SE 4485: Software Engineering Projects

Fall 2025

Requirement Documentation

Group Number	Group 4
Project Title	Intelligent EMR Note Generation Service
Sponsoring Company	[REDACTED]
Sponsor(s)	[REDACTED] [REDACTED]
Students	1. Samar Siddiqui – Team Lead 2. Kunuth Siddiqui 3. Pedro Garcia 4.Tammy Tran 5. Nashrah Siddiqui 6. Ziyad Alsoudani

Project Title: Intelligent EMR Note Generation Service
Course: SE 4485 – Software Engineering Project
Term: Fall 2025
Company Sponsor: [REDACTED]
Corporate Mentor: [REDACTED]
Business Contact: [REDACTED]
University: The University of Texas at Dallas
Team Size: 6 Students

ABSTRACT
This Requirements Documentation explains the functional and non-functional requirements for the Intelligent EMR Note Generation Service, sponsored by [REDACTED]. This project enhances the Behavioral Health CareChain (BHCC) Simulator by including AI-powered clinical note generation. There are two primary functions: manual generation started by clinicians, and automated generation via scheduled jobs.

The functional requirements include user interaction via a React-based UI, note generation using RESTful APIs, and integration with an external large language model (LLM). Non-functional criteria cover performance, dependability, usability, scalability, and maintainability. The publication also includes use case models, justification, and relevant engineering standards.

This specification will guide the system's design, development, and validation to ensure that it meets project goals, is clinically accurate, and integrates with the sponsor's BHCC Simulator.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Purpose and Scope

The Intelligent EMR Note Generation Service improves the sponsor's Behavioral Health CareChain (BHCC) Simulator by integrating AI-powered clinical note generation. The target of this system is to reduce manual documentation work while making sure that generated notes are accurate, reliable, and integrated with the simulator. The scope includes both human note generation and automated note generation.

Background

The BHCC Simulator currently provides mock patient data and supports manual entry of notes. While this enables testing of EMR workflows, the manual entry process is time-consuming and does not scale well for large patient populations.

Proposed Solution

The Note Generation Service will introduce:

- Manual AI Note Generation: A clinician selects a patient, provides minimal input, and generates an AI-powered note.
- Automated Note Generation: Scheduled jobs produce notes for multiple patients, suitable for large-scale simulation.
- User Interaction Features: Clinicians can preview, accept, discard, or regenerate AI-generated notes.
- System Integration – The service will integrate with the BHCC Simulator API and external AI models (LLMs).

Document Structure

This document includes the following sections:

- Use Case Model – Functional requirements modeled through diagrams and textual descriptions.
- Rationale – Explanation of how the use case model reflects project goals.
- Functional and Non-Functional Requirements – Testable statements defining system behavior and constraints.
- Configuration Management – Version control evidence for requirements documents.
- Engineering Standards – Relevant IEEE/ISO standards applied to the project.
- References – Supporting academic and industry sources.

USE CASE MODEL FOR FUNCTIONAL REQUIREMENTS

Graphic Use Case Model

- Actors:
- Clinician (Nurse/Doctor/Staff)
- Scheduler (System Actor, automated job)
- EMR Simulator APIs (external system)
- AI Note Generation Service (system component)
- Use Cases
- Generate Manual Note
- Generate Automated Notes (Batch)
- Review & Accept/Regenerate Note

- Save Note to Patient Record

Textual Descriptions

- Use Case 1: Generate Manual Clinical Note
- Actors: Clinician, AI Note Generation Service, EMR Simulator API
- Entry Condition: Clinician selects patients and note type in the User Interface.
- Normal Flow:
- Clinician accesses React-based UI
- Clinician enters minimal inputs (patient state, summary).
- System integrates with BHCC APIs to gather patient context.
- System constructs prompt and sends to AI service (Gemini).
- AI generates structured notes and returns it.
- System displays notes in preview area.
- Exit Condition: Clinician accepts note → saved to patient's record.
- Exceptions
- AI fails to generate response → error messages.
- Invalid patient ID → validation error.
- Special Requirements
- Must return response within 2-8 seconds for usability.

