CONFIDENTIAL

OCTOBER 26, 2025

Tech Due Diligence Report: MedicalCo.ai

Prepared for Investors & Founders of Micah 6 AI · Prepared by M&A Technology Consultant & AI Auditor, Micah 6 AI

This report reveals **critical-level technical debt, security vulnerabilities, and unaddressed algorithmic integrity issues** that pose existential threats to the business — particularly in the highly regulated medical domain.

Technical Health Scorecard

An at-a-glance overview of the system's health across seven critical dimensions. Scores are rated out of 5.

Category	Status	Score	Key Issues
Security	Critical Risk	1/5	Hardcoded API Keys, Over-Permissive Extension Permissions, Prompt Injection, Local File Storage, CORS Misconfiguration, XSS, Broken Access Control
Architecture	High Risk	2/5	Monolithic, Tight Coupling with AI Providers, Redundant Logic, Lack of API Gateway, Mixed Concerns
Code Quality & Practices	Medium Risk	2/5	Hackathon Origins, Magic Numbers, Inconsistent Naming, Dev Server in Prod, Direct DOM Manipulation, Incomplete Error Handling
Algorithmic Integrity (AI)	Critical Risk	1/5	Black-Box Opacity, Untested Bias, Arbitrary Model Parameters, No Explainability, No Bias/Fairness Evaluation Framework
Scalability & Performance	Medium Risk	2/5	Hardcoded max_instances: 1, Inefficient Table Bounding Box, Arbitrary Text Truncation, Dev Server Usage
Maintainability	Medium Risk	2/5	Redundant Logic, Outdated Dependencies, Hardcoded Logic, Minified Code without Source Maps, Brittle Date Parsing
Regulatory & Compliance	Critical Risk	1/5	High risk of non-compliance with HIPAA, GDPR, and ethical AI guidelines due to security flaws, data handling, and unmitigated bias
Overall Health	**Critically Unhealthy**	**1.6/5**	Fundamental security vulnerabilities, significant architectural debt, poor development practices, and severe unaddressed algorithmic integrity concerns

Construction Effort & Template Check

Estimated Hours to Build Current State

Based on observable complexity, excluding detected technical debt.

Role	Hours
Data Architect	40
Senior Backend Dev	200
ML/NLP Engineer	180
Fullstack Dev	120
Frontend Dev	150
DevOps Engineer	60
UI/UX Designer	50
Total	**800 hrs**

⚠ Template / Boilerplate Detection

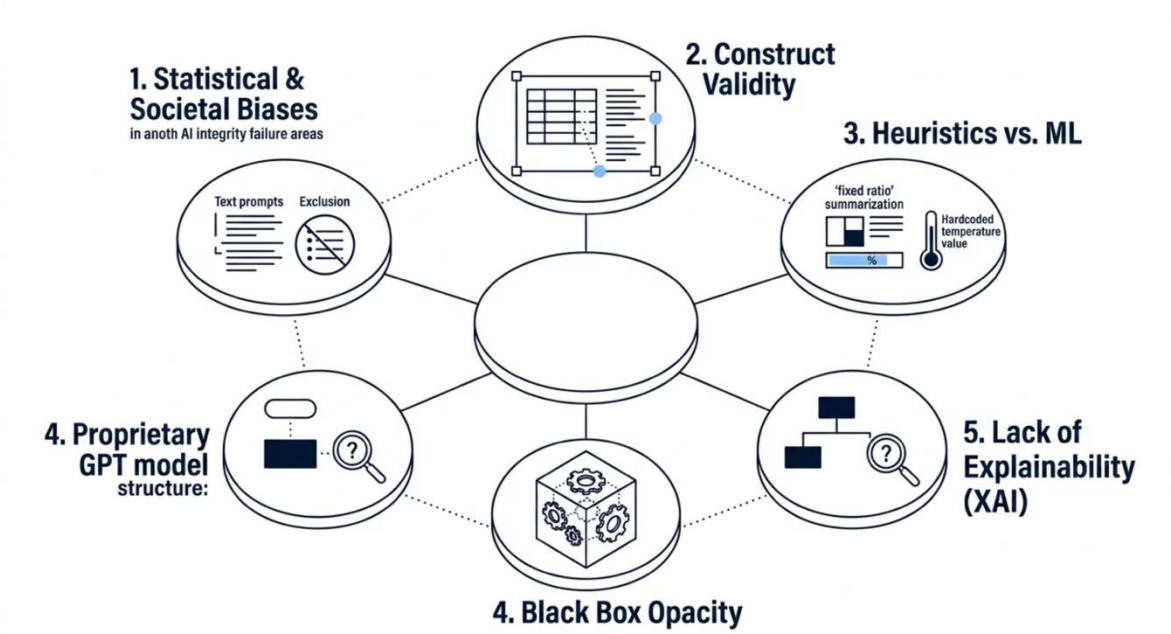
The frontend codebase **strongly appears to be a bought template or heavily derived from a popular starter kit.** Evidence includes:

