DUY TÂN UNIVERSITY

International School

Capstone Project 1

Project Proposal

Version 1.3

Date: August 24th, 2025

Applying Large Language Models (LLMs) for the Automatic Transformation of User Requirements into Software Design Models

Submitted by

Cuong, Doan Viet

Dat, Hoang Van Tien

Phap, Nguyen Van

Phong, Hoang Ba

Quynh, Ho Dang

Approved by

Nguyen Duc Man, Ph.D

Proposal Review Panel Representative:

_____

Name Signature Date

Capstone Project 1- Mentor:

_____

Name Signature Date

Da Nang, August …., 2025

PROJECT INFORMATION

Project acronym	SmartSpec		
Project Title	Applying Large Language Models (LLMs) for the Automatic Transformation of User Requirements into Software Design Models		
Start Date	August 18, 2025	End Date	December 14, 2025
Lead Institution	International School, Duy Tan University		
Project Mentor	Nguyen Duc Man Ph.D		
Scrum master / Project Leader & contact details	Phap, Nguyen Van Email:nguyenvanphap3504@gmail.com Tel: 0979066067		
Partner Organization	Duy Tan University		
Project Web URL			

Team members	Name	Email	Tel
28217331026	Cuong, Doan Viet	doancuong357@gmail.com	0862496551
28219005093	Dat, Hoang Van Tien	hngvtdat010@gmail.com	0961167003
28210250268	Phap, Nguyen Van	nguyenvanphap3504@gmail.com	0979066067
28217450789	Phong, Hoang Ba	hoangbaphongk20cva@gmail.com	0388969066
28211103076	Quynh, Ho Dang	hodangquynh2k4@gmail.com	0932342990

DOCUMENT APPROVALS

The following signatures are required for approval of this document.

Nguyen Duc Man Ph.D *Mentor*	Signature	Date

Phap, Nguyen Van Student ID: 28210250268 *Scrum Master*	Signature	Date
Cuong, Doan Viet Student ID: 28217331026 *Development Team*	Signature	Date
Dat, Hoang Van Tien Student ID: 28219005093 *Development Team*	Signature	Date
Phong, Hoang Ba Student ID: 28217450789 *Development Team*	Signature	Date
Quynh, Ho Dang Student ID: 28211103076 *Product Owner*	Signature	Date

REVISION HISTORY

Version	Date	Comments	Author
1.0	August 20th, 2025	Initial Release	C1SE.22
1.1	August 21th, 2025	Update Proposal	C1SE.22
1.2	August 21th, 2025	Update Proposal	C1SE.22
1.3	August 24 th, 2025	Update Proposal	C1SE.22

TABLE OF CONTENT

1. Project Title

SmartSpec - Applying Large Language Models (LLMs) for the Automatic Transformation of User Requirements into Software Design Models

2. Team Members

Table 2.1. *Team members information*

No.	Full name	Email	Role
1	Nguyen Van Phap *Student ID: 28210250268*	nguyenvanphap3504@gmail.com	Full Stack Developer, Scrum Master
2	Ho Dang Quynh *Student ID: 28211103076*	hodangquynh2k4@gmail.com	Backend Developer, Product Owner
3	Hoang Van Tien Dat *Student ID: 28219005093*	hngvtdat010@gmail.com	Backend Developer
4	Doan Viet Cuong *Student ID: 28217331026*	doancuong357@gmail.com	Frontend Developer
5	Hoang Ba Phong *Student ID: 28217450789*	hoangbaphongk20cva@gmail.com	Frontend Developer

3. Supervisor

Table 3.1. *Mentor information*

Name / Position	Nguyen Duc Man Ph.D
Position	Deputy Head of International School, Head of Software Technology Program (International School)
Phone Number	0904 235 945
E-mail	mannd@duytan.edu.vn

4. Problem Statement

During the software development lifecycle, Business Analysts (BAs) and System Analysts (SAs) often encounter numerous challenges. One of the most common issues lies in consolidating fragmented information from various sources, as well as manually creating software design models. Such practices frequently result in undesirable consequences, including excessive time consumption, human errors, lack of consistency, difficulties in documentation standardization, and ultimately, reduced efficiency and stability in software quality.

