**Software Design**

**Document**

**for**

**Abbott & Sugar Project**

**Version 4**

**Prepared by:**

**Janis Garcia, Virginia Guadalupe Gonzalez, Ulises Gutierrez, James Hy, Charlie Martinez Dominguez, Michael Perez, Dylan Paul Tomasello, Omar Tovar, Theodore Tran, Thomas Yeung**

**Abbott & Shuger**

**September 23, 2023**

Table of Contents....................................................................................................................pg. 2

Revision History......................................................................................................................pg. 3

1. Introduction................................................................................................................pg. 4
	1. Purpose...........................................................................................................pg. 4
	2. Document Conventions…………………….............................................................pg. 4
	3. Intended Audience and Reading Suggestions.................................................pg. 4
	4. System Overview.............................................................................................pg. 4
2. 2. Design Considerations........................................................................................................pg. 5
	1. Assumptions and dependencies.....................................................................pg. 6
	2. General Constraints........................................................................................pg. 6
	3. Goals and Guidelines......................................................................................pg. 7
	4. Development Methods...................................................................................pg. 7
3. 3. Architectural Strategies......................................................................................................pg. 7
4. 4. Policies and Tactics.............................................................................................................pg. 9
	1. Specific Products Used...................................................................................pg. 9
	2. Requirements traceability..............................................................................pg. 10
	3. Testing the software......................................................................................pg. 10
	4. Engineering trade-offs...................................................................................pg. 10
	5. Guidelines and conventions.......................................................................... pg. 10
	6. Protocols...................................................................................................... pg. 10
	7. Maintaining the software.............................................................................. pg. 10
	8. Interfaces...................................................................................................... pg. 10
	9. System's deliverables................................................................................... pg. 10
	10. Abstraction................................................................................................... pg. 10
5. 5. User Interface …………........................................................................................................pg. 12
6. 6. Database Design................................................................................................................ pg. 13
7. 7. Glossary............................................................................................................................. pg. 13
8. 8. References......................................................................................................................... pg. 13

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Reason For Changes | Version |
|  Janis Garcia |  9/3/2023 |  Intro/ Design Considerations/ Architectural Strategy |  1 |
| Janis Garcia | 10/3/2023 | System Architecture/ Policies and Tactics | 2 |
|  Janis Garcia |  11/3/2023 |  Adding in last sections of information |  3 |
| Janis Garcia | 12/7/2023 | Finishing touches/ last minute input | 4 |

**1. Introduction**

**1.1 Purpose**

The goal of our project as a team was to make the lives of university students like ourselves easier using AI. With all that we must worry about, we wanted to use what we have learned in our time at Cal State LA. Our group may be a little different from the others in that we split our team into four smaller teams in order to tackle different strategies. Such as providing a chatbot as an alternative to an advisor, using databases and chaining prompts to figure out what your educational plan should look like, or creating a virtual education assistant to help you with your classes.

**1.2 Document Conventions**

Main topics are in type 20x font and bolded. Sub-topics are in type 14x font and bolded. All remaining text is in type 12x font.

**1.3 Intended Audience and Reading Suggestions**

This report is intended for developers, university students, and by extension anyone in relation to Cal State LA or any other college or university, interested in implementing new strategies for improving university student lives. This SDD contains different strategies for improving students’ use of time using AI. It is not pertinent to read each section in order other than the intro and conclusion, each section is its own separate entity.

**1.4 System Overview**

In this project several software systems are used:

* Open AI
* cohere
* LLaMa by Meta
* LangChain
* Hugging Face
* Colab

These will be discussed in more detail in section 4.1.

**2. Design Considerations**

As a team when considering what real-life scenario, we wanted to improve, we considered our project Liaison and his interest in AI. Most of us did not have much experience or knowledge of AI. AI Is a field that has a lot to be discovered so in order to push ourselves it was decided that we first spend time researching and learning about different tools, systems, and large language models so that we could get a better understanding of what the current world of AI looks like. We would accomplish this by testing different strategies and LLMs and pushing them to perform different tasks. After becoming better informed we could confidently use what we have learned to address a singular topic which we felt promising.