- Direct mention of `black-dashboard-react.min.css` and `reactstrap.min.js`
- Loading of `bootstrap.min.css` (v4.2.1) from a CDN
- Presence of `demo.css`
- General structure aligning with common dashboard templates
- Hackathon origins context where templates are frequently used for rapid prototyping

⚠ A significant portion of the UI layer is **not original creation** and carries the inherent design and architectural decisions of the template.

Algorithmic Integrity & Bias: Overview

The core functionality of MedicalCo.ai relies heavily on LLMs and traditional NLP techniques within a highly sensitive medical insurance domain. The audit reveals **profound and unaddressed issues** regarding algorithmic integrity, bias, and explainability.



There is **no evidence of a systematic approach, framework, or methodology** implemented to test, measure, or mitigate algorithmic bias in the AI outputs. Without active evaluation, any inherent biases will go undetected and unaddressed.

Bias in Prompts, Exclusion Lists & Model Parameters

Hardcoded Prompt Bias

Prompt examples in `main.py` and resource files contain specific, sensitive medical scenarios such as *"diabetes was deemed self-inflicted"* and *"septoplasty deemed cosmetic."*

- No documentation on how examples were chosen
- No systematic approach to ensure fairness or diversity
- Risk of perpetuating historical biases in insurance denial patterns
- Could result in disproportionately negative outcomes for certain demographics

Poses severe ethical, legal (discrimination lawsuits), and reputational risks.

Biased Exclusion List in Summarizer

The hardcoded `excludeList` in `js-summarize.js` explicitly filters out critical medical terms like **"diabetes."**

- Directly impacts what the summarizer deems "important"
- Produces incomplete, misleading, and potentially dangerous summaries
- Pre-determines what information is irrelevant in a medical context

Arbitrary Temperature Settings

Temperature is set to `0.9` in `eventPage.js` and `0.33` in `main.py` — arbitrarily, without empirical tuning.

- High temperature (0.9) can generate hallucinated medical information
- Low temperature (0.33) may miss nuances in complex medical jargon
- Inconsistent behavior across different parts of the application

Black-Box Opacity & Lack of Explainability

> It's like having a doctor give a diagnosis without explaining their reasoning. In healthcare, this lack of transparency is unacceptable and potentially dangerous.

Black-Box LLM Reliance

Core intelligence relies on OpenAI's proprietary GPT-3/Davinci models. Internal logic, training data, and decision-making processes are entirely opaque. Micah 6 AI has **no direct control or visibility** into how the model arrives at specific outputs.

No Explainability (XAI)

There is **no mechanism or documentation** for understanding why the AI models generate specific summaries, Q&A responses, or appeal arguments. Users cannot verify reasoning, identify errors, or challenge unfair outcomes.

Untraceable Errors & Undetected Bias

Cannot diagnose why the AI made a mistake. Biases from training data or prompt engineering will operate undetected, leading to unfair or discriminatory outcomes with no means of identification or mitigation.

Legal & Regulatory Exposure

Operating in a medical domain without explainability violates ethical AI guidelines. Regulatory bodies (FDA, HIPAA) increasingly demand transparency for AI in healthcare. Severe legal repercussions if AI-generated output causes harm.

Critical Security Vulnerabilities

The system exhibits multiple critical-severity security flaws that represent immediate, existential threats to the business.

🔑 Hardcoded API Keys

OpenAI API keys (`sk-...`) are directly embedded in client-side JavaScript (`eventPage.js`, `index.js`). Publicly exposed and easily retrievable.

