

Generative AI to Assess Learning

DESIGN DOCUMENT

sdmay24-49

Client: Dr. Henry Duwe (duwe@iastate.edu)

Advisor: Dr. Mat Wymore (mlwymore@iastate.edu)

Akpobari Godpower – Team Leader

Abram Demo - Frontend/Programmer

Drake Rippey - Tester/Programmer

Alex Vongphandy - Programmer

sdmay24-49@iastate.edu

<https://sdmay24-49.sd.ece.iastate.edu/>

Revised: 12/3/23

Executive Summary

Development Standards & Practices Used

- Agile Methodology
- Git Version Control
- REST API standards
- CI/CD to automate regression testing and integration

Summary of Requirements

- Proof of Concept Design
- Rubric Upload and Assessment Configuration
- Topic question and follow up question generation
- Correct Grading of Student Responses
- Submission of grades and chat log to canvas

Applicable Courses from Iowa State University Curriculum

- COMS 127 - Introduction to Computer Programming
- COMS 227 - Object-oriented Programming
- COMS 228 - Introduction to Data Structures
- COMS 309 - Software Development Practices
- COMS 319 - Construction of User Interfaces
- COMS 472 - Principles of Artificial Intelligence
- SE 329 - Software Project Management

New Skills/Knowledge acquired that was not taught in courses

- Interacting with Language Learning Models
- Canvas API
- Cost Analysis
- Server Hosting
- Requirements Research

- OpenAI API

Table of Contents

1	Team	5
1.2	REQUIRED SKILL SETS FOR YOUR PROJECT	
1.3	SKILL SETS COVERED BY THE TEAM	
1.4	PROJECT MANAGEMENT STYLE ADOPTED BY THE TEAM	
1.5	INITIAL PROJECT MANAGEMENT ROLES	
2	Introduction	6
2.1	PROBLEM STATEMENT	
2.2	REQUIREMENTS & CONSTRAINTS	
2.3	ENGINEERING STANDARDS	8
2.4	INTENDED USERS AND USES	
3	Project Plan	9
3.1	Task Decomposition	9
3.2	Project Management/Tracking Procedures	11
3.3	Project Proposed Milestones, Metrics, and Evaluation Criteria	12
3.4	Project Timeline/Schedule	14
3.5	Risks And Risk Management/Mitigation	16
3.6	Personnel Effort Requirements	18
3.7	Other Resource Requirements	20
4	Design	20
4.1	Design Context	20
4.2	Design Complexity	21
4.3	Modern Engineering Tools	22
4.4	Design Context	23
4.5	Prior Work/Solutions	25
4.6	Design Decisions	25

4.7 Proposed Design	26
Design 0	26
Design Visual and Description	26
Functionality	
Design Visual and Description	
Design 1	29
4.8 Technology Considerations	30
LLM Selection and Evaluation	32
Programming Language	32
Backend Framework Selection	32
Data Framework Selection	32
Frontend Framework Selection	33
Database Selection	33
4.9 Design Analysis	34
5 Testing	35
5.1 Unit Testing	35
5.2 Interface Testing	35
5.3 Integration Testing	35
5.4 System Testing	36
5.5 Regression Testing	37
5.6 Acceptance Testing	38
5.7 Security Testing	38
5.8 Results	39
6 Implementation	39
7 Professionalism	39
7.1 Areas of Responsibility	39
7.2 Project Specific Professional Responsibility Areas	41
7.3 Most Applicable Professional Responsibility Area	43
8 Closing Material	44

8.1 Discussion	44
8.2 Conclusion	44
8.3 References	45
8.4 Appendices	45
8.4.1 Team Contract	45

1 Team

1.1 TEAM MEMBERS

- Akpobari Godpower – Team Leader
- Abram Demo
- Drake Rippey
- Alex Vongphandy

1.2 REQUIRED SKILL SETS FOR YOUR PROJECT

REST API's. Language Learning Models. Python, Java and other various languages. Git. Server Management. React. NoSQL.