**2.1 Assumptions and Dependencies**

1. **Internet Connection:**
	* **Dependency:** A stable internet connection is required to access the model.
	* **Consideration:** Offline usage may not be possible, and any interruptions in internet connectivity can impact the ability to interact with the model.
2. **Availability:**
	* **Dependency:** Continued availability and support of the model.
	* **Consideration:** Changes in service, versions, or policies could impact the integration and functionality.
3. **API Rate Limits:**
	* **Dependency:** Adherence to rate limits.
	* **Consideration:** Frequent requests beyond the rate limits may result in service disruptions or additional costs.
4. **Model Updates:**
	* **Dependency:** Updates or new versions of the model.
	* **Consideration:** Changes in the model may affect the behavior and responses, and users need to adapt to new versions as necessary.
5. **Token Limits:**
	* **Dependency:** The model has a maximum token limit for each interaction.
	* **Consideration:** Long conversations may be truncated, and the need to manage interactions within the token limits.
6. **Costs:**
	* **Dependency:** Usage of the API is subject to associated costs.
	* **Consideration:** The need to be aware of pricing models, potential usage costs, and stay within allocated budgets.

**2.2 General Constraints**

Some constraints we encountered frequently were cost, the cost of using AI tools was a challenge as well as the use of tokens. Tokens are the fundamental units of text in ChatGPT and other similar language models, such as GPT-3.5. They play a critical role in measuring the length of input and output text, determining billing costs, and ensuring that your conversations fit within the model's limitations. The goal was to use as few tokens as possible since ChatGPI API charges $0.002 per one thousand tokens. When it comes to other tools like OpenAI and other technology there was a concern for dependence on data. Any model would require a large amount of data to be trained effectively, this leads to AI model bias and deciding what data to use to avoid any bias. Some ethical issues also arise when it comes to training models and where the data is obtained from.

**2.3 Goals and Guidelines**

1. **Meet deadlines:**
	* **Goal:** Brainstorm and design a system in the allocated amount of time.
	* **Guidelines:** Setting realistic standards and time frames.
2. **Scalability and Flexibility:**
	* **Goal:** Design a system that can scale to accommodate future requirements and is flexible to changes.
	* **Guidelines:** Discuss design decisions that support scalability, modularization, and flexibility. Consider future enhancements and how the design accommodates them.
3. **Maintainability:**
	* **Goal:** Facilitate easy maintenance of the software system over its lifecycle.
	* **Guidelines:** Discuss coding standards, naming conventions, and provide documentation on how to maintain and extend the system.
4. **Performance Considerations:**
	* **Goal:** Address performance requirements and considerations in the design.
	* **Guidelines:** Discuss algorithms, data structures, and design choices that impact system performance. Include any trade-offs made to achieve performance goals.
5. **Error Handling and Reliability:**
	* **Goal:** Design the system to handle errors gracefully and ensure overall reliability.
	* **Guidelines:** Document error-handling mechanisms, fault-tolerant features, and recovery strategies. Include information on logging and monitoring.

**2.4 Development Methods**

The development method which best encompasses our workflow is the waterfall model. When dealing with a topic that is so new to us, we found a linear approach was desirable. We wanted to research many different topics, so our strategy became to divide and conquer. We divided our team into four sub teams and began each week by planning what each team was going to research or test that week. The plan was to research first, design later, implement next, and test last.

**3. Architectural Strategies**

We as a team spent the majority of the semester learning about LLMs and testing different models and machine learning platforms so we could decide on what would be most efficient in our endeavors. Ultimately, we will be creating our own version of a chatbot or virtual assistant which will be used by students. In order to meet deadlines, it was necessary to scale back, we are currently only using data which corresponds to what is relevant to CS students. We were able to produce a rough strategy for the organization of our system.