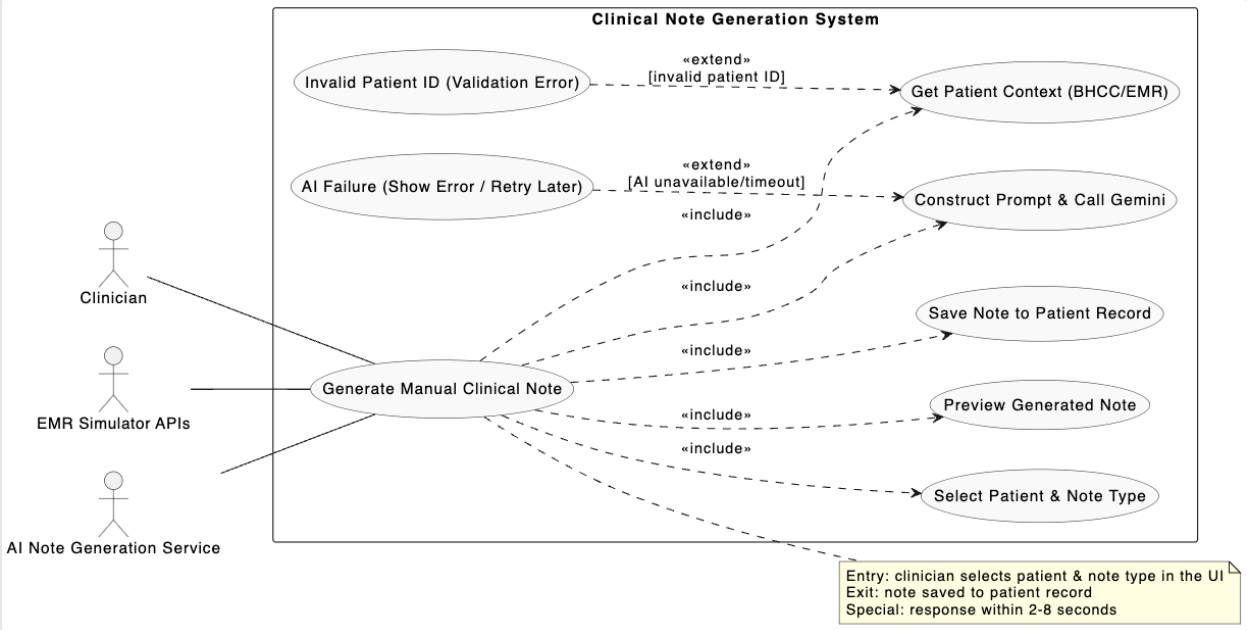

Figure 1.1. Use Case 1: Generate Manual Clinical Note

- Use Case 2: Generate Automated Notes (Batch)
- Actors: Scheduler, AI Note Generation Service, EMR Simulator API
- Entry Condition: Scheduled job is triggered (daily run or on-demand).
- Normal Flow:
- Scheduler requests list of eligible patients from EMR Simulator API.
- For each patient, system selects note type.
- AI Note Generation Service creates notes using patient history.
- Notes are automatically saved into EMR system.
- Exit Condition: All eligible patients have new notes added.
- Exceptions:
- API calls fail → skipped patient logged.

- AI service downtime → job retries next cycle.

