SE 4485: Software Engineering Projects

Fall 2025

Project Management Plan

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. Nashrah Siddiqui 4. Tammy Tran 5. Pedro Garcia 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 document presents the project plan for the Intelligent EMR Note Generation Service, sponsored by [REDACTED]. The system will enhance [REDACTED]'s already existing [REDACTED] Simulator Application by adding an AI-powered note generation capability. The project aims to integrate a large language model (LLM) when developing the Note Generation Service to create clinically accurate notes either on demand or through an automated process. Deliverables include: a React-based front end, RESTful back-end service, AI integration module, scheduling component, a complete testing framework, documentation, and live demonstration. In addition, the plan also sets out the team's structure, development lifecycle, risk analysis, required resources, project schedule, progress-tracking methods, and the professional standards the team will follow.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Purpose and Scope:

The purpose of the project is to design, develop, and implement an Intelligent EMR Note Generation Service that will be integrated with [REDACTED]'s [REDACTED] Simulator. Creating manual and automated note generation features, enhancing user interface, building back-end services, integrating an AI model, and developing scheduling and testing frameworks are all included in the scope.

Product Overview:

- Purpose: Minimize manual effort for clinicians by generating notes automatically and reduce reliance on manual note entry.
- Capabilities: Manual/Automatic AI note generation, batch scheduling, prompt engineering library, integration with [REDACTED] APIs.
- Scenarios: A manual scenario would be a nurse may want to generate a progress note from minimal inputs; automated jobs may generate daily notes for multiple patients.

Document Structure:

This document has multiple sections covering project organization, lifecycle model, risk analysis, resource requirements, schedule, monitoring, and professional standards.

PROJECT ORGANIZATION

- The development team is divided into six roles.
    - Front-End Developer: Builds React UI enhancements.
    - Back-End Developer: Implements REST APIs and persistence logic.
    - AI Integration Engineer: Designs prompt library and LLM integration.
    - Scheduler Engineer: Builds automation and batch features.
    - QA & Testing Engineer: Develops test suites.
    - Project Manager: Oversees schedule, documentation, and communication.
- Dependencies:
    - UI depends on back-end APIs.
    - Automation depends on AI integration.
    - Testing depends on completed features.

Rationale: Everyone will be involved in the development process, but it is also important to designate someone to take the lead on a specific task. This division is in alignment with the project's deliverables and ensures ownership while also promoting collaboration.

Table 1.1 Team Roles and Responsibilities

Role	Primary Responsibilities	Rationale
Front-End Developer (UI Enhancements)	● Implement React-based UI changes (buttons, preview pane, loading indicators) ● Integrate with backend APIs ● Ensure usability and responsiveness.	UI needs are small but critical; having a dedicated member seamless user interaction.
Back-End Development (REST API & Service Logic)	● Build and maintain REST APIs endpoints ● Implement data persistence logic ● Handle errors and logging	Core of the system; needed for integration between UI, AI, and scheduler.
AI Integration Engineer (Prompt Engineering)	● Design prompt templates for each note type ● Call LLM APIs securely ● Parse and validate AI responses	Ensures accurate, structured, and clinically relevant AI-generated notes.
Scheduler & Automation Engineer	● Develop daily and on-demand batch job feature ● Manage scheduling logic for scalability ● Ensure automation reliability	Adds scalability and realism to the simulator; reduces manual workload.
QA & Testing Engineer	● Write unit, integration, and functional tests ● Validate AI outputs and system interactions ● Ensure performance and reliability	Critical for catching errors early and meeting quality standards.
Project Manager	● Coordinate schedule and team communication ● Manage sponsor updates and mentor meetings ● Oversee documentation and reporting	Keeps the team aligned, ensures timely delivery, and maintains project records.