- Unauthorized billing by malicious actors
- Potential service interruption via key revocation
- 80 hrs to fix

🌐 Over-Permissive Extension

`manifest.json` requests broad host permissions (`http://*/*`, `https://*/*`) granting unfettered access to read data from any webpage visited.

- Massive privacy breach potential
- Direct HIPAA/GDPR/CCPA violation
- 60 hrs to fix

🚀 Prompt Injection

User-controlled input is directly concatenated into prompts sent to OpenAI models. No client-side validation present.

- Data exfiltration via crafted prompts
- Harmful content generation
- 70 hrs to fix

⚡ XSS Vulnerability

Frontend renders raw AI outputs and user messages into the DOM without HTML sanitization (`dangerouslySetInnerHTML` with untrusted input).

- Session cookie theft
- User credential compromise
- 50 hrs to fix

High-Severity Vulnerabilities

1

Local File Storage of Uploads

Uploaded PDF files are saved locally to `./uploads/` before being moved to Google Cloud Storage. Creates unnecessary copies of PHI in an unsecured location. Direct HIPAA violation. **40 hrs to fix.**

2

CORS Misconfiguration

`CORS(app)` initialized without specific origins configuration, potentially defaulting to allowing all origins (`*`). Enables cross-origin attacks and data leakage. **20 hrs to fix.**

3

Outdated Dependencies

Python 3.7, Flask 1.1.2, OpenAI 0.2.4, React 16.13.0, Bootstrap 4.0.0, Manifest V2. Known CVEs, compatibility issues, and maintenance burden. **120 hrs to fix.**

4

Broken Access Control

API calls to `/summary` and `/qa` use a `doc` hash parameter with no visible authorization mechanism. Attackers could enumerate hashes to access other users' medical documents. **60 hrs to fix.**

5

Hardcoded Backend URLs

Backend API URLs hardcoded in `Ask.js`, `Letter.js`, and `Summary.js`. Requires code changes for every environment change. Reveals internal server structure. **15 hrs to fix.**

Architecture & Scalability Risks

Monolithic Architecture & Tight AI Coupling

The system tightly couples with OpenAI's specific GPT-3/Davinci engine and API version (`openai==0.2.4`). Summarization logic is dispersed and redundant. The extension directly calls OpenAI.

- **Vendor Lock-in:** High dependency on a single AI provider
- **Upgrade Hell:** API upgrade from v0.2.4 to v1.x requires significant changes across multiple files
- **No Flexibility:** Cannot experiment with different AI models or specialized domain-specific models
- **Redundant Code:** Duplicated summarization logic leads to maintenance overhead and inconsistent behavior

Fix: Introduce an abstraction layer (`AIService` interface), centralize all AI calls through a backend API gateway, and enable configuration-driven AI selection. **80 hrs to fix.**

Scalability: max_instances: `1`

The `app.yaml` specifies a single instance — a single lane on a highway.

- Single point of failure
- Cannot handle traffic spikes
- Poor user experience at scale
- Impossible to serve 10k+ users
- Blocks third-party insurance sales

⚠ This configuration makes the stated business goals of scale **technically impossible** without immediate remediation.

Fix: Configure dynamic auto-scaling with appropriate min/max instances and a load balancer. **20 hrs to fix.**

Code Quality, Debt & Medium-Risk Issues

The project's "PennApps 2020" hackathon origins are evident throughout the codebase, creating a cracked foundation for any production system.

Hackathon Origins & Technical Debt

Rapid development practices led to hardcoding, lack of error handling, and outdated dependencies throughout. Accumulated debt slows future development, increases bug count, and makes the codebase harder to maintain. Experienced developers may be reluctant to work on such a project. **~100 hrs initial cleanup.**

Incomplete Error Handling

`axios.post` calls without comprehensive `try-catch` blocks, `json.loads` without error handling. Users get stuck on loading screens with no feedback. Unhandled exceptions can crash the server. Stack traces may be exposed, creating security vulnerabilities. **50 hrs to fix.**

Chrome Extension Manifest V2

Deprecated by Google — unsupported for existing extensions by June 2024. The extension will cease to function in the Chrome Web Store. Requires a full rewrite/migration to Manifest V3 with stricter content security policies and service workers. **80 hrs to fix.**

Low-Severity Issues

Missing API Key Defaults

`os.getenv()` calls use `""` as a default for `GPT_SECRET_KEY`, `GOOGLE_BUCKET_NAME`, etc. If environment variables are not set, the application attempts to use empty strings, leading to subtle authentication failures that are hard to diagnose.

