SE 4485: Software Engineering Projects

Fall 2024

Requirements Documentation

|  |  |
| --- | --- |
| Group Number | 7 |
| Project Title | CIO-Brain Architectural Assessment System |
| Sponsoring Company | The Fellows Consulting Group (FCG) |
| Sponsor(s) | Tom Hill |
| Students | Gehrig French  Brandon Hernandez  Debra Samia  Samuel Williford  Bilal Zubair |

**Requirements Documentation**

**Software Engineering Capstone Project**

CIO-Brain Architectural Assessment System

**Group 7**

Gehrig French

Brandon Hernandez

Debra Samia

Sam Williford

Bilal Zubair

Department of Computer Science

University of Texas at Dallas

800 West Campbell Road

Richardson, TX 75080

**Industry Sponsor**

Tom Hill

The Fellows Consulting Group (FCG)

7356 Lane Park Ct

Dallas, TX 75225

**ABSTRACT**

This document presents the initial functional and non-functional requirements for CIO-Brain, a conversational, private software architecture assessment LLM. It provides a preliminary understanding of the system’s functionality and constraints, which may evolve as we continue to collaborate with stakeholders. The document includes an introduction to its structure, a detailed use case model, supporting rationale, and outlines and special requirements and key non-functional criteria necessary for the project’s success.

**TABLE OF CONTENTS**

Title ............................................................................................................................................................. 1

Abstract ....................................................................................................................................................... 2

Table of Contents ........................................................................................................................................ 3

List of Figures ............................................................................................................................................. 3

List of Tables .............................................................................................................................................. 3

Introduction ................................................................................................................................................. 4

Use Case Model for Functional Requirements ........................................................................................... 4

Rational for your Use Case Model ............................................................................................................. ?

Non-Functional Requirements .................................................................................................................... ?

Evidence the Document has been placed under Configuration Management ............................................. ?

Engineering Standards and Multiple Constraints ........................................................................................ ?

Additional References ................................................................................................................................. ?

**LIST OF FIGURES**

**LIST OF TABLES**

**INTRODUCTION**

This document outlines the key components needed for the CIO-Brain Architectural Assessment System. It highlights the system’s main functionalities and expected non-functional aspects, based on our initial understanding of the requirements.

This document includes a use case model (graphical and text), a rationale for the model, and non-functional requirements. It also provides evidence of configuration management, engineering standards, multiple constraints, and additional references.