* **Design Decision 1: Natural Language Processing (NLP) as the Core**

**Strategy:** Leverage Natural Language Processing (NLP) as the core technology for understanding and generating human-like responses.

**Reasoning:**

* *User-Friendly Interaction:* NLP allows the virtual assistant to understand and respond to users in a more natural and human-like manner, enhancing user experience.
* *Intent Recognition:* Enables the extraction of user intent from messages, allowing the virtual assistant to provide relevant responses based on the context.
* **Design Decision 2: Intent Recognition and Entity Extraction**

**Strategy:** Implement intent recognition and entity extraction to understand user requests and extract relevant information.

**Reasoning:**

* *Contextual Understanding:* Identifying user intents helps the virtual assistant understand the purpose of the conversation, allowing it to provide appropriate responses.
* *Personalization:* Extracting entities (such as dates, locations, courses, important data) allows the virtual assistant to personalize responses and take specific actions based on user input.
* **Design Decision 3: Dialog Management System**

**Strategy:** Incorporate a dialog management system to maintain context and manage multi-turn conversations.

**Reasoning:**

* *Context Retention:* Enables the virtual assistant to retain context across multiple user interactions, ensuring coherent and contextually relevant responses.
* *Seamless Conversations:* Facilitates smooth transitions between user turns, allowing the virtual assistant to understand and respond coherently in ongoing conversations.
* **Design Decision 4: Integration with External Systems and APIs**

**Strategy:** Integrate the virtual assistant with external systems and APIs for fetching data, performing actions, or providing additional information. Such as ChatGPT API.

**Reasoning:**

* *Enhanced Functionality:* Allows the virtual assistant to perform tasks beyond simple conversation, such as retrieving real-time information, or interacting with external databases i.e., the University catalog.
* *Dynamic Responses:* Enhances the virtual assistant's capabilities by tapping into external resources, making it a more powerful and versatile tool.
* **Design Decision 5: User Authentication and Authorization**

**Strategy:** Implement user authentication and authorization mechanisms for secure interactions, especially in scenarios where sensitive information is involved.

**Reasoning:**

* *Security:* Ensures that only authorized users can access specific functionalities or receive certain information. Only students enrolled in Cal State LA would be able to attain certain information.
* *Personalization:* Supports personalized interactions by recognizing individual users and their preferences.

**4. Policies and Tactics**

**4.1 Choice of which specific products used.**

1. **OpenAI:**
* An artificial intelligence research laboratory and technology company.
* Known for developing powerful language models, including GPT (Generative Pre-trained Transformer) models.
* Provides API access to its language models for natural language processing tasks.
1. **Cohere:**
* A company specializing in natural language processing (NLP) technologies.
* Offers NLP models and APIs that enable developers to integrate language understanding capabilities into applications.
* Focuses on building efficient and scalable NLP solutions.
1. **LLaMa by Meta:**
* LLaMa (Language Model for Many Applications) is a project by Meta (formerly Facebook).
* Aims to create a universal language model that can understand and generate text for a wide range of applications.
* Part of Meta's broader efforts in advancing natural language understanding and AI research.
1. **LangChain:**
* LangChain is a blockchain-based platform for natural language processing and understanding.
* Focuses on leveraging blockchain technology to enhance language-related applications, including translation services and content analysis.
1. **Hugging Face:**
* A platform and community for machine learning practitioners.
* Known for its library and model hub that provides access to a wide range of pre-trained models, including those for natural language processing.
* Supports model sharing, collaboration, and the development of state-of-the-art machine learning applications.
1. **Colab (Google Colaboratory):**
* A free, cloud-based platform provided by Google for developing and running Jupyter notebooks.
* Offers access to GPUs and TPUs for machine learning tasks.
* Widely used for collaborative coding and experimentation in machine learning and data science.