Fix: Change to `os.getenv("VAR")` and raise a descriptive `ValueError` if not set. Implement startup checks. **10 hrs.**

Resource Leaks

File handles opened in `pagerank_summarizer.py` and `summarize_documents.py` are not explicitly closed or managed with `with` statements. Over time, open file handles exhaust system resources, leading to performance degradation and crashes.

Fix: Refactor all file operations to use `with open(...) as f:`. **5 hrs.**

Privacy Risk — Text-to-Speech

The Chrome extension uses `chrome.tts.speak(summary)` to audibly read sensitive medical summaries. In public or shared environments, private medical details could be accidentally broadcast to others.

Fix: Implement explicit user consent toggle, contextual warnings, and auditory/visual indicators when TTS is active. **10 hrs.**

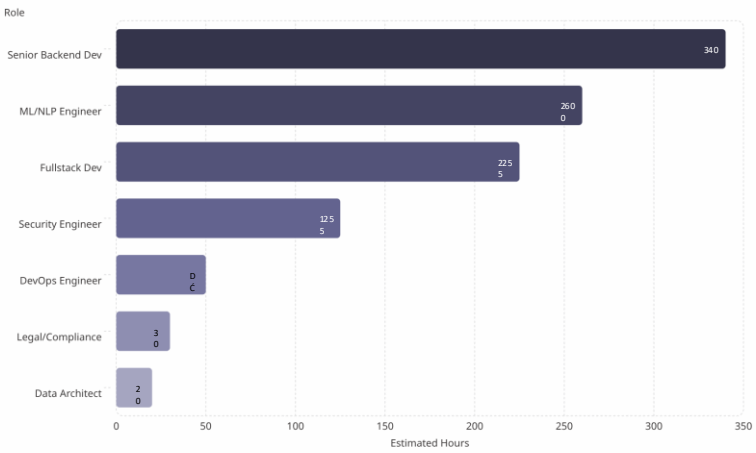
Hardcoded Logic — General

UI theme, API response parsing strings, UI layout dimensions, demo data, Gunicorn command, and Google Cloud region are all hardcoded. Requires code changes and redeployments for simple configuration adjustments. Makes the system brittle to external API format changes.

Fix: Externalize all configurable parameters into `.env` or `config.json` files. Use well-named constants for truly static values. **30 hrs.**

Total Remediation Effort

This summary covers estimated hours to remediate **Critical, High, and Medium** risk items only. Low-priority items and ongoing technical debt are excluded.



Total Remediation Summary

Role	Hours
Senior Backend Dev	340
ML/NLP Engineer	260
Fullstack Dev	225
Security Engineer	125
DevOps Engineer	50
Legal/Compliance	30
Data Architect	20
TOTAL	**1,050 hrs**

⚠ 1,050 hours ≈ **6–7 person-months** of dedicated expert work — and this excludes the deeper architectural overhaul and ongoing technical debt.

Final Recommendation: For SMBs & Founders

REFUSE WITH REVIEW OPPORTUNITY

While the product addresses a clear market need, the current codebase presents an **existential threat** to the business in the highly regulated medical domain.

💣 Security Catastrophe Awaiting

Hardcoded API keys, over-permissive Chrome extension, prompt injection, XSS, and local storage of medical documents are ticking time bombs. Massive data breaches, HIPAA/GDPR fines, and complete service shutdown are imminent risks.

⚖️ Ethical & Factual AI Risks

Complete lack of bias evaluation, arbitrary AI parameters, black-box opacity, and biased exclusion lists for medical terms are unacceptable. Could lead to discriminatory outcomes and factually incorrect summaries — directly undermining the product's core value proposition.

🏚️ Unsustainable Technical Debt

Hackathon origins are evident throughout. Monolithic architecture, outdated dependencies, and poor development practices make scaling impossible. `max_instances: 1` and no API gateway guarantee constant outages.

