Anne Burnett Marion School of Medicine at Texas Christian University

DiseaseQuest
Software Requirements Specification

Version 1.0

Revision History

Date	Version	Description	Author
10/21/2025	1.0	Initial Revision	DiseaseQuest Team
04/15/2026	2.0	Finalized Version	DiseaseQuest Team

Table of Contents

Software Requirements Specification

1. Introduction

1.1 The Purpose of DiseaseQuest

Medical students that are working through their clinical studies in the Burnett School of Medicine are wanting to work more asynchronously with the class curriculum. Their current asynchronous study option is only allowing them to study strictly knowledge-based concepts, and want a way to study a full end-to-end case study that will test the clinical reasoning of the student rather than general concepts. This will be done through a patient-centered dialogue. Students that learned through gamified-features believed they were more engaged in the material. The gamified solution, DiseaseQuest, is not meant to replace the curriculum, but rather act in tangent with it to be supplementary material to learn.

1.2 The Purpose of this Document

The purpose of this document is to describe the functional and nonfunctional requirements for software release 1.0 of DiseaseQuest. This document serves as a comprehensive reference for the software project's requirements, providing a clear and detailed understanding of what the system is expected to achieve. It is intended to align stakeholders, developers, and testers by defining the scope, functionality, and constraints of the system. By consolidating all requirements into one structured document, it ensures consistency, reduces ambiguity, and acts as a foundation for development, testing, and future maintenance.

1.3 References

1. DiseaseQuest Glossary: URL
2. DiseaseQuest Vision and Scope Document: URL
3. DiseaseQuest Use Cases and Business Rules: URL
4. DiseaseQuest Software Architecture: URL
5. DiseaseQuest Prototype: URL
6. DiseaseQuest API Document: URL
7. DiseaseQuest Business Domain Model: URL
8. DiseaseQuest MVP Presentation: URL

2. Project Glossary

The project glossary is available here: URL

3. Vision and Scope

The vision and scope document is available here: URL

4. Software Architecture

The UML diagram for DiseaseQuest is available here: URL.

4.1 System Context Diagram

DiseaseQuest serves as the central platform, enabling admin/instructors to manage classrooms, students, and playable cases, while students use it to play through cases and view their scores and progress. The system integrates with an email system to send automated email notifications for platform registration to Instructors/Students. The system also integrates with OpenAI Agents for realistic simulated patient/doctor interactions in cases.

4.2 Container Diagram

The DiseaseQuest system is composed of three key containers: the **Web Application**, the **Agent API Application**, and the **Postgres Database**, all supported by integration with the **Mail System** for email communication. The **Web Application**, built with Nuxt, is delivered to students' browsers to provide a user-friendly interface for case playthrough and manages interactions with the **Postgres Database** for user authentication and data retrieval/storage. The **Agent API Application**, implemented using Typescript and **Supabase** edge functions, handles backend agent functionality, handling game flow and managing interactions with the **Postgres Database**. The **Postgres Database**, a relational database, stores critical data, such as user information, case information, and state context for the **Agent API Application**, while CRUD operations are executed through the Web Application. Additionally, the Web Application integrates with the **Mail System** using SMTP to send automated registration notifications.

4.3 Operating Environment

OE-1: DiseaseQuest shall operate correctly on all supported web browsers.

4.4 Design and Implementation Constraints

CO-1: All pull requests must be approved by at least one other team member before the request can be merged with the main branch.

4.5 Assumptions and Dependencies

DE-1: The operation of DiseaseQuest depends on the OpenAI agents to run the game flow.

5. Functional Requirements

5.1 Use Cases

The use cases are available here: URL.

6. Business Rules

The business rules are available here: URL.

7. Data Requirements

7.1 Business Domain Model

The domain model is available here: URL

7.2 Data Acquisition, Integrity, Retention, and Disposal

DI-1: DiseaseQuest shall retain data from inactive students for 1 year following the last case completion date.

DI-2: DiseaseQuest shall retain data from expired classroom for 1 year following the end date.