1.3 SKILL SETS COVERED BY THE TEAM

- Python Programming Experience - All
- Java Programming Experience - All
- REST API Experience - All
- React - All
- Server Management - All
- ChatGPT Knowledge - Akpobari

- NoSQL - All

1.4 PROJECT MANAGEMENT STYLE ADOPTED BY THE TEAM

We will be working in an waterfall agile hybrid environment to monitor project progress and to be able to check in on each other. It won't be a traditional agile in the sense that we will not be having progress reports every day but we will be able to make up for that by communicating often through other channels such as Discord. We will set up process deadlines to also keep us on track and put a time block on things so we don't spend too long on anything. We plan on having weekly meetings for most of the semester.

1.5 INITIAL PROJECT MANAGEMENT ROLES

- Akpobari Godpower – Team Leader/Programmer
- Abram Demo - Front End Programmer
- Drake Rippey - Tester/Programmer
- Alex Vongphandy - Programmer/Server Management

2 Introduction

2.1 PROBLEM STATEMENT

Our project's main objective is to provide an interactive assessment experience for students while maintaining a low overhead cost for instructors. By using Artificial Intelligence, this project aims towards automating test generation and grading of assessments. Assessment takers would experience adaptive learning which would evaluate and validate their knowledge. Simply, we are making use of AI to create assessments that should make the process easier for students and instructors.

2.2 REQUIREMENTS & CONSTRAINTS

Integration with Canvas API (functional)

Use API to record the final grade and verify that they are eligible to be assigned a final grade.

Automated Interactive Assessment Generation (functional)

Able to allow instructors to specify topic and optional assessment settings like rubric, initial prompt types (multiple choice), level of interaction depth, max time limit per assessment

Automated Assessment Grading (functional)

AI evaluate entire interaction to assess students' knowledge based on rubric

Response Time (quantitative)

Should be fast and responsive, each question should respond within 30 seconds.

Cost (constraint)

Maximum \$3 per assessment per student

Instructor needs to be able to review interactions (functional)

Student continues exam if the site is closed (functional)

Usability (subjective)

Easy for instructors to use to create and review assessments.

Security (non-functional)

Stay in compliance with FERPA as implemented by Iowa State University policies and especially dealing with student data.

Reliability (non-functional)

Ensure the system has high availability to deal with influx of test generation and grading.

Scalability (non-functional)

System should be able to scale up to 500 students taking the assessment.

Cost (constraint)

Maximum \$3 per assessment per student, we estimate this number would be similar to a student purchasing a textbook if an instructor assigns 10-12 assessments a semester. By our calculations, \$3 an assessment will give enough to complete the task with some money left over for a profit margin for the program. This was calculated by estimating the cost of an ideal interaction with the student provided 10 questions generated and 4 questions. It was also assumed that this assessment would occur multiple times during the semester therefore included in the total cost per student

API Ability (Constraint)

One of the constraints we will be experiencing within this project would be the capability of the API. The predictive ability of the API is important because it contributes towards our systems ability understand the assessment configurations outlined by the instructor to generate ideal question

2.3 ENGINEERING STANDARDS

IEEE 7001 - Transparency of Autonomous Systems

IEEE 7002 - Data Privacy or ISO 29100

OpenAPI Standards

WCAG (Web Content Accessibility Guidelines)

OAuth2

HTTPS

LTI

2.4 INTENDED USERS AND USES

The main benefactor of this project would be instructors who want to gauge the understanding of their students. class coordinators would also be interested in the existence of this technology because it would make it easier to plan their course content in a way that is both beneficial for the students and the instructors. It will be used in a way that incorporates conversation into their learning and by involving conversation as a mechanism of the assessment allows for the instructor and student to clearly dictate whether they understand the content.

