Tikvah Health & Wellness

MVP Technical Proposal

Prepared by: Douglas Thomas

Date: May 15, 2025

Version: 1.0

Executive Summary

Tikvah Health is building a modern search experience to help individuals discover and connect with holistic health providers. The platform will launch with a regional focus on Charleston, SC, with the long-term vision of supporting national and international reach. This proposal outlines the design and development of Tikvah's MVP: a filterable, cloud-deployable provider database that forms the foundation for future B2B and B2C growth.

The MVP will be developed in two iterative phases:

- **Phase 1** delivers a locally-run prototype, emphasizing backend services and basic provider search functionality.
- **Phase 2** transitions the product to a secure, cloud-hosted platform, including a demo-ready web interface and supporting infrastructure.

This approach provides Tikvah with:

- **An opportunity to validate the technical direction early** and ensure the implementation aligns with Tikvah's expectations before investing in cloud deployment and broader integration
- A flexible, scalable foundation built using modern tools
- A clear architecture for future enhancements, including UI expansion, insurance integrations, and user onboarding
- A demo-ready product aligned with Tikvah's short-term investor conversations and long-term platform strategy

Strategic Importance of the MVP

- **Validates Tikvah's core concept** with a working product that demonstrates value to employers, providers, and early users
- **Supports fundraising efforts** by offering investors a concrete, demoable artifact that showcases technical progress and execution readiness
- **Provides architectural clarity** that reduces future rework and accelerates time-to-market for Phase 2 and beyond
- **Aligns internal stakeholders and partners** around shared priorities and a practical rollout timeline

The MVP is scoped to fit within a **171-hour development budget**, including generous buffers for polish, infrastructure, and documentation. It is designed for speed, clarity, and extensibility, delivering real value quickly while setting the stage for future growth.

Scope of Work

The MVP will deliver a foundational platform that enables users to search for and view holistic health providers. It will consist of two sequential development phases, each with clearly defined objectives and deliverables.

What's Included

- A **searchable provider directory** built on a structured backend, allowing users to filter by specialty, location, and other attributes
- A **basic web interface** that displays search results and detailed provider profiles
- A **cloud-ready API layer** designed to support both internal validation and future frontend expansion
- **Infrastructure for deployment**, including containerization, hosting setup, and CI/CD automation
- **Documentation** covering architecture, API usage, and deployment strategy

What's Not Included in This MVP

To stay focused on the most critical early functionality, the following features are **explicitly out of scope** for the MVP:

- User authentication, profiles, or login functionality
- Calendar booking, scheduling, or availability features
- Provider onboarding workflows or dashboards
- Search UI enhancements (e.g., map integration, autocomplete)
- Payment processing or monetization features

- Deep analytics or tracking
- Integration with Artificial Intelligence models

These features may be considered in future iterations based on user feedback, partnerships, and platform evolution.

Target Users

The MVP is designed primarily for:

- **Internal stakeholders and early investors**, to evaluate the product's direction and market potential
- **Employers and navigators**, who will ultimately access the platform on behalf of members
- **Practitioners**, who may review the platform's representation of their profiles and specialties

This focused, phased approach enables Tikvah to validate product-market fit, gain stakeholder alignment, and lay the technical groundwork for rapid iteration.

Architecture Overview

The MVP will be built with a modular, modern architecture that emphasizes speed, flexibility, and a clear path to future enhancements. The system consists of three core components:

1. **Backend API**
   A central service that processes requests, performs searches, and manages provider data. It exposes a structured interface (API) that other parts of the system, such as the web interface or future services, can use to access information.
2. **Database**
   A secure, cloud-compatible database stores information about practitioners, organizations, specialties, and categories. It is designed to scale over time and can be extended to support future needs such as user accounts or analytics.

3. **Web Interface**
   A simple, server-rendered website allows users to search for providers and view their profiles. The interface is intentionally lightweight to enable fast iteration and early feedback without investing in complex frontend architecture.

Key Design Principles

- **Separation of concerns**
  The data layer, business logic, and user interface are clearly separated, making the system easier to maintain and extend.

- **Phased deployment**
  The product will be developed locally first, then deployed to the cloud once core functionality is validated. This allows early feedback and iteration without the cost or complexity of full-scale infrastructure.

- **Cloud readiness**
  The architecture is designed from day one to be cloud-compatible, using modern tools that can scale as Tikvah grows.

- **Simplicity first**
  The MVP avoids unnecessary complexity, prioritizing a stable, usable foundation over advanced features. Every component is chosen to balance short-term delivery speed with long-term adaptability.

Phase 1 – Local-First MVP

Objective

Build and validate the core functionality of Tikvah's provider search experience in a local development environment. This phase establishes the foundational backend services, data structures, and simple UI needed for early feedback and internal alignment.