8. External Interface Requirements

8.1 User Interfaces

The Prototype can be found here: URL

UI-1: Nuxt.js Front-End Application

UI-1.1: The front end shall communicate with the Nuxt.js server back end using REST endpoints exposed under /server/api.

UI-1.2: The front end shall render application pages (dashboards, classrooms, case lists) using Nuxt's routing and component lifecycle system.

UI-1.3: TailwindCSS (and SCSS if required) shall be used to style all user-facing pages and components.

UI-2: Visualization & Component Libraries

UI-2.1: TailwindCSS shall be used to style all user-facing pages and components.

UI-2.2: The UI shall use shadcn-vue components to build dashboards, student interfaces, instructor tools, and interactive controls.

UI-2.3: Component communication shall use standard Vue/Nuxt reactive bindings for state, props, and event emission.

8.2 Software Interfaces

SI-1: Supabase Platform (Authentication, Database, Storage)

SI-1.1: The system shall communicate with **Supabase Authentication** to create, verify, and manage user accounts using email-based magic-link registration.

SI-1.2: The system shall transmit invitation emails through the **Resend email provider**, generating secure invitation links for new users.

SI-1.3: The system shall store and retrieve classroom data, patient cases, user identities, and activity logs through **Supabase Postgres** using REST or RPC calls provided by Supabase's client libraries.

SI-2: OpenAI API (Agentic Workflow)

SI-2.1: The system shall send patient-case data and query context to the OpenAI API to perform reasoning, scoring, feedback generation, or simulation-based evaluation.

SI-2.2: The system shall receive structured responses from the Agent API in JSON or text form and map them to application data models (e.g., score objects, rubrics, error feedback).

SI-2.3: All data sent to the Agent Platform shall be stripped of personal identifiers to maintain FERPA/HIPAA compliance.

SI-3: Vercel Deployment & Nuxt Server API

SI-3.1: The Nuxt.js server API shall communicate with Vercel hosting for automatic deployment and CI/CD pipelines.

SI-3.2: The Nuxt server runtime shall interface with Supabase using server-side API calls, ensuring no secret keys are exposed to the client.

SI-3.3: Server API endpoints shall return JSON responses to the front end, including user authentication status, classroom data, case content, and AI-generated evaluations.

SI-4: Email Delivery (Resend)

SI-4.1: Invitation links and authentication emails shall be generated and sent using Resend email provider.

SI-4.2: No email attachments shall be sent by the system to maintain security and avoid PHI/PII exposure.

8.3 API Document

The API document is available here: URL.

8.4 Communications Interfaces

CI-1: DiseaseQuest shall use Resend email delivery service to send invitation links to users, and all invitation emails shall be restricted to valid ".edu" addresses (per BR-1).

CI-2: Invitation emails shall contain a unique, single-use, time-limited link that expires exactly 1 week after being issued (per BR-2). Expiration shall be handled using Supabase's built-in token TTL.

CI-3: The system shall notify instructors via email or in-platform alerts when: a classroom is nearing expiration (BR-4), a classroom has successfully expired, or a classroom has been reactivated (BR-5).

9. Quality Attributes

9.1 Usability

USE-1: The system shall provide contextual guidance indicating which clinical reasoning step the student is currently performing.

USE-2: Users shall be able to revisit any previously collected patient information within a given case.

USE-3: The classroom enrollment workflow shall make clear whether the invite link is still valid or has expired.

USE-4: The system shall present clear indicators of classroom membership status.

9.2 Performance

PER-1: 95% of patient dialogue responses shall be generated within 3 seconds.

PER-2: The system shall support 500 total users and 150 concurrent users without performance degradation.

PER-3: Case summaries and DDx scoring shall load in under 5 seconds.

PER-4: Classroom creation or expiration routines shall complete within 2 seconds.

PER-5: Invitation email dispatch shall occur within 10 seconds after an instructor initiates a send request.

9.3 Security

SEC-1: All communications involving personal data, performance metrics, or classroom membership shall be secure end-to-end.

SEC-2: Only users with valid ".edu" email domains shall be eligible to receive invite links.

SEC-3: Invitation links shall expire exactly one week after being generated.