3 Project Plan

3.1 TASK DECOMPOSITION

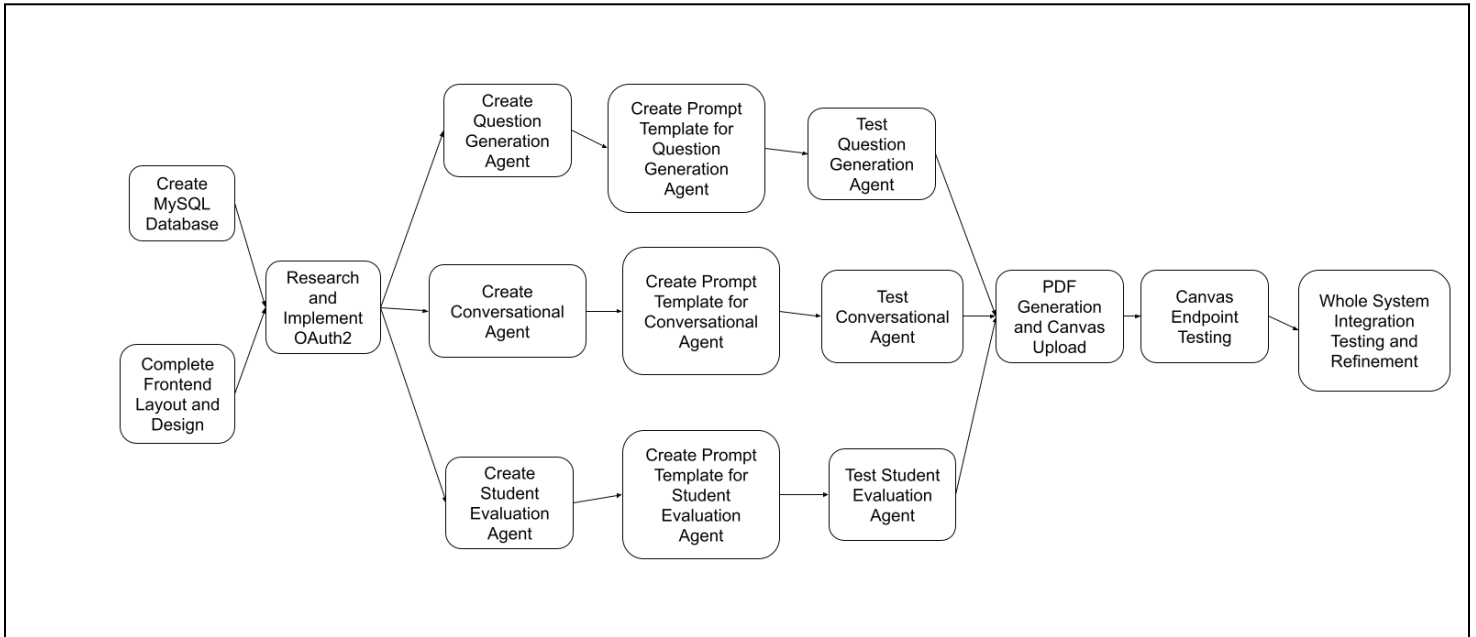


Figure 1. Task Decomposition Outline

Frontend Tasks

- Frontend Deployment
 - Description: Implement web page hosting for assessment configurations, canvas quiz linkage, rubric upload for instructors, and chat interface for students
 - Justification: Ensures frontend interface and structure is accessible online allowing for testing and development.
- Student and Instructor Authentication
 - Description: Implement user authentication for students and instructors.
 - Justification: Necessary to secure access to the platform and ensure only authorized users can perform actions.
- Rubric Upload Interface

- Description: Create an interface for retrieving the desired rubric from canvas. Provided a canvas link, the rubric will be retrieved from canvas and sent to the backend.
- Justification: Rubrics are essential for the AI to understand the criteria to assess a student's understanding.
- Quiz Linking
 - Description: a feature that allows instructors to link assessments in canvas with the rubric provided by the instructor
 - Justification: In order for the AI agent to upload the pdf and grades through Rest API calls, the quiz must be linked to a quiz on canvas so the grades and pdf can be posted.
- Assessment Configuration
 - Description: Configure assessment details such as number desired questions and level of detail per question.
 - Justification: Necessary for instructors to articulate what they would like the student to show that they understand.
- Chat Interface
 - Description: The student will be asked questions generated by the AI and will be given follow up questions.
 - Justification: The chat interface facilitates real-time communication between the AI and the student. Chat history will be used to assess student understanding in relation to the criteria and generate a pdf