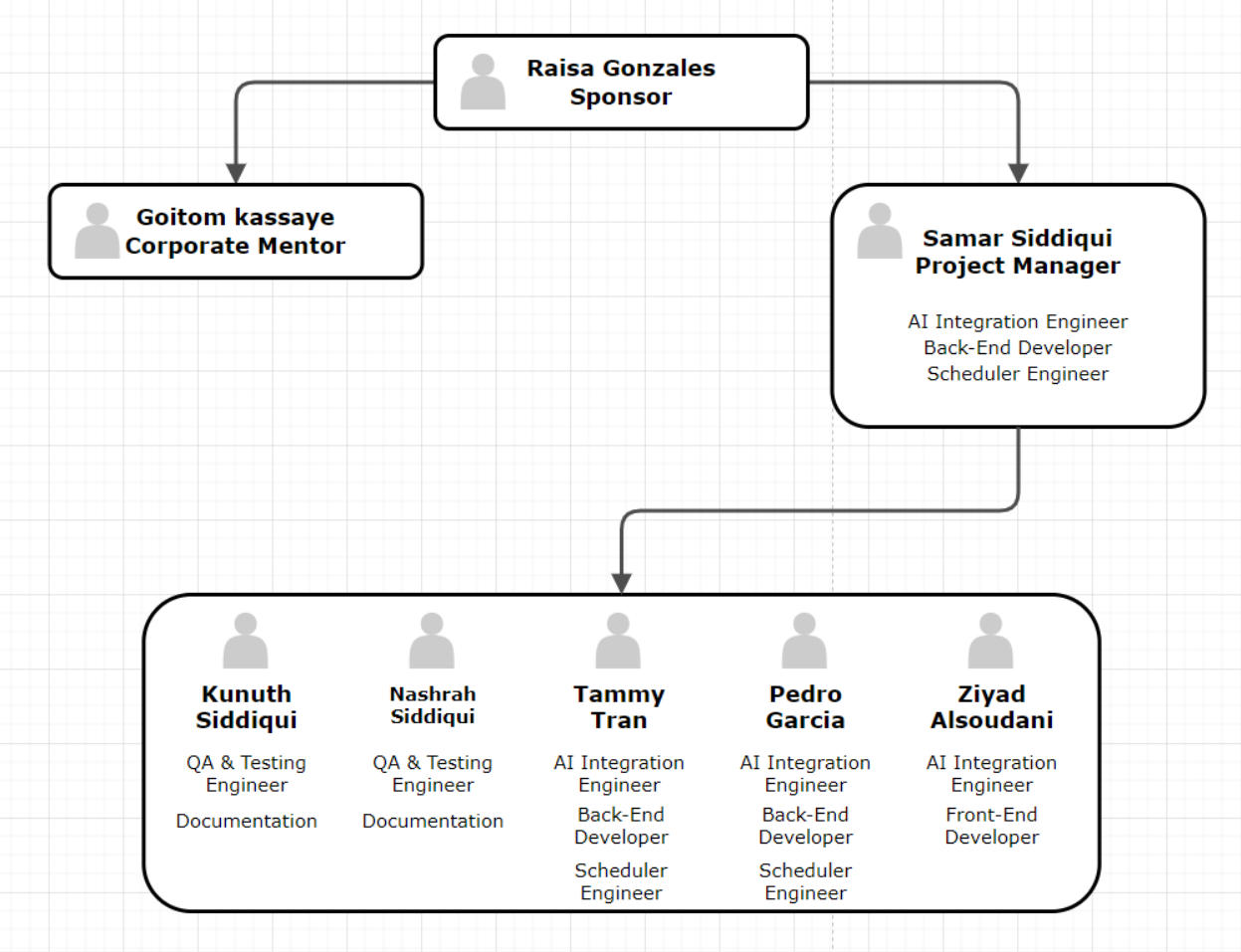

Figure 1.1 Functional Organizational Chart

LIFECYCLE MODEL USED

Our team decided on using the Incremental Model. Using this system will give us the ability to build this project in parts, whilst adding new features incrementally. Doing this will also let us deliver a working version early and add improvements step by step.