**4.2 Plans for ensuring requirements traceability.**

Our plans for ensuring requirements traceability are the following but are not limited to:

1. Providing Design Rationales:
	* For each design decision or element, we must explicitly state the rationale behind it.
	* Connect design rationales back to the specific requirements they address or support.
2. Linking Design Elements to Requirements:
	* Using hyperlinks, cross-references, or annotations within the design document to link directly to the relevant requirements (textual and/ or visual, depending on the document format.)
3. Including Traceability Information in Design Descriptions:
	* Within the description of design elements, explicitly mentioning the requirements they address or implement.
	* Providing clear and concise explanations of how each design component fulfills specific requirements.

**4.3 Plans for testing the software.**

When testing the software, we may begin with basic functional testing such as:

1. **Checking if the virtual assistant understands and responds correctly to basic user inputs.**

We will do so by:

* + Greeting the virtual assistant and observing its initial response.
	+ Asking common questions and assessing the accuracy of responses.
	+ Testing predefined commands or actions.
1. **Verifying that the virtual assistant correctly recognizes user intents.**

We will do so by:

* + Posing a question or statement related to different intents and checking the virtual assistant's responses.
1. **Testing the virtual assistant's ability to handle variations in user input.**

We will do so by:

* + Providing inputs with spelling mistakes, abbreviations, or synonyms.
	+ Observe how the virtual assistant adapts to different phrasings of the same query.
1. **Assessing how the virtual assistant handles errors or misunderstandings.**

We will do so by:

* + Intentionally providing ambiguous or unclear inputs.
	+ Checking if the virtual assistant provides informative error messages or asks for clarification.
1. **Evaluate the flow of a conversation and context retention.**

We will do so by:

* + Engage in multi-turn conversations and observe how well the virtual assistant maintains context.
	+ Test interruptions or changes in conversation topics.
1. **Verify any integrations with external systems or APIs.**

We will do so by:

* + Test functionalities that involve external data retrieval or actions.
	+ Ensure that integrated services are responding as expected.

**5. User Interface**

Our ideal virtual assistant/ virtual assistant interface would be designed to seamlessly blend advanced technology with user-friendly interactions, creating an intuitive and engaging experience. This interface prioritizes natural language understanding, multimodal interactions, and personalization to cater to diverse user needs.

**6. Database Design**

When it comes to building a virtual assistant/ virtual assistant we considered the use of using relational database as well as vector databases. Ultimately, a hybrid approach where a combination of both databases seemed efficient, in order to benefit from the strengths of each. For example, using a relational database for structured data like user profiles and conversation history, and a vector database for storing and processing high-dimensional vectors related to NLU or recommendation systems. We want the benefits of conversation history with the benefits of semantic search. When inquiring about what courses to take or a specific question on a bit of text from a textbook, we want the system to be able to remember past info but also be able to understand underlying meaning of the query and identify related concepts.