With the rapid advancement of technology, particularly the remarkable progress in Artificial Intelligence (AI), the integration of Large Language Models (LLMs) into the software development process promises to deliver substantial benefits. Specifically, LLMs can assist in synthesizing and analyzing input data, automatically generating documentation compliant with IEEE/ISO standards, and producing software design models in an automated manner. These capabilities not only help save time and effort but also significantly enhance efficiency and overall product quality throughout the entire software development lifecycle.

5. Current Status of Art

Table 5.1. *Feature Comparison*

Criteria	IBM DOORS	Jama Connect	SmartSpec
Automated Analysis of Raw Input			**X**
Automated "Single Source of Truth" (SSoT) Generation			**X**
Automated UML Diagram Generation			**X**
Automated Consistency Assurance			**X**
Formal Review & Approval Workflows	**X**	**X**	
Cost & Accessibility	**Very High**	**High**	**Free**

6. Objectives and Scope

6.1. Objectives

The primary goal of this project is to significantly enhance the efficiency, accuracy, and comprehensiveness of the software system analysis and design process through the strategic application of Large Language Models (LLMs).

Specifically, this project aims to:

- Automate and accelerate the extraction and synthesis of requirements from various input formats (docx/pdf documents, images, audios, keyboard input).
- Automatically generate activity diagrams, sequence diagrams, use case diagrams, database schemas, user stories, and test cases by leveraging LLMs' understanding and generation capabilities, thereby ensuring consistency and completeness across analysis and design artifacts.
- Foster a more streamlined and collaborative workflow by providing integrated tools that leverage LLM-generated insights, ultimately reducing manual effort and potential for human error in the initial phases of software development.

Deliver a proof-of-concept system demonstrating the tangible benefits of LLM integration in optimizing the crucial early stages of software project development.

6.2. Scope

6.2.1. In Scope

- Automatic document generation from requirements.
- Summary, compilation, standardization and document version management.
- Diagram generation (UML, ERD, Flowchart) from text descriptions.
- End-to-end linkage between URD → Use Case Specification Document → Design (UML, Database Schema) → Test Case.
- **Target users:** Software Business Analysts / System Analysts, Test Engineers / QA, Software Developers. Clients / End Users.

6.2.2. Out of Scope

- Completely replacing human roles in system analysis and design.
- Supporting domains outside software development.
- Integration with development tools (Jira, GitHub, CI/CD).

7. Key Features & Requirements

7.1. Key Features

Table 7.1. *Key Feature Description*

No.	Name	Description
1	User Authentication & Account Management	- Users can log in, log in with the account system or via a third-party service (Google OAuth2). - Support account management, password reset, layered authentication (2FA).
2	Project & Team Management	- Users create, edit, delete projects. - Allows inviting members to join the project, assigning permissions (Viewer, Editor).
3	Multi-format Input & Document Processing	- Allows importing requests from multiple formats: Word, PDF, images, audios, plain text and manual input. - The system automatically standardizes input data, eliminates duplicates and unnecessary formats.
4	Document Version Control	- Manage document versions (version control) to track editing history.
5	Automatic Software Design Generation	- AI suggests additional missing requirements based on context. - Automatically generate visual models (UML, Database Schema) and matching Test Cases from analyzed requirements to ensure consistency between design and validation.
6	Multi-format Export	- Allow exporting documents to multiple formats (PDF, Word, PNG).
7	API Key Management	- Admins can provide access to different LLM models through API keys, enabling users to integrate and utilize them in their projects.

7.2. Quality Attributes

Table 7.2. *Quality Atributes*

No.	Description
1	Performance: The system processes input requests (≤50MB) in ≤5 seconds for normal tasks.
2	Security: All sensitive data is encrypted, supports 2FA, secure login session management.
3	Usability: Intuitive, simple interface, suitable for non-IT users.
4	Maintainability: Modular system architecture, easy to maintain and expand.

8. Constraints and Assumptions

Table 8.1. *Constraints*

Technical Constraints	- Backend : Node.js (Express.js), Python - Frontend: Vue.js (HTML, CSS, JavaScript) - Database: MongoDB. - Version Control System: GitHub - Team Management: Trello, Zalo, Google drive, Google doc - Develop tools: Visual Studio Code, Postman
Business Constraints	- Resource: 5 people. - Budget: Not applicable - Time: 4 months (August 18 - December 14)

9. Target Users

- **Software Business Analysts / System Analysts**
  - **Usage**: Provide initial requirements in natural language.
  - **Benefits:** Automatically generate UML diagrams (Use Case, Activity, Sequence), and User Stories for clearer communication with stakeholders.
- **Software Developer**
  - **Usage**: Utilize the generated UML diagrams, database schema, and user stories as references for implementation.
  - **Benefits:** Reduce misunderstandings in requirement interpretation and accelerate the coding phase.

- **Test Engineers / QA (Quality Assurance)**
  - **Usage**: Obtain automatically generated test cases derived from requirements and user stories.
  - **Benefits:** Improve test coverage, reduce manual effort in test design.
- **Clients / End Users (Indirect Stakeholders)**
  - **Usage**: Review simplified system models (use case diagrams, user stories) to validate that the system aligns with business needs.
  - **Benefits**: Facilitate communication between technical and non-technical stakeholders.

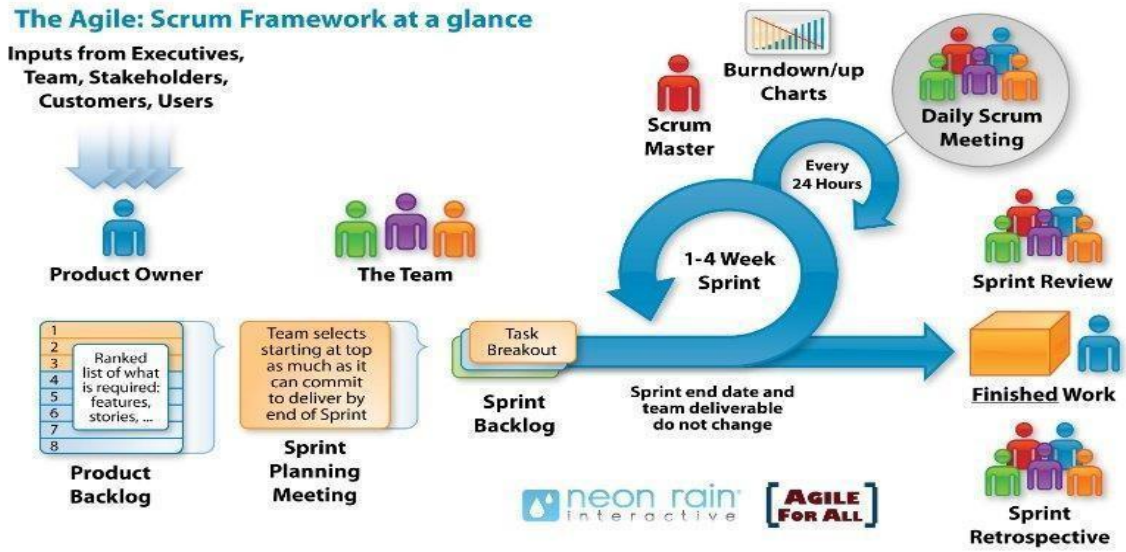
10. Technology Stack

- **Frontend**
  - Vuejs (HTML, CSS, JavaScript): Develops modern, responsive, and scalable web interfaces.
  - TailwindCSS / Material UI: Provides prebuilt components and utility classes for fast, consistent, and user-friendly UI design.
- **Backend**
  - Node.js (Express.js): Handles business logic, user management, and API services while ensuring scalability.
  - Python : Dedicated to processing modules (LLM, NLP, OCR, Speech-to-Text).
- **Database**
  - MongoDB: Stores user data, and system logs in a flexible NoSQL structure.
  - Redis: Provides caching and session management to improve performance and responsiveness.
- **AI**
  - LLMs (OpenAI, Gemini): Extract requirements from natural language, generate and perform advanced text analysis.

- **OCR and STT technologies**
  - OCR (Tesseract): Extracts text from scanned images or documents.
  - Speech-to-Text (Whisper): Converts audio recordings into text, enabling voice-based input.
- **Supporting Tools**
  - Docker: Containerization and orchestration for portable, reliable deployment across environments.
  - GitHub: Source code management with CI/CD pipelines for continuous integration and delivery.
  - Trello: Project management tools for task assignment and progress monitoring.
  - Postman: API testing and documentation.

11. Methodology & Development Plan

11.1. Scrum Process



- Scrum is an iterative and incremental agile software development framework for managing software projects and product or application development.
- Scrum focuses on project management institutions where it is difficult to plan ahead.
- Empirical process control uses feedback loops as the core management technique, instead of traditional command-and-control.

- Its approach to planning and managing projects is by bringing decision-making authority to the level of operation properties and certainties.
- Benefit of the methodology:
  - Project can respond easily to change.
  - Problems are identified early.
  - Customers get the most beneficial work first.
  - Work done will better meet the customer's needs.
  - Improved productivity.
  - Ability to maintain a predictable schedule for delivery.

11.2. Development Plan

Table 11.1. *Development Plan*

No.	Task Name	Duration	Start	Finish
1.	**Initial**	**10 days**	**18-Aug-2025**	**27-Aug-2025**
1.1	Gathering Requirement	4 days	18-Aug-2025	21-Aug-2025
1.2	Create Proposal Document	6 days	22-Aug-2025	27-Aug-2025
2	**Start Up**	**10 days**	**28-Aug-2025**	**06-Sep-2025**
2.1	Project Kick-off Meeting	2 days	28-Aug-2025	29-Aug-2025
2.2	Create Document	8 days	30-Aug-2025	06-Sep-2025
3	**Development**	**84 days**	**07-Sep-2025**	**29-Nov-2025**
3.1	Sprint 1	21 days	07-Sep-2025	27-Sep-2025
3.2	Sprint 2	21 days	28-Sep-2025	18-Oct-2025
3.3	Sprint 3	21 days	19-Oct-2025	08-Nov-2025
3.4	Sprint 4	21 days	09-Nov-2025	29-Nov-2025
4	**Project's Retrospective Meeting**	**7 days**	**30-Nov-2025**	**06-Dec-2025**
5	**Final Release**	**8 days**	**07-Dec-2025**	**14-Dec-2025**

12. System Architecture Overview

12.1. System context diagram

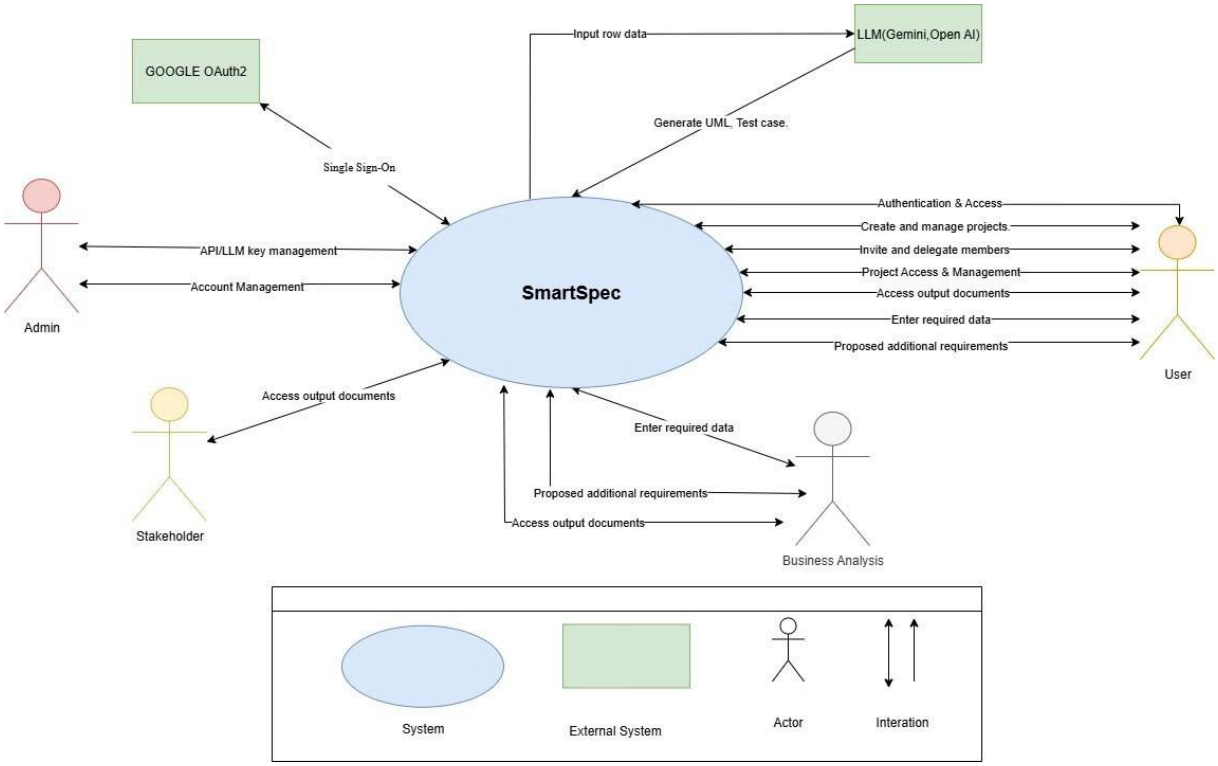


Figure 12.1. *System context diagram*

12.2. System context description

- **Admin:**
  - **Account Management**: Create, update, and delete user accounts in the system.
  - **API/LLM Key Management**: Manage API/LLM integration keys (Gemini, OpenAI…), including add, remove, and check status of keys.
- **User (End User – PM, Developer, Tester, Stakeholder, etc.):**
  - **Authentication & Access**: Log in and access the system.
  - **Create and Manage Projects**: Create new projects and manage existing ones.
  - **Invite and Delegate Members**: Invite other members to join a project and assign roles (Viewer, Editor, Owner).

- **Project Access & Management**: Manage permissions and access control within projects.
- **Enter Required Data**: Upload documents (docx, pdf, image OCR, audio, manual text input) as input for analysis.
- **Proposed Additional Requirements**: Review and approve suggested requirements proposed by the LLM.
- **Access Output Documents**: Download generated artefacts such as UML Diagrams, Test Cases, Database Schemas.

- **Business Analyst (BA):**
  - **Enter Required Data**: Provide or upload requirement documents for analysis.
  - **Proposed Additional Requirements**: Review, refine, and select LLM-generated requirement suggestions..
  - **Access Output Documents**: Download, review, and edit deliverables such as Use Cases, UML Diagrams).
- **Stakeholder (Developers / QA / Testers):**
  - **Access Output Documents**: Retrieve generated artefacts (UML, Test Cases, DB Schema).
- **Google (OAuth2):**
  - **Single Sign-On (SSO):** Enables users to log in quickly using their Google accounts.
- **LLM (Gemini, OpenAI):**
  - **Input Raw Data:** Receives requirement documents (text, docx, pdf, image OCR, audio transcription).
  - **Analyze Requirements:** Processes documents and extracts requirements).
  - **Generate Artefacts:** Produces UML diagrams (Use Case, Activity, Sequence), Test Cases.

13. Potential Risks and Mitigation Strategies

Risk	Description	Mitigation Strategy
Output results lack accuracy	Caused by insufficient or poor-quality input data (e.g., low-resolution images, noisy audio, incomplete text).	- Apply preprocessing steps such as text cleaning, noise reduction in audio, and image quality enhancement before inputting data into the system. - Establish a mechanism for checking and alerting when the input data does not meet standards (e.g., alerting users when images are blurred or audio files are distorted). - Allowing users to manually input supplementary data when the system's recognition is inaccurate.
LLM Generates Irrelevant or Fabricated Information	LLM may generate information that does not exist or is inaccurate, leading to discrepancies in design documentation.	- Combine cross-checking with rule-based validation or domain-specific ontology. - Add a human-in-the-loop step for review before releasing documents. - Limit the scope of output by using precise prompts and employing RAG (Retrieval Augmented Generation) with standardized data.
Dependency on third-party APIs and services	The system relies on APIs from OpenAI, Google Gemini, or OCR/STT services, which carry risks of downtime, pricing changes, or quota limits.	- Design a fallback mechanism with multiple alternative providers - Implement temporary caching to avoid disruptions. - Monitor SLAs and set up alerts when APIs exceed thresholds..
Security and Privacy Risks	Sensitive project data (e.g., user information, training datasets) may be exposed to unauthorized access or leaks.	- Apply encryption for data at rest and in transit - Implement strict authentication and role-based access control. - Anonymize or mask sensitive datasets when possible. - Conduct regular security audits and penetration testing..

Cultural and Job Adaptation Issues	After deployment, the system may impact existing workflows, organizational culture, or even reduce the need for certain job roles, leading to resistance or dissatisfaction among staff.	- Involve stakeholders early to communicate the benefits of the systems. - Provide training and role adaptation programs to help employees adjust to new workflows - Ensure transparency about how the system affects job responsibilities, and emphasize tasks where human expertise remains essential. - Gradually integrate the system to allow time for adaptation rather than sudden replacement
System Performance Risk	When applying preprocessing steps (e.g., noise reduction, image enhancement), the processing time may increase significantly.	- Optimize algorithms and balance between quality and speed. - Continuously monitor system performance and adjust configurations dynamically (adaptive tuning).

14. Deliverables

- **Application (Web Platform)** – A fully functional web-based system that supports requirement extraction, modeling, and documentation generation.
- **Source Code Repository** – A public/private repository (e.g., GitHub/GitLab) containing:
  - **Frontend** (Vue.js + TailwindCSS/Material UI)
  - **Backend** (Node.js + Express, Python services for AI/ML)
  - **Processed Modules** (LLM integration, OCR, Speech-to-Text)
  - **Database Scripts** (MongoDB, Redis setup)
- **Technical Documentation & Demo Video** – Includes installation guide, system architecture, API references, and a recorded demo video highlighting main functionalities..
- **Final Report** – A comprehensive project report covering problem statement, methodology, implementation, evaluation, and proposed future improvements.

15. References

[1] IEEE. *Software Development Standards for the Guidance and Control Software Project*. IEEE Standards Association.

[2] Sommerville, Ian. *Software Engineering* (10th Edition). Pearson, 2015.

[3] ISO/IEC/IEEE 29148:2018. *Systems and Software Engineering — Life Cycle Processes — Requirements Engineering*.

[4] OpenAI. *GPT-4 Technical Report*. https://openai.com/research/gpt-4

[5] MongoDB Inc. *MongoDB Documentation*. https://www.mongodb.com/docs/

[6] Vue.js Core Team. *Vue.js Official Documentation*. https://vuejs.org/

[7] Express.js Team. *Express.js Documentation*. https://expressjs.com/

[8] Python Software Foundation. *Python 3* https://docs.python.org/3/

[9] Node.js Foundation. *Node.js Documentation*. https://nodejs.org/en/docs/

[10] W3Schools. *W3Schools Online Web Tutorials.* https://www.w3schools.com/

[11] Google DeepMind. *Gemini Technical Report.* https://deepmind.google/gemini/

[12] Smith, Ray & Google Tesseract OCR Team. *Tesseract OCR Documentation.* https://tesseract-ocr.github.io/

[13] Radford, Alec et al. *OpenAI Whisper: Robust Speech Recognition via Large-Scale Weak Supervision.* https://github.com/openai/whisper

[14] Object Management Group (OMG). *Unified Modeling Language (UML) Specification.* https://www.omg.org/spec/UML/

[15] Duy Tan University. (n.d.). *C2SE12-Proposal_LET_template.*

16. Description of product requirements

Group: C1SE.22

Project: Applying Large Language Models (LLMs) for Requirements Analysis and Automated Generation of Software Design Documentation

Date: August 24th, 2025

16.1 Short description of product ideas

SmartSpec is a web platform that utilizes Large Language Models (LLMs) to automate the conversion of user requirements into software design models. Its goal is to enhance efficiency and accuracy throughout the software development lifecycle by automatically generating design documents and documentation. The system can handle various input formats, including text, documents, images, and audio. It produces standardized outputs such as use case specification documents, UML diagrams, database schemas, and test cases. The project focuses on providing a streamlined workflow and collaboration for different stakeholders involved in the software development process.

16.2 Requirements

High-level Functional Requirements	1. User Authentication & Account Management
	2. Project & Team Management
	3. Multi-format Input & Document Processing
	4. Document Version Control
	5. Automatic Software Design Generation
	6. Multi-format Export
	7. API Key Management

Quality Attributes Requirements	1. Learnability: Complex features should include tooltips with explanations.
	2. Usability: The main workflow, from file upload to receiving the diagram, should not exceed 5 primary user interactions.
	3. Consistency: The entire application must follow a single Design System to ensure consistency in colors, icons, and layout.
	4. System Feedback: Error messages must be clear, explaining the cause and suggesting possible solutions.

Operation Requirements	1. Speed: The system needs to process and generate outputs quickly to meet performance requirements.
	2. Accuracy: The generated outputs (diagrams, test cases) must be accurate, despite the risk that LLMs may produce irrelevant or misleading information.
	3. Stability: The system must remain stable, even when relying on third-party APIs that may risk downtime.
	4. Load Capacity: The system must be capable of handling multiple user requests and data inputs without significantly degrading performance.
	5. Scalability: The backend system (Node.js/Express.js, Python) and the database (MongoDB, Redis) must be designed to handle increasing workloads and user data.

Environment & Operation Requirements	1. Integration with External Systems: The system must interact with external LLMs such as Google Gemini and OpenAI, as well as Google OAuth2 for single sign-on.
	2. Packaging: The system will be packaged using Docker for easy and reliable deployment across different environments.
	3. Version Control System: The project's source code will be managed on GitHub.
	4. Project Management Tools: The team will use Trello, Zalo, Google Drive, and Google Docs for collaboration and task tracking.
	5. Development Tools: Development will be carried out using Visual Studio Code and Postman.

Requirements for Maintenance & Support	**1.** Modular Architecture: The system architecture should be modular for easy maintenance and scalability.
	2. Documentation: Technical documentation, including installation guides, system architecture, and API reference documentation, will be provided to support maintenance.
	3. Source Code Repository: The source code will be available in a repository (GitHub), allowing for easy updates and bug fixes.
	4. Automated Processes: The use of CI/CD pipelines through GitHub will streamline continuous integration and delivery.
	5. Bug Fixing & Updates: The team should be prepared to address issues such as inaccurate LLM outputs and performance bottlenecks

Evaluate the complexity of engineering problems	1. Involving wide-ranging or conflicting technical issues
	2. Having no obvious solution
	3. Addressing problems not encompassed by current standards and codes
	4. Involving diverse groups of stakeholders
	5. Including many component parts or sub-problems
	6. Involving multiple disciplines
	7. Having significant consequences in a range of contexts

Standard requirements	1. Code standard (AirBnB JavaScript Style Guide & Google JavaScript Style Guide)
	2. Design standard. (design patterns, object-oriented analysis and design,…).
	3. IEEE (1058, 1540, 830, 1016, 829, 1012, 1008)
	4. ISO/IEC/IEEE 12207:2017 (TCVN 10539:2014); ISO/IEC 25051:2006(TCVN 10540:2014);