- Manual AI note generation.
- Automated scheduling.
- Testing and deployment.

Rationale: The Incremental Model was selected because it supports step-by-step development, enables early feature delivery, provides opportunities for mentor feedback, and lower risk by testing each part as they are built. All requirements are clear from the start and will not change throughout development which also supports the team's decision to use the Incremental Model.

RISK ANALYSIS

- AI Output Risk: LLM may generate clinically irrelevant or incorrect notes.
  - Risk Level: Medium
  - Mitigation: Use structured prompts, JSON outputs, and validation logic.
- Integration Risk: API mismatches with [REDACTED] simulator.
  - Risk Level: Medium
  - Mitigation: Frequent testing with [REDACTED] API stubs.
- Scheduling Risk: Automated jobs may overload the system.
  - Risk Level: Low
  - Mitigation: Schedule jobs during off-peak hours (as per requirements).
- Team Risk: Lack of experience in prompt engineering.
  - Risk Level: Medium
  - Mitigation: Assign learning tasks early.

Rationale: These risks were identified because they directly affect project success. Addressing AI accuracy, system integration, scheduling, and team skills early will assist with reducing delays, help with proper assignment of roles, and ensure a working product by the end of the semester.

SOFTWARE AND HARDWARE RESOURCE REQUIREMENTS

- Software:
  - React (UI updates)
  - Backend frameworks (Node.js/Flask/Django)
  - GitHub (version control)
  - Jira (issue tracking)
  - LLM API (Gemini)
- Hardware:
  - Standard laptops with 8GB+ RAM and internet access.
  - UTD lab machines if needed for testing and deployment.

Rationale: These tools provide everything needed for coding, collaboration, testing, and running the AI note generation service. Most of the LLM processing happens on the providers servers (API requests/responses) and not on local machines. Node.js, Flask, and Django are small REST APIs and can be easily managed. The React UI changes are minor. If necessary, UTD lab resources are available.