Key Features

- A searchable provider directory with filterable fields (e.g., specialty, location)

- Backend services that expose this data through a structured interface (API)

- Basic web pages for search and profile viewing
- A seed data loader to ingest initial provider lists from CSV or JSON
- Full local execution using Docker (no cloud infrastructure required)

Technical Components

- **Backend API:** Built using FastAPI, structured around modular routes and clear data models
- **Database:** MongoDB used to store and query provider and organization information
- **Web Interface:** Jinja2-rendered HTML pages (e.g., search page, provider profile)
- **Development Environment:** Docker-based setup for running the API and database locally
- **Testing:** End-to-end API test coverage using pytest, with manual UI testing for feedback

Timeline and Effort

The Phase 1 implementation is scoped at **97 total hours**, including:

- **Core engineering:** 68 hours (API development, database integration, ingestion tools, testing)
- **UI layer:** 13 hours (basic search form, results page, and profile views)
- **Built-in buffer:** Multiplier applied to account for iteration, polish, and minor rework

Deliverables

- A working prototype that runs entirely on a local machine
- Searchable list of sample providers with working filters and profile pages
- Complete source code with documentation and local setup instructions
- Foundation for Phase 2 deployment with no throwaway work

Phase 2 – Cloud Deployment & Hosting

Objective

Transition the working MVP from a local prototype to a secure, cloud-hosted application. This phase ensures the product can be accessed via the web, presented to investors or partners, and operated with modern infrastructure best practices.

Key Features

- Cloud deployment of the backend services and web interface
- Hosted domain and SSL configuration for secure public access
- Continuous integration and deployment (CI/CD) pipeline for future updates
- Light polish of the user interface for external demo use
- Architecture and deployment documentation for internal use and future scaling

Technical Components

- **Cloud Infrastructure:** Hosting via AWS (e.g., EC2 or ECS), with MongoDB Atlas as the managed database
- **Web Hosting:** Static or server-rendered site hosted through the backend or optionally served via a separate platform (e.g., Vercel or S3)
- **CI/CD Pipeline:** Automated testing and deployment using GitHub Actions
- **Security & Reliability:** API authentication, secure environment configuration, and logging for observability
- **Documentation:** A complete technical overview of the cloud environment, deployment steps, and handoff guide

Timeline and Effort

Phase 2 is scoped at **79 total hours**, including:

- Cloud infrastructure setup and deployment
- CI/CD configuration and security hardening
- UI polish for presentation
- Full documentation covering the deployed architecture and delivery process
- Buffer included for typical AWS-related overhead, domain setup, and minor iterations

Estimated Monthly Cloud Costs (MVP Level)

Given the MVP's limited usage, expected traffic will be low, primarily Tikvah team members, internal stakeholders, and occasional investor demos. Estimated cloud costs are modest:

- **Minimum setup** (free-tier Mongo, light traffic): ~$12–15/month
- **Typical MVP usage** (paid Mongo, full-time EC2): ~$20–25/month
- **Upper bound** (extra buffer, more DNS, always-on infra): ~$30–35/month

Why costs vary:

- **MongoDB tiers:** We'll start with the free tier (M0), which is sufficient for small data volumes and light usage. If performance or storage needs grow, we can easily scale to the M2 tier (~$9/month) within MongoDB Atlas. This upgrade is fully managed and can be done without code changes or downtime.
- **Always-on EC2 vs. on-demand:** Keeping infrastructure online ensures the MVP is accessible anytime without manual startup, which is ideal for demos and stakeholder reviews. Costs can be reduced if the server is paused between usage windows.
- **Infrastructure scaling:** Additional services like domain configuration, logging, or bandwidth are inexpensive but add marginally to the total, especially with always-on availability.

These choices are fully controllable and can be scaled up or down based on Tikvah's actual usage patterns over time.

Project Timeline & Budget

The MVP will be developed over two sequential phases. Each phase is structured to allow for early feedback, technical validation, and a smooth transition to a cloud-hosted demo environment. All estimates include built-in buffer time and reflect a development commitment of approximately 20 hours per week.

Development Phases

Phase	Description	Estimated Hours
Phase 1	Local-first MVP: backend services, provider search, basic UI	97
Phase 2	Cloud deployment, hosting, CI/CD, and documentation	79
Total		**176**

Budget Considerations

- The total estimate fits comfortably within the 186-hour budget, leaving **~10 hours of headroom** for minor additions or refinements.
- Buffers are applied across all epics using a 1.2x multiplier, providing **~29 hours of built-in flexibility** to accommodate scope adjustments or edge-case handling.
- Cloud infrastructure costs are expected to be **$20–25/month**, with detailed service-level estimates included in the Phase 2 section.

This phased, budget-conscious approach gives Tikvah a functional and extensible MVP while preserving space for iteration, polish, and growth.

Appendix

Effort Distribution by Epic

Epic	Description	Allocated Hours
Core Engineering	Backend API, database integration, search logic	68
UI Layer	Server-rendered templates for search and profiles	13
Cloud Infrastructure (includes documentation)	Hosting, deployment automation, domain setup, delivery docs	66
Total (before multiplier)		**147**
Buffer (1.2x multiplier applied)		**+29**
Final Total		**176**

Glossary of Terms

Term	Description
API	A structured interface that allows software systems to communicate with each other. In this case, it lets the web app interact with the backend services.
Backend	The part of the application responsible for storing, processing, and retrieving data—not visible to the end user.
CI/CD	Short for "Continuous Integration and Continuous Deployment," a set of tools and practices that automate testing and rollout of new code.
Docker	A tool that packages the application and its dependencies into isolated units for consistent deployment across environments.

EC2	Amazon's virtual server offering used to host the backend and frontend components of the MVP.
FastAPI	A modern Python web framework used to build the backend services.
Jinja2	A templating engine that generates HTML pages from backend data (used in the UI).
MongoDB Atlas	A cloud-based database platform used to store provider and organization data.
MVP	Minimum Viable Product—a simplified, functional version of the platform built to validate core functionality and direction.
Server-rendered UI	Web pages generated by the backend rather than a separate frontend application. Easier to build and deploy for MVP purposes.