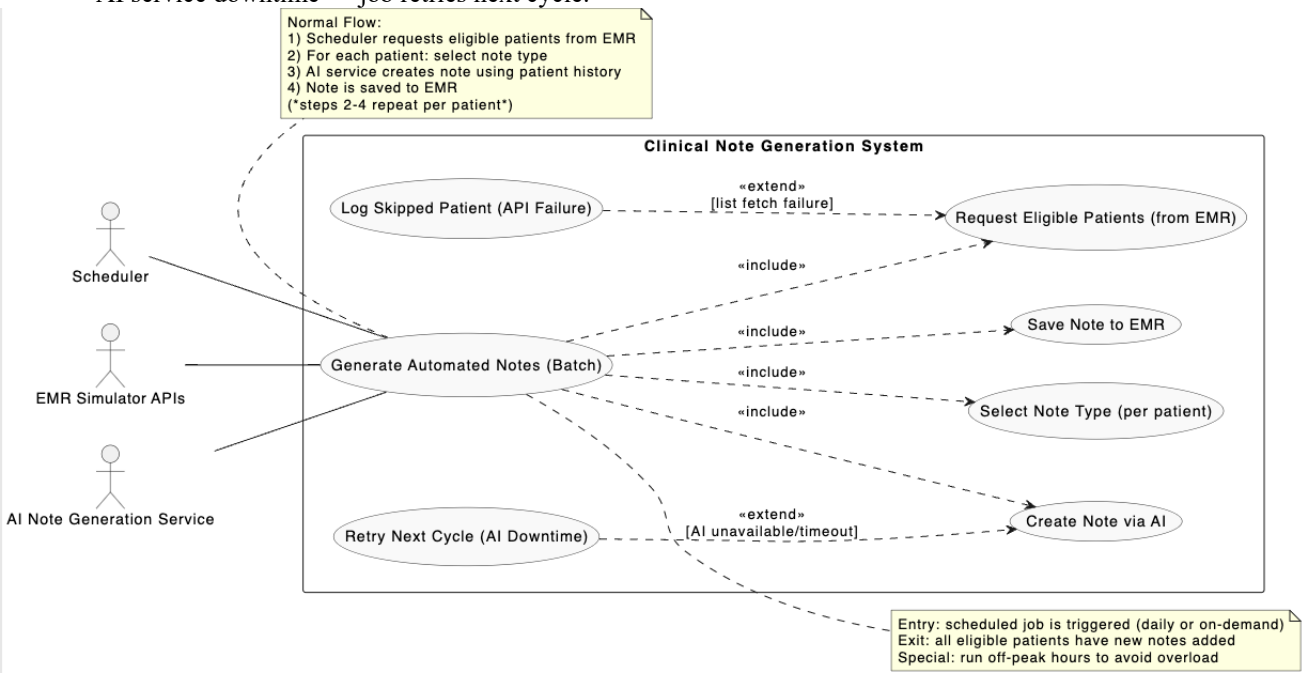

Figure 1.2. Use Case 2: Generate Automated Notes (Batch)

Use Case 3: Review & Accept/Regenerate Note

- Actors: Clinician, AI Note Generation Service
- Entry Condition: A generated note is displayed in the UI.
- Normal Flow:
- Clinician reviews the note preview.
- Clinicians choose one of: Accept, Discard, or Regenerate.
- If “Accept,” note saved. If “Regenerate,” AI service is called again.
- Exit Condition: A final note is saved to EMR or discarded.
- Exceptions:
- Multiple regeneration attempts exceed limit → system prompts manual entry.