**USE CASE MODEL FOR FUNCTIONAL REQUIREMENTS**

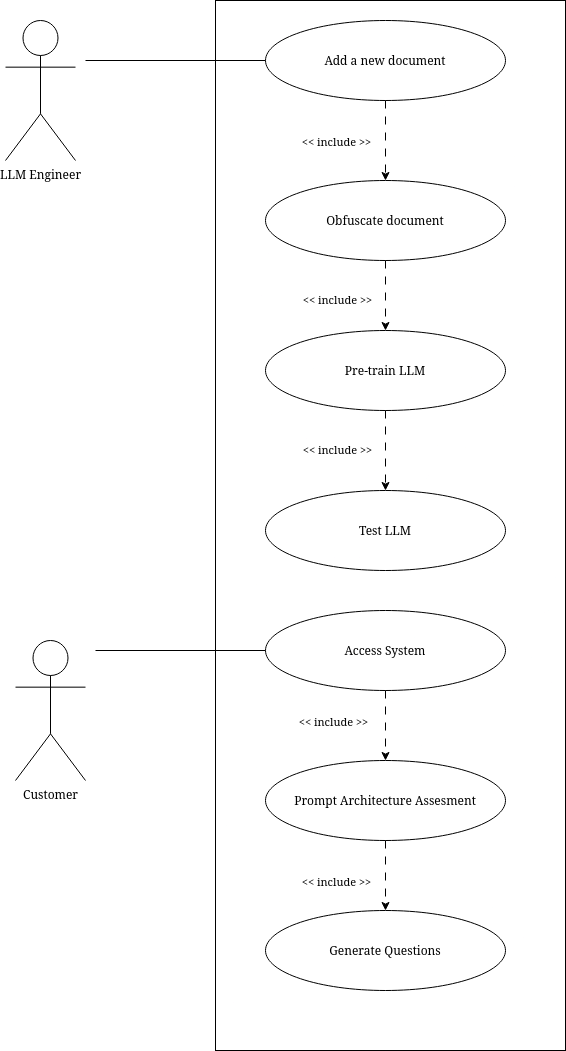


Figure 1: Graphical Use-Case Model

|  |  |
| --- | --- |
| **Use Case Name** | **Add New Document** |
| *Participating Actors* | LLM Engineer |
| *Entry Conditions(s)* | * The LLM Engineer is logged into the system * The system is available for document uploads |
| *Normal Flow of Events* | * The Engineer selects “Add Document Option”. * The system prompts the engineer to upload a document. * The engineer selects and uploads the document. * The system confirms the successful upload and stores the document for processing. |
| *Exit Conditions(s)* | * The document is successfully uploaded and stored, ready for obfuscation. |
| *Exceptions* | * If the document upload fails, the system notifies the engineer and requests a retry. * If the document format is unsupported, the system displays an error message. |
| *Special Requirements* | * The system must support multiple document formats (e.g., PDF, DOC, DOCX, XLS) |

Table 1: Add New Document Engineer Use Case

|  |  |
| --- | --- |
| **Use Case Name** | **Obfuscate Document** |
| *Participating Actors* | LLM Engineer |
| *Entry Conditions(s)* | * The document has been successfully uploaded. * The LLM Engineer is logged in and initiates the obfuscation process. |
| *Normal Flow of Events* | * The LLM selects “Obfuscate Document” for the uploaded document. * The system prompts the engineer to provide a list of words for obfuscation and their replacements. The engineer inputs the terms and confirms the list. * The system applies replacements to the document. * The system displays results to the engineer. * The engineer reviews changes and selects “Confirm” |
| *Exit Conditions(s)* | * The LLM engineer selects “Confirm” after reviewing the document changes. |
| *Exceptions* | * If the obfuscation process fails, the system logs an error and notifies the engineer. |
| *Special Requirements* | * The system must support multiple document formats * *Future improvement: The system may include generated suggestions based on natural language processing.* |

Table 2: Obfuscate Document Engineer Use Case

|  |  |
| --- | --- |
| **Use Case Name** | **Pre-Train LLM** |
| *Participating Actors* | LLM Engineer |
| *Entry Conditions(s)* | * The document has been obfuscated correctly. |
| *Normal Flow of Events* | * The engineer selects “Pre-Train LLM” * The system processes the obfuscated document and updates the LLM’s Knowledge * The system confirms that the LLM has been successfully trained. |
| *Exit Conditions(s)* | * The LLM’s knowledge base has been updated with the new obfuscated data. |
| *Exceptions* | * If the training fails, the system logs an error and notifies the engineer. |
| *Special Requirements* | * The training process should be performed in a timely and efficient manner. |

Table 3: Pre-Train LLM Engineer Use Case

|  |  |
| --- | --- |
| **Use Case Name** | **Test LLM** |
| *Participating Actors* | LLM Engineer |
| *Entry Conditions(s)* | * The LLM has been trained using obfuscated data. * The LLM Engineer selects “Test LLM” option |
| *Normal Flow of Events* | * The LLM Engineer initiates the test * The system generates sample outputs based on the test inputs. * The engineer reviews the sample outputs to verify that the new data has been incorporated * The engineer confirms that the LLM has incorporated the new data into the response. |
| *Exit Conditions(s)* | * The LLM has been confirmed to respond correctly, with no exposure of confidential or obfuscated data. |
| *Exceptions* | * If the LLM does not incorporate the new data, the LLM Engineer can review and adjust LLM training. |
| *Special Requirements* | * The system must provide a test environment isolated from production. * Test results must be logged for auditing and review.” |

Table 4: Test LLM Engineer Use Case

|  |  |
| --- | --- |
| **Use Case Name** | **Access System** |
| *Participating Actors* | Customer |
| *Entry Conditions(s)* | * The customer has valid login credentials. * The system is available and online. |
| *Normal Flow of Events* | * The customer enters their login credentials. * The system verifies the credentials. * The customer is granted access to the system’s dashboard. |
| *Exit Conditions(s)* | * The customer is successfully logged into the system. |
| *Exceptions* | * If the login credentials are incorrect, the system displays an error and prompts the customer to retry. * If the system is offline or unavailable, the system displays a relevant error message. |
| *Special Requirements* | * The system must ensure secure login using encryption. * The system must handle failed login attempts securely (e.g., lockouts after multiple failures). |

