCS mappings, relevance scoring, and iterative feedback loops based on real user behavior.

R4. Complexity of AI-Based Bill Review

AI-driven document scanning for bill review may be too complex to deliver within the capstone timeline due to the variability of bills, OCR challenges, and the need for medical coding accuracy.

Impact: High

Likelihood: High

Mitigation: Classify this feature as a Should-Have rather than a Must-Have for MVP; begin with a simplified rules-based scanner or structured form parsing before full AI integration.

R5. Incorrect Cost Estimates Leading to User Misinformation

If cost data is incomplete, not normalized correctly, or incorrectly interpreted, the system may generate misleading estimates. This could cause users to make decisions based on inaccurate pricing.

Impact: High

Likelihood: Moderate

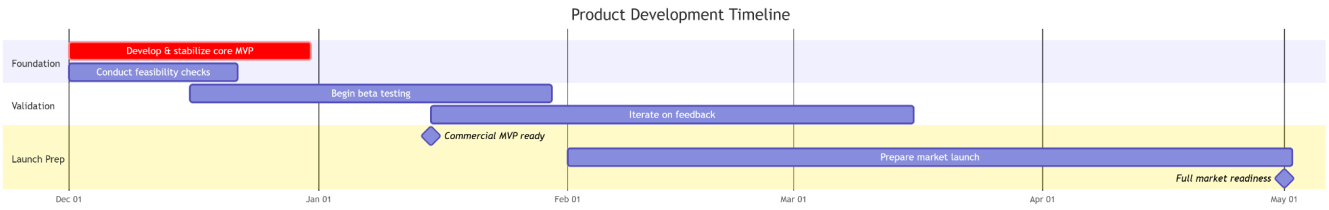
Mitigation: Cross-validate data across multiple sources, add error margins to displayed estimates, and provide disclaimers when data confidence is low.

Project Plan

The project will be completed using an Agile-Scrum methodology with two-week sprints, regular client check-ins, and progress tracking. This will be done through GitHub.

- **Phase 1: Research and Setup**
    - Finalize system requirements and design architecture diagrams.
    - Configure Azure resources (App Service, SQL Database, Blob Storage).
    - Begin preliminary UI mockups and prototype user login functionality.
- **Phase 2: Core Development**
    - Implement user authentication and a pricing search engine.
    - Integrate the healthcare pricing database API and internal data normalization pipeline.
    - Begin frontend–backend integration testing and internal code reviews.
- **Phase 3: Data Validation and Testing**
    - Test scraping and parsing engines with real sample data.

    - Conduct unit testing, fuzz testing for input validation, and load testing on Azure.
    - Begin implementing community feedback and data correction mechanisms.
- **Phase 4: Deployment and Client Review**
    - Deploy MVP to the Azure web environment.
    - Demonstrate full “search → compare → view pricing → export” workflow.
    - Gather client feedback for the next iteration.
- **Phase 5: Final Documentation and Handoff**
    - Deliver technical documentation, setup guide, and risk summary.
    - Conduct final presentation and repository handoff to client and mentor.



Summary and Outlook

By aligning milestones with specific functional requirements, the project ensures that every technical objective supports the overall goal of transparent healthcare pricing. Each phase ends with tangible, verifiable deliverables, functional code, prototypes, or documented results. This will allow the client to validate progress. The team will update the Gantt chart regularly to reflect timeline adjustments and evolving priorities.

Conclusion

Healthcare is a tricky industry to tackle, but through this software, we are looking to help healthcare consumers and healthcare professionals understand the inner workings of it better. Our client's main goal to get this up and running as a product is to make the search feature work as soon as possible. The search feature will allow users to find healthcare and healthcare prices very quickly, which is a very achievable goal for the first iteration of the product. This foundational feature will directly address the core obstacle of information blackouts by making comprehensive pricing data available.

In summary, we outlined the issues this project will be designed to address and what we aim to achieve with the product. We have shown what features we are implementing in the product for the client. We have shown how these features will work and perform. Lastly, we have shown the risks and plan for mitigating such risks in the production phases. This plan of action should result in a well-functioning product for our client in the very near future. Our analysis confirms that the proposed solution is both technically feasible and effectively targeted to solve the critical challenges of healthcare pricing transparency. Our current plan of action will result in a properly functioning product for our client. We are confident that our platform will successfully empower patients with the knowledge they need to make financially sound and informed healthcare decisions.