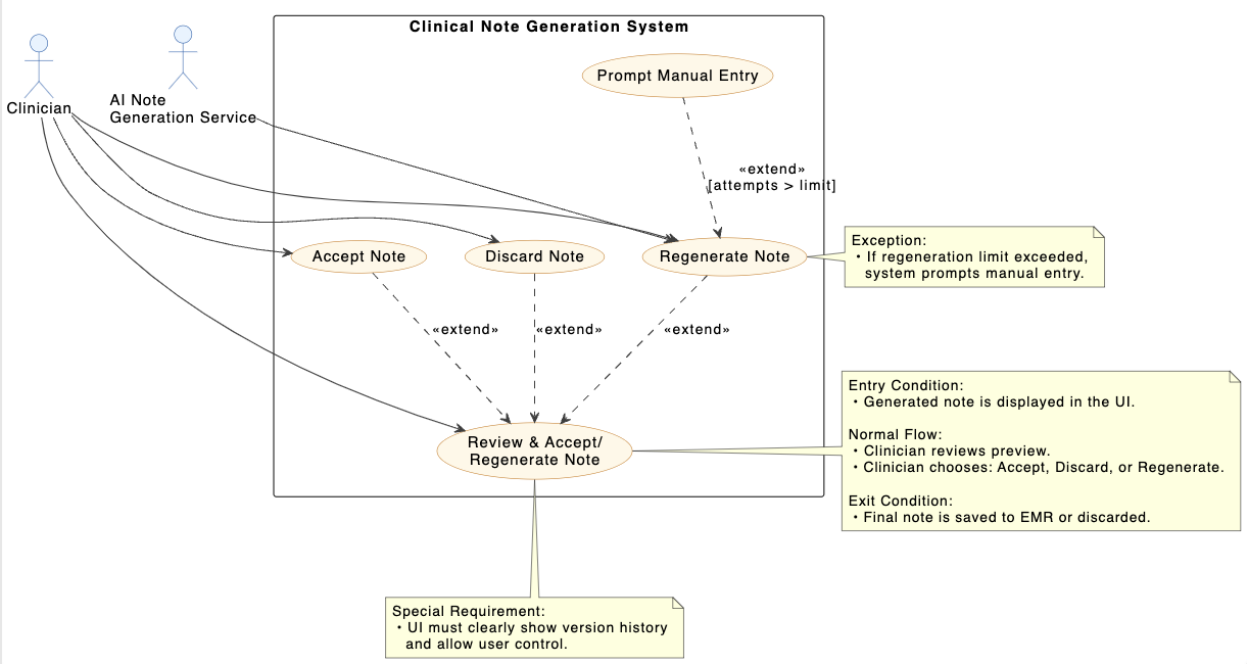

Figure 1.3. Review & Accept/Regenerate Note

Use Case 4: Save Note to Patient Record

- Actors: Clinician, AI Note Generation Service, EMR Simulator API
- Entry Condition: A final clinical note has been generated (manual or automated) and is ready for storage.
- Normal Flow:
- Clinician confirms acceptance of a generated note (manual mode) or the scheduler completes automated batch note generation.
- The AI Note Generation Service formats the note according to EMR standards.
- The system sends a note to the EMR Simulator API.
- EMR Simulator API validates the note and stores it in the patient record.
- System confirms the successful storage to the clinician or logs completion in batch mode.
- Exit Condition: The note is permanently stored in the patient's record in the EMR system.
- Exceptions:
- If the EMR Simulator API is unavailable → system retries or logs an error.
- If the patient record is invalid or missing → error message returned, note not stored.
- Special Requirements:
- Notes must be saved securely using standard encryption (HTTPS/TLS).
- Each stored note should have a unique ID so it can be tracked later.

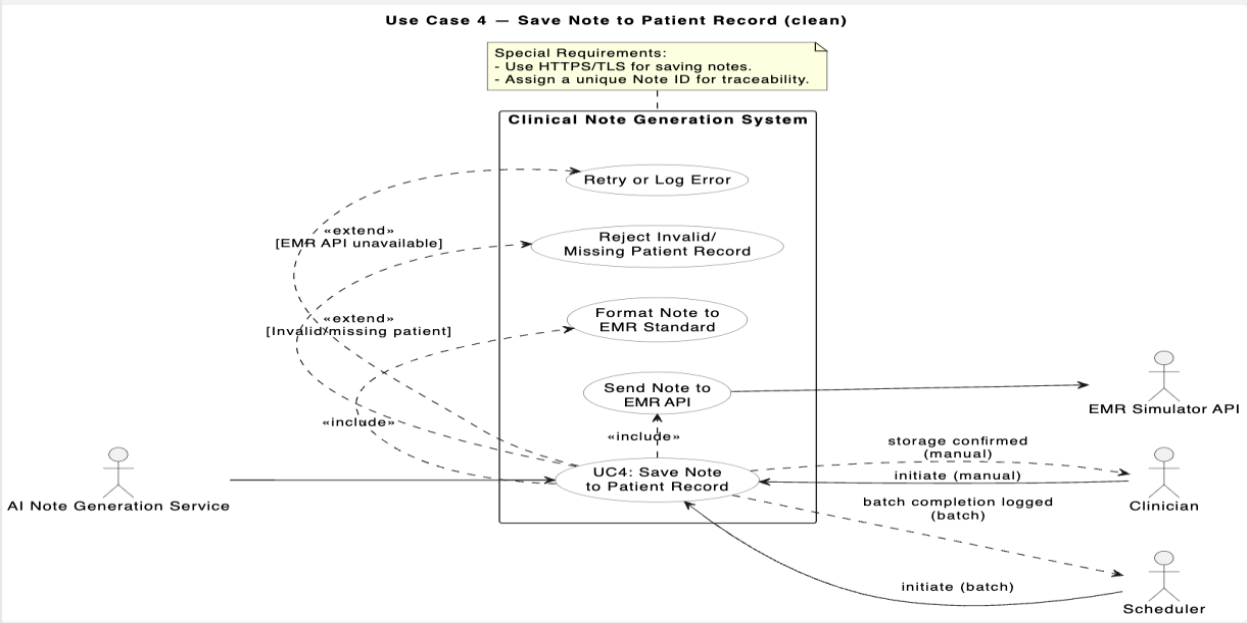

Figure 1.4. Use Case 4: Save Note to Patient Record

RATIONALE FOR YOUR USE CASE MODEL

The use cases were selected because they describe the system's core objectives and stakeholder needs. The Intelligent EMR Note Generation Service is used to reduce clinician labor while providing automated documentation.