SEC-4: Only administrators may upload patient cases, and all uploaded cases must comply with FERPA/HIPAA de-identification requirements.

SEC-5: Students shall only access patient cases associated with the classrooms they are enrolled in.

SEC-6: Each classroom must always have at least one instructor assigned, and access control shall enforce this rule.

SEC-7: Instructor and admin access shall time out after 15 minutes of inactivity.

SEC-8: System logs shall exclude personally identifiable information.

9.4 Safety

SAF-1: All medical content and AI-driven reasoning shall be reviewed for accuracy prior to release.

SAF-2: All patient cases shall remain fully synthetic or de-identified in compliance with FERPA/HIPAA.

SAF-3: All simulated results shall be clearly labeled as educational and not for real clinical use.

SAF-4: No personal student data shall appear in case transcripts, feedback summaries, or shared results.

9.5 Availability

AVL-1: DiseaseQuest shall maintain ≥99% availability from 6 A.M.–midnight local time.

AVL-2: Classrooms that are reactivated shall become accessible within 1 minute of instructor request.

AVL-3: The system shall notify users at least 12 hours in advance of scheduled maintenance.

9.6 Robustness

ROB-1: User progress in a case shall be recoverable after an interruption.

ROB-2: If an invite link is expired or invalid, the system shall present a clear recovery path to request a new link.

ROB-3: If a classroom expires, students shall lose access to cases immediately but retain personal data and progress logs for auditing.

ROB-4: Malformed case uploads shall be rejected with clear validation errors.

ROB-5: No user action shall be able to circumvent class-based access restrictions.

9.7 Maintainability

MAI-1: The system shall allow admins to update or replace patient cases without requiring redevelopment.

MAI-2: Instructors shall be able to reactivate classrooms or resend invites without administrative support.

9.8 Scalability

SCA-1: The platform shall support up to 1,000 concurrent users during the Scale Study phase.

SCA-2: The system shall handle up to 2,000 active invite links across institutions.

10. Deployment

DiseaseQuest will be deployed as a cloud-hosted web application using a serverless / managed-services architecture to minimize operational overhead and support elastic scaling.

10.1 Deployment Architecture

DEP-1: The frontend and backend web application shall be implemented as a NuxtJS application written in TypeScript and deployed on Vercel as a server-side / edge rendering where appropriate.

DEP-2: Application data (e.g., users, classrooms, cases, scores, progress) shall be persisted in a managed PostgreSQL database hosted via Supabase.

DEP-3: AI-driven interactions (simulated patient dialogue, reasoning feedback, summaries) shall be handled by an Agentic workflow that integrates Supabase Edge Function services with the OpenAI API.

DEP-4: All environment-specific configuration (API keys, database URLs, SMTP credentials, etc.) shall be managed as environment variables in the deployment platform(s) and never hard-coded in the source code or client bundle.

10.2 Operations & Monitoring

DEP-5: Application logs from Nuxt server routes, agent services, and database query logs shall be centralized and viewable by the development team for debugging and auditing.

DEP-6: Basic health checks (uptime, error rate, AI API latency, database connection status) shall be monitored, and alerts shall be configured for thresholds that threaten availability requirements AVL-1 through AVL-3.

DEP-7: Regular backups of the PostgreSQL database shall be configured via Supabase according to institutional policy, with a minimum retention period of 30 days.

11. Internationalization and Localization Requirements

LR-1: The DiseaseQuest platform is intended to be used within the United States and uses American English.

12. Other Requirements

12.1 Legal & Regulatory Requirements

OR-1: The system shall display a clear disclaimer on all case-related pages stating that DiseaseQuest is an educational simulation and must not be used for real clinical diagnosis or treatment decisions.

OR-2: All third-party services (OpenAI, Supabase, SMTP provider) must be used under institution-approved data handling and security agreements.

12.2 Operational Constraints

OR-3: The system shall degrade gracefully if OpenAI services become temporarily unavailable, giving clear error messages without corrupting case progress.

OR-4: Institutional branding (logo, colors, official name of Burnett School of Medicine) must follow university branding guidelines when displayed in the UI.