Table 5: Access System Customer Use Case

|  |  |
| --- | --- |
| **Use Case Name** | **Prompt Architecture Assessment** |
| *Participating Actors* | Customer |
| *Entry Conditions(s)* | * The customer is logged into the system and has access to the assessment feature. |
| *Normal Flow of Events* | * The customer selects the option to initiate an architectural assessment. * The system retrieves any existing data it has about the customer’s company. * The system asks the customer basic questions to gather project-specific information. (e.g., project name, goals, timeline). * The customer answers each question to provide necessary high-level project details. * The system confirms that the basic project information has been collected. |
| *Exit Conditions(s)* | * The system has gathered the basic project information needed for the architectural assessment. |
| *Exceptions* | * If the system doesn’t understand a response, or if the customer response provides insufficient information, the system asks clarifying questions to ensure that sufficient information is collected. |
| *Special Requirements* | * The system must securely store and retrieve company data. * The system should ensure that the basic questions are clear and sufficient to gather the necessary high-level project information. * The system should be capable of handling ambiguous or unclear responses and asking clarifying questions when needed. |

Table 6: Prompt Architecture Assessment Customer Use Case

|  |  |
| --- | --- |
| **Use Case Name** | **Generate Questions** |
| *Participating Actors* | Customer |
| *Entry Conditions(s)* | * The customer has responded to all initial, high-level project questions. * The system has enough data to generate relevant, architecture-specific questions. |
| *Normal Flow of Events* | * The system analyzes the completed basic project information. * The system generates an initial list of architecture-specific questions based on the project data. * The customer begins answering questions. * Based on the customer’s answers, the system may update the list of questions, generating follow-up questions as needed. * The customer continues answering until no further questions are generated. * The system outputs an architectural assessment based on the customer’s responses. |
| *Exit Conditions(s)* | * The customer has completed the question-answer process. * The customer receives the architectural assessment based on the gathered data. |
| *Exceptions* | * If the system doesn’t understand a response, or if the customer response provides insufficient information, the system asks clarifying questions to ensure that sufficient information is collected. |
| *Special Requirements* | * The system should ensure that questions are relevant, tailored, and adaptive based on the customer’s answers. * The questions generated for the customer must be simple and easy for the customer to understand and answer. * The system should dynamically generate follow-up questions based on customer responses. * The architectural assessment must be accurate and reflective of the customer’s input. |

Table 7: Generate Questions Customer Use Case

**RATIONALE FOR YOUR USE CASE MODEL**

This use case model outlines the system's basic functionality, designed to evolve as requirements change. It balances simplicity for the Customer with control for the LLM Engineer. The engineer manages document uploads, obfuscation (with future potential for LLM-driven suggestions), model training, and testing to ensure data security. For customers, the system provides easy access, collects basic project information in plain language, and generates relevant follow-up questions without using jargon.

The model is built to accommodate future features like automatic obfuscation suggestions and clarifying questions. It ensures the system remains user-friendly while maintaining secure data handling for the engineer.

**NON-FUNCTIONAL REQUIREMENTS**

**Performance**

The system should run quickly and smoothly, especially when uploading documents, processing data, and generating assessments. It should handle tasks efficiently even as the amount of data grows. Additionally, the system’s generated output should be high-quality, providing relevant questions and reliable advice to customers.

**Usability**

The system must be easy to use for both technical and non-technical users. The interface should be simple, clear, and free of jargon, guiding users through tasks with minimal effort.

**Security**

Data security is critical. All sensitive information should be protected through encryption, and obfuscated data must be securely handled to ensure no leaks or recoverability of confidential information.

**Availability**

The system should be available nearly all the time, with minimal downtime. Any maintenance or outages should be communicated in advance, and the system should recover quickly from failures.

**Reliability**

The system should be available with minimal downtime. Any maintenance or outages should be communicated in advance, and the system should recover quickly from failures.

**Scalability**

As usage grows, the system must be able to handle more users and larger amounts of data without slowing down. It should easily scale up or out as needed**.**

**Compatibility**

The system should work with different browsers, operating systems, and file formats (e.g., PDF, DOCX), and integrate smoothly with other tools or platforms.

**Portability**

The system should be easy to deploy across different environments or platforms with minimal changes, whether it’s in the cloud or on local servers.

**EVIDENCE THE DOCUMENT HAS BEEN PLACED UNDER CONFIGURATION MANAGEMENT**

**ENGINEERING STANDARDS AND MULTIPLE CONSTRAINTS**

* IEEE Std 830-1998: Software Requirements [[pdf](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=720574)]
* IEEE Std 29148: Requirements Engineering [[pdf](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6146379)]

**ADDITIONAL REFERENCES**

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