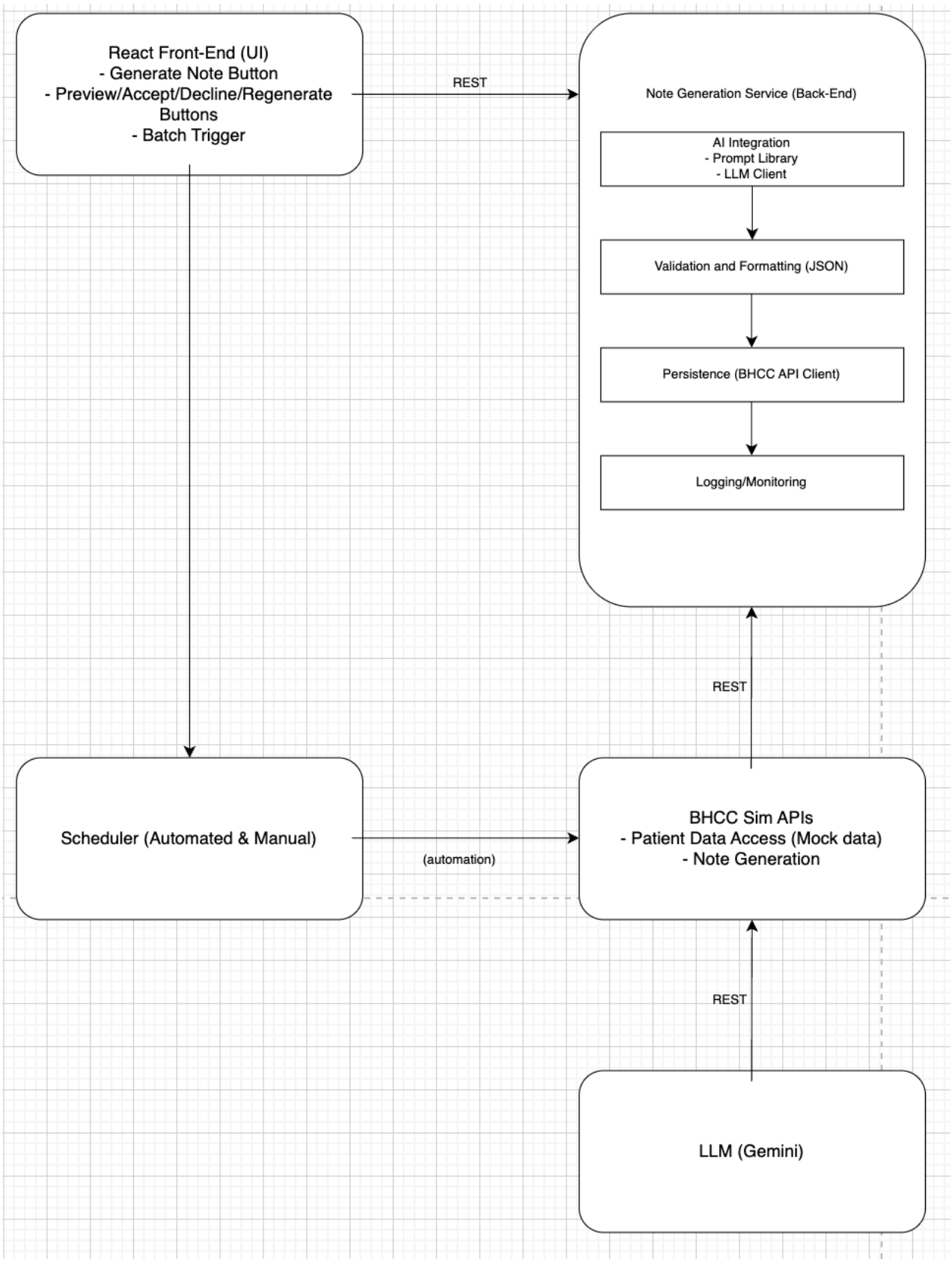

Figure 2.1 Software Architecture Design (High-Level)

DELIVERABLES AND SCHEDULE

- Activities and Deliverables:
    - Project setup and team roles
    - Front-end UI updates (React)
    - Back-end REST APIs
    - AI integration and prompt library
    - Scheduler for automation
    - Testing suite (unit, integration, functional)
    - Documentation and final demo
- Dependencies:
    - UI depends on back-end APIs.
    - Automation depends on AI integration.
    - Testing depends on completed features.

Rationale: This schedule is organized so each part builds on the previous one. This ensures steady progress and a working system by the final demo.

Table 2.1 Schedule

Week	Summary	Deliverable Due Dates
9/8 - 9/14	Project Planning & Expectations	9/12 Project Management Plan
9/15 - 9/21	Wireframes and Workflows UI/UX	
9/22 - 9/28	Front-end UI updates/Back-end implementation AI integrations and prompting	9/26 Requirements Documentation
9/29 - 10/5	Testing suite	
10/6 - 10/12	Documentation and final demo for part 1	10/10 In-class Midterm Meeting
10/13 - 10/19	Part 2 Planning & Expectations	
10/20 - 10/26	Wireframes/Workflows/Business Use Cases	10/24 Architecture Documentation
10/27 - 11/2	Front-end UI updates/Back-end implementation AI integrations and prompting	
11/3 - 11/9	AI refinement	11/7 Detailed Design Documentation
11/10 - 11/16	Testing suite	
11/17 - 11/23	Testing suite & documentation	11/21 Test Plan
11/24 - 11/30	Documentation and final demo part 2	
12/1 - 12/7	Closing remarks	12/2 Final Project Presentation Slides & Demo 12/5 Final Project Report

MONITORING, REPORTING, AND CONTROLLING MECHANISMS

- Weekly Status Reports: Summarize completed tasks, blockers, and next steps.
- Weekly Mentor Meetings: Progress review with [REDACTED] mentor.
- GitHub Activity Tracker: Monitor commits, branches, pull requests.
- Issue Tracking: Jira

Rationale: Frequent communication and transparency will help in reducing risk for misalignment.