- Generate Manual Clinical Note is the fundamental functionality for end users (clinicians) who need to take quick, accurate notes with little effort.
- Generate Automated Notes (Batch) addresses the need for scalability by allowing the system to generate notes for several patients without operator intervention.
- Review & Accept/Regenerate Note allows that doctors maintain control and can check or refine AI outputs before they are stored in the EMR.

These use cases align with the project's two objectives: (1) improving clinician usability and efficiency, and (2) enabling automated, large-scale simulations using the sponsor's BHCC platform. They work together to create a balanced model that allows for both human involvement and automated procedures, while also maintaining reliability and control over AI-generated clinical information.

Table 1.1 - Use Case Summary Table

Use Case	Name	Actors	Trigger	Outcome	Expectation
UC-1	Generate Manual Clinical Note	Clinician, AI Service, EMR API	Clinician initiates note	Note displayed and saved to patient record	AI fails to respond / Invalid ID
UC-2	Generate Automated Notes (Batch)	Scheduler, AI Service, EMR API	Scheduled job	Notes auto generated for patients	API failure / AI downtime
UC-3	Review & Accept/Regenerate Note	Clinician, AI Service	Note displayed in UI	Accepted or regenerated note saved	Regeneration attempts exceed limit
UC-4	Save Note to Patient Record	Clinician, EMR API	Finalized note	Stored in EMR database	Database error

NON-FUNCTIONAL REQUIREMENTS

- NFR-1 (Accuracy): Clinical notes that are generated by AI must be structured and validated against defined medical standards to ensure accuracy.

- NFR-2 (Scalability): Must support automated generation of notes for at least 1,000 patients per batch without system failure.
- NFR-3 (Performance): Generate and display manual clinical notes within 2 seconds of user request under normal operating conditions.
- NFR-4 (Reliability): Achieve 99% uptime during scheduled runs, excluding planned maintenance.
- NFR-5 (Usability): The UI shall provide intuitive, clearly labeled options (Accept, Discard, Regenerate) for managing AI-generated notes.
- NFR-6 (Maintainability): All source code should follow consistent coding standards and include inline documentation for future maintenance.
- NFR-7 (Traceability): All requirements, test cases, and changes must be tracked in GitHub version control.
- NFR-8 (Security): API keys and patient data shall be transmitted using secure protocols (HTTPS/TLS 1.2 or higher).

EVIDENCE THE DOCUMENT HAS BEEN PLACED UNDER CONFIGURATION MANAGEMENT
Configuration management will be handled through GitHub for each deliverable. For simultaneous collaboration, GitHub will be used to identify differences between two consecutive versions. For large changes/official submissions, GitHub will be used to maintain the version number of each document and check in/check out major changes on deliverables. Each document will have a designated version number (e.g. project_plan.pdf v1.0.0).

Table 2.1 Configuration Management Tool

Version	Date	Change Type	Description of Changes	Difference Highlights	Reviewers
v1.0.0	2025-09-24	Added	Requirements Documentation: Creation of initial draft.	Initial commit.	@presto21 (Pedro G.) — Ship-It v @TammyTran (Tammy T.) — Ship-It
v1.1.0	2025-09-26	Fixed	Requirements Documentation: Removed section for functional requirements as it was not a necessary for this document and removed tables as they were redundant.	-2 tables -1 section	@presto21 (Pedro G.) — Ship-It @TammyTran (Tammy T.) — Ship-It

ENGINEERING STANDARDS AND MULTIPLE CONSTRAINTS

- IEEE Std 830-1998: Software Requirements [pdf]
- IEEE Std 29148: Requirements Engineering [pdf]
- ISO/IEC/IEEE Std 29148-2018: Systems and Software Engineering
  - Life Cycle Processes
  - Requirements Engineering [pdf]

ADDITIONAL REFERENCES

- Lamsweerde, A.V., 2009. *Requirements Engineering: From System Goals to UML Models to Software Specifications.* John Wiley