**6. Glossary**

1. **AI (Artificial Intelligence):**
	* *Definition:* Artificial Intelligence refers to the simulation of human intelligence in machines that are programmed to think and learn like humans.
2. **SRS (Software Requirements Specification):**
	* *Definition:* A document that describes the intended purpose, functionality, and requirements of a software system.
3. **LLM (Large Language Model):**
	* *Definition:* A type of artificial intelligence model designed to understand and generate human-like language.
4. **NLP (Natural Language Processing):**
	* *Definition:* A field of AI that focuses on the interaction between computers and humans through natural language.
5. **API (Application Programming Interface):**
	* *Definition:* A set of rules and tools that allows different software applications to communicate with each other.
6. **Token:**
	* *Definition:* In the context of language models, a token is a unit of text used for processing. It can be as short as one character or as long as one word.
7. **ChatGPT:**
	* *Definition:* A language model developed by OpenAI, capable of generating human-like text responses.
8. **Waterfall Model:**
	* *Definition:* A sequential software development process, where progress is seen as flowing steadily downwards through several phases, such as conception, initiation, analysis, design, construction, testing, and maintenance.
9. **Intent Recognition:**
	* *Definition:* The process of identifying the user's intention or purpose behind a given input.
10. **Entity Extraction:**
	* *Definition:* The process of identifying and classifying entities (e.g., dates, locations, courses) in text.
11. **Dialog Management System:**
	* *Definition:* A system that manages the flow of conversation, ensuring context retention and coherent responses in multi-turn interactions.
12. **Requirements Traceability:**
	* *Definition:* The ability to trace and document the life of a requirement, linking it back to its source and forward to the design and testing.
13. **GPT (Generative Pre-trained Transformer):**
	* *Definition:* A type of neural network architecture used in large language models for natural language processing tasks.
14. **Multimodal Interactions:**
	* *Definition:* Interactions that involve multiple modes of communication, such as text, images, and voice.
15. **User Authentication:**
	* *Definition:* The process of verifying the identity of a user, ensuring that only authorized individuals can access specific functionalities.
16. **User Authorization:**
	* *Definition:* The process of granting or denying access to specific functionalities based on the authenticated user's permissions.
17. **Functional Testing:**
	* *Definition:* Testing the software to ensure that it functions according to specifications and requirements.
18. **User Interface:**
	* *Definition:* The point of interaction between the user and the software, including screens, pages, and visual elements.
19. **Multiturn Conversations:**
	* *Definition:* Interactions that involve multiple back-and-forth exchanges between the user and the system.
20. **Sensitivity to API Rate Limits:**
	* *Definition:* The awareness and adherence to limitations on the number of requests that can be made to an API within a specific timeframe.
21. **Bias in AI Models:**
	* *Definition:* The presence of unfair or prejudiced behavior in AI models, often resulting from biased training data.
22. **Ethical Issues in AI:**
	* *Definition:* Concerns and considerations related to the moral and societal implications of AI technologies.

**7. References**

1. “Building Generative AI Applications with Gradio.” *DeepLearning.AI*, www.deepLearning.ai/short-courses/building-generative-ai-applications-with-gradio/. Accessed 6 Dec. 2023.
2. “Generative AI for Everyone.” *DeepLearning.AI*, www.deeplearning.ai/courses/generative-ai-for-everyone/?utm\_campaign=genai4e-launch&utm\_medium=email&\_hsmi=280781425&\_hsenc=p2ANqtz-\_HjwSZ7YWPUM119AW7Wt3k8SVF8WZfPyI7dkRKCeXJ7zdoujKwLtKNQxbNLhZQZLFpIqBqWaf1SlAGoOFUmryueYeYfw&utm\_content=280781425&utm\_source=hs\_email. Accessed 6 Dec. 2023.
3. “How to Write Efficient Python Code: A Tutorial for Beginners.” *KDnuggets*, www.kdnuggets.com/how-to-write-efficient-python-code-a-tutorial-for-beginners. Accessed 6 Dec. 2023.
4. “Langchain Blog.” *LangChain Blog*, blog.langchain.dev/. Accessed 6 Dec. 2023.
5. LangChain. “Langchain Templates.” *LangChain Blog*, LangChain Blog, 10 Nov. 2023, blog.langchain.dev/langchain-templates/.
6. Muhammad Raza. (2023, November 19). *One liners python edition: Muhammad*. https://muhammadraza.me/2023/python-oneliners
7. *New Models and Developer Products Announced at DevDay*, openai.com/blog/new-models-and-developer-products-announced-at-devday. Accessed 6 Dec. 2023.
8. “Vector Databases: From Embeddings to Applications.” *DeepLearning.AI*, www.deepLearning.ai/short-courses/vector-databases-embeddings-applications/. Accessed 6 Dec. 2023.
9. “Welcome.” *Welcome - MindStudio Docs*, docs.youai.ai/youai/. Accessed 6 Dec. 2023.
10. Weng, Lilian. “Prompt Engineering.” *Lil’Log (Alt + H)*, 15 Mar. 2023, lilianweng.github.io/posts/2023-03-15-prompt-engineering/?ref=blog.langchain.dev%29.