💰 High Remediation Cost

1,050 hours (~6–7 person-months) of dedicated expert work for critical and high-priority items alone — before accounting for the deeper architectural overhaul required to truly stabilize the system.

ⓘ **Review Opportunity Conditions:** A review may be considered *only if* founders commit to: (1) complete halt of new feature development, (2) immediate investment in full security and architectural re-platforming, (3) a clear funded roadmap to audit and mitigate algorithmic bias with expert oversight, and (4) a transparent plan toward a sustainable development lifecycle.

Final Recommendation: For Investors (VC/PE/Angels)

STRONG PASS

Build vs. Buy Analysis

Given the extensive critical vulnerabilities, significant technical debt, and profound ethical/accuracy issues, the recommendation is to **BUILD from scratch rather than BUY this codebase.**

- **Cost of Repair > Cost of Build:** 1,050+ hours of remediation plus fundamental architectural redesign likely exceeds the cost of a new, well-architected system
- **Risk Mitigation:** Starting fresh enables modern security best practices, scalable architecture, robust MLOps, and a bias-aware AI development lifecycle from day one
- **Talent Attraction:** A greenfield project is significantly more attractive to top-tier engineering talent than inheriting a deeply problematic codebase

Intellectual Property Value Assessment

20%

Uniqueness

Concept is unique; implementation relies on off-the-shelf AI services and boilerplate templates

15%

Robustness

Demonstrably fragile due to hardcoded logic, monolithic architecture, and outdated dependencies

10%

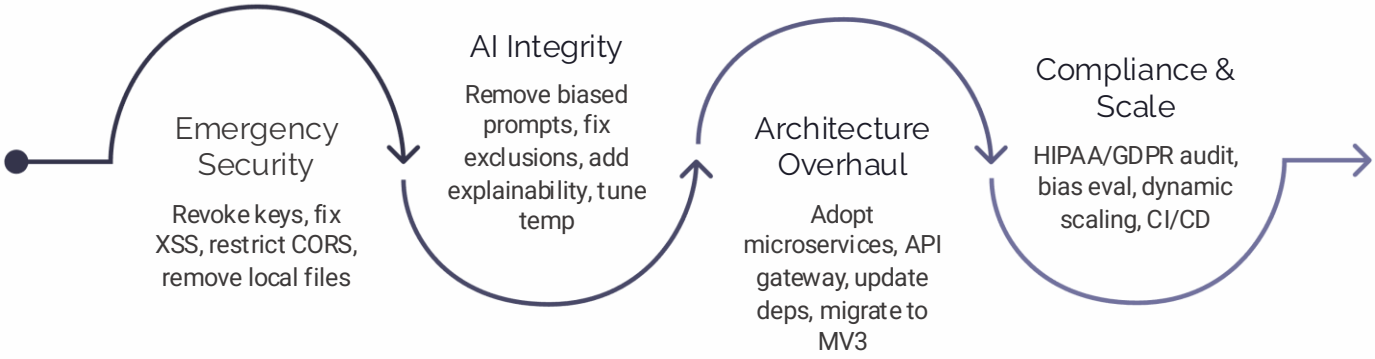
Defensibility

Weakly defensible — poor code quality, heavy reliance on OpenAI's IP, no unique algorithms

> Any investment must be treated as a **seed investment for a completely new build**, leveraging the founding team's vision and market insight — but not their existing technical implementation. Investment must be contingent on complete re-platforming and adoption of stringent security, ethical AI, and engineering best practices.

Remediation Roadmap

A prioritized path from critically unhealthy to production-ready. Each phase must be completed before advancing to the next.



This four-phase approach prioritizes the most dangerous vulnerabilities first, ensuring the system is secured before architectural improvements are layered on top. Regulatory compliance and true scalability are achievable only after the foundational work is complete.

Need Human Validation?

Don't let algorithms alone decide your company's fate. Book a bespoke Human-in-the-Loop Tech Due Diligence Audit with our executive data architects.

Visit Micah 6 A

Evaluating a Second Sprint or New Acquisition?

Access the audit platform and use code `WELCOMEBACK20` at checkout for **20% off** your next automated audit.

Access Audit Platfor

This report was generated via the Micah 6 AI Audit Engine. Its content is AI-generated and should be validated by human engineers.