Backend Tasks

- REST API Configuration/Setup
 - Description: Setup necessary APIs to communicate with front and backend and with other external systems like Canvas and OpenAI.
 - Justification: Allows exchange of data between different components of the system and external entities.
- Database Deployment
 - Description: Design, deploy, and configure database to store essential assessment data.
 - Justification: Needed for storing data locally.
- Prompt Engineering
 - Description: Develop, refine, and test AI prompts to ensure quality and satisfaction in accordance with requirements. Verify meaningful responses and if AI behaves as intended.
 - Justification: Essential in guiding AI in crafting questions in regards to requirements.
- Install and configuration of Langchain on a server
 - Description: The langchain package has to be installed on a server

- Justification: To use any of the features the langchain framework provides it must be installed.
- Langchain connection with OpenAI
 - Description: An OpenAI key has to be embedded into the langchain framework.
 - Justification: Required for quiz question generation, chat history assessment, and possibly pdf generation.
- Rubric Application
 - Description: AI Agent will utilize the rubric provided by the instructor and apply it to the chat history of an interaction with a student.
 - Justification: Required for assessment of students' understanding. Results of the assessment will then be sent to canvas.
- Agent tool to send post requests containing pdf and score to canvas
 - Description: A langchain agent tool that will use a post request to send the grade for the linked quiz and the pdf of the entire conversation given the grade of the student and chat history pdf
 - Justification: It is required to show both the instructor and the student the grade of the students' understanding.
- Agent tool to analyze student conversation
 - Description: Given the chat history of a student interaction with the api and the rubric provided by the instructor, the tool will use an AI agent to categorize the understanding of a student based on the rubric.
 - Justification: Automation of responses based on rubrics streamlines the grading process.
- PDF generation from chat
 - Description: After a student answers questions generated by the AI, the chat history will be converted into a PDF file.
 - Justification: The pdf can be reviewed by the instructor in the class to ensure that the AI was asking the correct questions.
- Establish Langchain chaining for chat interface
 - Description: The chaining feature of the langchain framework will be used to chain the questions and responses in the chat.
 - Justification: Chaining will be used to create the follow up questions based on the answers provided by the student.

3.2 PROJECT MANAGEMENT/TRACKING PROCEDURES

We will be using a waterfall-agile hybrid as the project management style of our project. By incorporating agile and waterfall methodologies during development it can provide flexibility to adapt to changing requirements, potential technical constraints or unexpected obstacles. Waterfall also allows us to set clear project milestones and deadlines

which can be helpful for our project that has deadlines and technical requirements to meet.

Our group will use Gitlab planning features. The issues board is where we will write up a view of potential problems that arise within the project as well as the requirements for future features. The issues board is where we will see the list of issues that need to be solved and the milestones is where we will outline the milestones for our project. We will also use the wiki features for the project to upload any documentation pertaining to solving past and future issues on the board.

3.3 PROJECT PROPOSED MILESTONES, METRICS, AND EVALUATION CRITERIA

Milestones:

1. Prompt Engineering
 - a. Creating initial set of prompts to test capabilities
 - b. Refining prompts based on testing
 - c. Identify and scope constraints based on prompt outputs
 - d. Quality Assurance of prompts and responses
 - e. Constraining scope of prompt outputs
 - f. Develop follow up question generation
2. Langchain Agent Implementation
 - a. Rubric interpretation
 - b. Chat assessment
 - c. PDF generation
 - d. Posting PDF and grades on canvas
3. AI Integration
 - a. Integrate OpenAI with Langchain and testing
 - b. Automated Quiz question generation with a 70% accuracy
 - c. Validate AI adaptive testing methods
 - d. Optimize Token Usage: Implement constraints or strategies to minimize the amount of tokens without compromising quality.
 - i. The maximum amount of tokens a student may use is \$3 per quiz.

4. Automated Grading System
 - a. Ensure grading accuracy based on rubric and standards
5. Post-Assessment Report
 - a. Ensure instructors and students can review assessments after completion
 - b. