Introduction to Software Engineering

Project Proposal

Project Proposal documentation for the assigned course project, following the attached template.

Software Engineering Department
Faculty of Information and Technology
University of Science

Table of Contents

Project Proposal

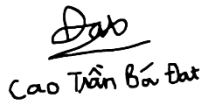
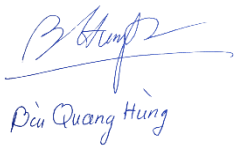
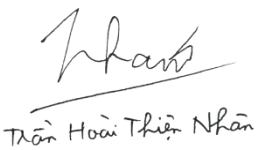
Objectives

This document focus on the following topics:

- ✓ Completing the Project Proposal document with the following sections:
  - Preliminary Problem Statement
  - Proposed Solution
  - Development Plan
  - Human Resources & Costing Plan
- ✓ Understanding the Project Proposal document.

1 Member Contribution Assessment

GROUP 10:

ID	Name	Contribution (%)	Signature
23127097	Trần Trí Nhân	100%	
23127168	Cao Trần Bá Đạt	100%	
23127193	Bùi Quang Hùng	100%	
23127234	Nguyễn An Nghiệp	100%	
23127238	Trần Hoài Thiện Nhân	100%	

2 Preliminary Problem Statement

*The project aims to develop an **ChatBot_HelpDesk**— a web-based system that allows individuals and businesses to create, manage, and deploy AI-powered chatbot agents tailored for customer interaction, support, and internal service assistance. The platform provides a unified interface for users to design custom conversational agents, define their knowledge base, and integrate them seamlessly into websites or internal systems.*

*The system is designed to assist two primary user categories — **individual users** and **business organizations** — with flexible pricing models and scalability in mind.*

- ***Individual users** can subscribe to the platform with a monthly cost of 200,000 VND.*
- ***Business organizations** can access advanced collaboration and management features with a monthly subscription of 500,000 VND. Additionally, the system applies a pay-per-use model, charging **100 VND per AI-generated ticket response** to maintain fair usage and cost efficiency.*

System Overview:

*The **Agent Web Platform** provides a variety of features divided into several major functional modules:*

1. Agent Creation and Configuration (Core Flow)

Users can create AI agents with different purposes — such as answering customer inquiries about products, handling complaints, or assisting staff in serving clients more effectively.

Each agent can be customized with:

- *A knowledge base, where users input domain-specific data stored in the database.*
- *A response configuration module, allowing users to define instructions that shape how the AI answers questions.*
- *A testing feature, enabling users to preview and refine their agents' responses before deployment.*

2. Agent Sharing and Collaboration

The system supports two types of sharing mechanisms:

- *Personal sharing, where an agent is shared directly with another user via email.*
- *Group sharing, where the creator can form a team or department group, allowing multiple employees to use the same agent collaboratively.*
- *This feature enhances cooperation within companies and enables collective access to the same AI resources.*

3. Helpdesk and Customer Support Integration

- *Agents can be used as helpdesk assistants, managing customer interactions efficiently.*

  Key capabilities include:
- *Automatically answering customer inquiries and generating ticket responses for unresolved cases.*
- *Managing customer information and employee data within the same system.*
- *Providing statistical reports to track performance, ticket resolution time, and interaction metrics.*

4. Web Embedding and Automation

Users can generate embed scripts to integrate their AI agents directly into their company's websites or portals. If an agent cannot answer a question, the query is automatically logged as a ticket response, allowing administrators to review and improve the knowledge base later.

Operating Environment

- ***Frontend & Backend Framework**: Next.js*
- ***Deployment**: Vercel*
- ***Database**: Firebase*

Design and Implementation Constraints

- ***Programming Language:** TypeScript (Next.js framework)*
- ***Database:** Firebase*
- ***Development Methodology:** Scrum Agile*

Conclusion

In summary, this project aims to provide an intelligent, scalable, and user-friendly web platform that empowers both individuals and businesses to create, deploy, and manage AI chatbot agents efficiently.

The system will streamline customer service workflows, improve response time, and reduce the workload of support staff by automating common interactions — while maintaining transparency, collaboration, and control through integrated management tools.

3 Proposed Solution

3.1 Software

3.1.1. Features

Need	Requirement
As a staff, I want to ask about the company's regulation	*Has bot chat with company's regulation knowledge*
As a customer, I want to ask about a product's details like stock level, price,...	*Has bot chat with knowledge about the asking products*
As a customer, I want to have support for complaints	*Has ticket system*
As a customer, I want to communicate with a human staff when needed	*Be able to contact to human staff*
As a manager, I want to view monthly tickets and activities	*Reports/statistics visualization + export*
As a manager, I want to the bot to know about company's regulation as well as the details about our products	*Add company's regulation and product details to the bot's knowledge base*
As a manager, I want to know my bot's knowledge base	*View chatbot's knowledge base*
As a manager, I want to change the AI model of my bot	*Has a list AI models*
As a manager, I want to share the bot to my employees	*Has an invite system*
As a manager, I want to integrate the chat bot to other applications	*Create a chat widget integration*
As an admin, I want to change users' passwords and their roles	*Manage access and permissions*

3.1.2. Software Architecture

1. ***Components***

The system is divided into the following logical component blocks:

1.1. ***Presentation Layer***

- *User Interface (UI): Utilizes a UI library/framework to display web pages based on the design, manage UI states, and handle user interactions.*

1.2. ***Application & Business Logic Layer***

- *Application Server: Utilizes an application server framework to handle requests and execute business logic. The entire server is deployed on a serverless deployment platform and processes requests via the platform's serverless functions.*

1.3 ***AI Service Layer***

- *AI Service (LLM Service): A third-party Large Language Model (LLM) service to provide natural language generation and embedding creation (data vectorization).*
- *Data Ingestion Pipeline: A process (typically a Serverless Function) to automatically process internal documents when they are uploaded.*

1.4 ***Data & Storage Layer***

- *Database: A NoSQL database is used to store application data (e.g., chat history, user information).*
- *File Storage: A cloud file storage service to store original internal documents (PDF, DOCX...) uploaded by administrators.*
- *Vector Database: A vector database solution to store document "embeddings" for Retrieval-Augmented Generation (RAG).*

1.5 ***Support & Security Layer***

- *Authentication Service: An authentication service is used to manage access and user authentication.*

2. ***Relationships & Data Flows***

The components connect and exchange information through the following flows:

2.1. ***Basic Communication Flow***

- *Client ⇔ Server: The website (Client) communicates with the application server via restful API calls. After the server loads data and handles logic, it returns the rendered web pages to the client.*
- *Server ⇔ Database/Services: The application server uses the cloud provider's SDKs to connect directly to the NoSQL database and authentication service.*
- *Security Constraint: The client never connects directly to the cloud provider's services.*

2.2. ***Data Ingestion Flow (Admin)***

- *An administrator uploads a document via the UI.*
- *The UI sends this file to File Storage.*
- *The file upload triggers the Data Ingestion Pipeline.*
- *This pipeline: a. Retrieves the file from File Storage. b. Splits the file into small chunks. c. Calls the AI Service to create "embeddings" (vectors) for each chunk. d. Stores these vectors in the Vector Database.*

2.3. ***Chatbot Query Flow (User)***

- *A user submits a query via the UI.*
- *The UI sends the query to the Application Server.*
- *The server calls the AI Service to create an embedding for the user's query.*
- *The server uses this embedding to query the Vector Database and find the most relevant internal document chunks.*
- *The server constructs a complete prompt (including the original query + relevant document chunks).*
- *The server sends this complete prompt to the AI Service.*
- *The AI Service returns the final answer.*
- *The server returns this answer to the UI for display.*

3. ***Principles & Constraints***

- *Serverless Architecture (FaaS): The entire system requires no physical server management and is deployed on a serverless platform and cloud service provider.*
- *Security by Layer: All sensitive data and business logic are handled by Server Components. The client (Client Component) is never exposed to API keys or database credentials.*
- *Technology Stack: The system is JavaScript & Typescript based for both backend and frontend logic.*
- *Data Privacy Constraints: As this is internal data, strict rules on data residency and ensuring third-party AI APIs do not use this data for their own training are required.*
- *Performance Constraints: Serverless functions may experience "Cold Starts," which can affect the chatbot's initial response time.*
- *Cost Constraints: The "Pay-as-you-go" model of the serverless platform and AI APIs requires close cost monitoring to stay within budget.*

3.2 Hardware

3.2.1. Service-Based Infrastructure Analysis (Service-Based Infrastructure)

Instead of managing "hardware," the system manages and consumes "services." The infrastructure automatically scales based on usage.

- *Because the application is deployed on a Serverless architecture, the server hardware is entirely managed by the serverless platform and cloud provider's infrastructure. The development team does not need to manage this hardware.*

3.2.2. Client-Side Hardware

This is the physical "hardware" that end-users own and operate, serving as the execution environment for the application.

- *Terminal Devices: Personal computers (desktop, laptop), smartphones, and tablets*
- *Processing: Any modern CPU and RAM are supported*
- *Network: Devices must have an NIC (Network Interface Card) to connect to the Internet.*

4 Development Plan

4.1 Requirements Analysis

Task	Deliverables	Begin Date	End Date
1. Requirements Analysis	Requirement analysis documentation files	8/11/2025	19/11/2025
1.1 Problem statement	A documentation file of approximately 1-2 page(s) describing the software problem, including the operating environment, design, and implementation constraints	8/11/2025	12/11/2025
1.2 Requirements Overview	Requirement overview documentation files	8/11/2025	14/11/2025
1.2.1 Identify Stakeholders	List of stakeholders and descriptions of their roles	8/11/2025	14/11/2025
1.2.2 Functional Requirements Specification	List of functional requirements	8/11/2025	14/11/2025
1.2.3 Non-Functional Requirements Specification	List of non-functional requirements	8/11/2025	14/11/2025
1.3 Requirement analysis	Requirement analysis documentation files	14/11/2025	17/11/2025
1.3.1 Use case models	A pdf/image file showcasing the use case diagrams of the system	14/11/2025	17/11/2025
1.3.2 Use case specification	Tables of detailed descriptions for each existing use case	14/11/2025	17/11/2025
1.4 Prototype/Mockup	A pdf/image file presenting the wireframe of the program	17/11/2025	19/11/2025

4.2 Software Design

Task	Deliverables	Begin Date	End Date
2. Software Design	Software design documentation files	6/11/2025	30/11/2025
2.1 Conceptual Model	A pdf/image file illustrating the semantic entities within the software	20/11/2025	21/11/2025
2.2 Architectural design	Architectural design documentation files	22/11/2025	26/11/2025
2.2.1 Architectural diagram	A pdf/image file presents the architecture as a tree diagram, highlighting special features	22/11/2025	26/11/2025
2.2.2 Class diagram	A pdf/image file presents the class diagram showcasing the relationship between the classes	22/11/2025	26/11/2025
2.2.3 Class specifications	Tables of detailed descriptions for each class	22/11/2025	26/11/2025
2.3. Data design	Data design documentation files	27/11/2025	30/11/2025
2.3.1 Data diagram	A pdf/image file presents the data diagram	27/11/2025	30/11/2025
2.3.2 Data specifications	Description for each collection, and it documents	27/11/2025	30/11/2025
2.4. UI/UX design	UI/UX design documentation files	6/11/2025	12/11/2025
2.4.1 Screen diagram	An pdf/image file of a screen diagram illustrating the relationships and transitions between the screens	6/11/2025	12/11/2025
2.4.2 Screen specifications	Description format and event handling for important scenes	6/11/2025	12/11/2025

4.3 Implementation

Task	Deliverables	Begin Date	End Date
3. Implementation	Application's source code, unit test files and git branches	5/11/2025	6/12/2025
3.1 Setup	Git branches for each features	5/11/2025	6/11/2025
3.2 Coding	Working source code for listing features	6/11/2025	6/12/2025
3.3 Unit Testing	Test files and test results for working source code	6/11/2025	6/12/2025
3.4 Integration	Merged code in main/dev branch	6/11/2025	6/12/2025

4.4 Testing

Task	Deliverables	Begin Date	End Date
4. Testing	Test documentation files	7/12/2025	11/12/2025
4.1 Test Plan	A detailed documentation for the test plan	10/12/2025	9/12/2025
4.2 Test Case	Test case documentation	10/12/2025	11/12/2025
4.2.1 Test Case List	List of test cases	10/12/2025	11/12/2025
4.2.2 Test Case specifications	Detailed descriptions of the most important test cases	10/12/2025	11/12/2025

4.5 Deployment and Maintainance

Task	Deliverables	Begin Date	End Date
5. Deployment and Maintainance	Build scripts and hosted website	12/12/2025	18/12/2025
5.1 Build application	Build scripts and configuration files	12/12/2025	17/12/2025
5.2 Deploy	The URL for hosted app	12/12/2025	18/12/2025

5 Human Resources & Costing Plan

5.1 Project Overview

- *Project Duration: October – December 2025 (approx. 1 month)*
- *Total Estimated Workload: ~172 hours*
- *Additional Costs: None (all tools used are free)*

5.2 Human Resource Plan

Team member	*Main Responsibilities*	Estimated Hours
Trần Trí Nhân	Software features, Testing theory, Automated test reports, Presentation intro	~34h
Cao Trần Bá Đạt	Software architecture, UI/UX design, Automated tests, Doctor module	~34h
Bùi Quang Hùng	Human resource & cost plan, Use case design, Testing, Security, Nurse module	~35h
Nguyễn An Nghiệp	Business description, Operating environment, UI wireframes, Patient module	~34h
Trần Hoài Thiện Nhân	Requirements analysis, Testing specs, Demo video, Hospital Director module	~35h

5.3 Cost Plan

Cost Item	Description	Estimated Cost (VND)
Software Tools	Visual Studio Code, Figma, Android Studio	0
Hardware	Personal laptops (existing)	0
Communication	Google Meet, Discord (free), Slack	0
Version Control	Github	0
Task Tracking	Jira	0
Miscellaneous	None	0

Total Project Cost: 0 VND

5.4 Project Timeline

Phase	Deliverables	Duration
PA1 – Project Proposal	Proposal document, HR & Cost Plan	31/10 – 4/11/2025
PA2 – Requirement Analysis	Problem Statement, Use Cases, Mockups	8/11 – 19/11/2025
PA3 – Software Design	Architecture Design, UI/UX Design	20/11 – 30/11/2025
PA4 – Software Testing	Test Cases, Test Reports	4/12 – 11/12/2025
PA5 – Automated Testing & Presentation	Automated Tests, Demo Video, Presentation	12/12 – 27/12/2025
Implementation (Parallel)	Full System Functions	6/11 – 6/12/2025

6 Tools setup

- *Moodle: Used for posting and submitting assignments.*
- *Slack (or a similar tool like MS Teams or Discord or Telegram): Used for discussions and interactions among group members for each project.*
    - *TAs will create a Slack project and add all students to it.*
    - *There will be a general channel for everyone, used for general project discussions.*
    - *Each group will have a channel linked to Trello. Any updates from Trello will be posted in this channel. Group members must use this channel to discuss their project.*
    - *Slack has desktop and mobile applications; please open the app and check it frequently.*
- *Discord Chatbot_HelpDesk: Used for discussions and interactions among group members for this project.*
    - *Project Work Category:*
        - *tasks: Ongoing task updates, to-do lists, and task coordination.*
        - *ideas: Brainstorming new concepts, suggestions, or improvements.*
        - *resources: Shared documents, links, and reference materials.*
        - *meetings: Notes and summaries from team meetings.*
    - *Development Category:*
        - *development: Code-related discussions, updates, and coordination.*
        - *github-updates: Automated feed of commits, pull requests, and issues from GitHub.*
- *JIRA:* Tasks are organized into sprints
- *GitHub Shinoaki0145/ChatBot_HelpDesk: Used to store source code and documentation:*
    - */src: Used to store source code.*
    - */docs: Used to store documentation, which has these subfolders:*
        - *Management: Storing planning documents, reports (weekly report, project status report, etc.).*
        - *Requirements: Storing all requirements, including vision documents and use cases.*
        - *Analysis and design: Storing all analysis and design related documents,*

including software architecture documents, UML models, and UI designs.

- *Test: Storing all test documents such as test plans, test cases, and test reports.*

- */pa: Including subfolders to store submissions. Each subfolder contains one PA submission.*