PROFESSIONAL STANDARDS

Team members are expected to act honestly, meet deadlines, attend meetings, and produce quality work. Clear and respectful communication, along with personal accountability, are essential. Scholastic dishonesty, missed work without reason, or poor behavior will not be accepted. Refer to Appendix A for more details.

Rationale: Clear standards ensure fairness, teamwork, and reliable progress. They help maintain professionalism and align with course and sponsor expectations.

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 3.1 Configuration Management Tool

Version	Date	Change Type	Description of Changes	Difference Highlights	Reviewers
v1.0.0	2025-09-22	Added	Project Plan: Initial project plan and repository structure.	Initial commit	@ksiddii (Kunuth S.) — Ship-It 👍 @nashrah-s (Nashrah S.) — Ship-It 👍
v1.0.1	2025-09-22	Fixed	Project Plan: Added more evidence for the CM tool. Removed GitHub link under *Evidence CM* section. Removed Google Drive mentioned under *Evidence CM* section.	+1 image	@ksiddii (Kunuth S.) — Ship-It 👍 @nashrah-s (Nashrah S.) — Ship-It 👍

			Replaced previous screenshot with a new screenshot of GitHub CM tool.		
v1.0.2	2025-09-22	Fixed	Project Plan: Added *ENGINEERING STANDARDS AND MULTIPLE CONSTRAINTS* to Table of Contents. Added *ADDITIONAL REFERENCES* to Table of Contents.	+2 additions to Table of Contents.	@ksiddii (Kunuth S.) — Ship-It 👍 @nashrah-s (Nashrah S.) — Ship-It 👍
v1.1.2	2025-09-23	Fixed	Project Plan: Removed screenshot of GitHub CM tool. Created a table representing GitHub CM tool. Edited Table of Contents to correspond to pages. Added revised document to the documents folder in Github repository. Converted document to .docx instead of .pdf	-1 image +1 table +1 item to List of Tables +1 item in the documents folder.	@ksiddii (Kunuth S.) — Ship-It 👍 @nashrah-s (Nashrah S.) — Ship-It 👍

ENGINEERING STANDARDS AND MULTIPLE CONSTRAINTS

- IEEE Std 1058-1998: Software Project Management Plans [pdf]
- PMBOK® Guide: Project Management Body of Knowledge [pdf]
- IEEE Std 12207: Software Life Cycle Processes [pdf]
- IEEE Std 15939: Measurement Process [pdf]
- ISO/IEC/IEEE Std 29148-2018: Systems and Software Engineering
  - Life Cycle Processes
  - Requirements Engineering [pdf]

ADDITIONAL REFERENCES

- Larson, E. and Gray, C., 2014. Project Management: The Managerial Process. McGraw Hill
- Humphrey, W.S. and Thomas, W.R., 2010. Reflections on Management: How to Manage Your Software Projects, Your Teams, Your Boss, and Yourself. Pearson Education

APPENDIX A

The following provides a professional standards guideline for the teams. This guideline may be tailored.

Guideline:

On the first occurrence of unacceptable behavior, determine the circumstances involved, resolve the problem, and document the event in the meeting minutes.

On a second occurrence, notify the instructor of the problem. A meeting will be set up to evaluate the situation and resolve the problem.

On a third occurrence, again notify the instructor of the problem. A meeting will be set up to evaluate the situation and resolve the problem. At this point, the team will have the *option* of removing the team member. If removed, then the team member receives a pro-rated grade based on the number of weeks they have participated in the group.

Examples of unacceptable behavior may include not delivering on time, delivering poor quality work, missing team meetings, being unprepared for team meetings, disrespectful or rude behavior, etc. Reasons such as “too busy” or “I forgot”, or “my dog ate my design model” are unacceptable.

Valid reasons that must be considered include those listed for obtaining an incomplete standing in a course (illness, death in the family, travel for business or academic reasons, etc.)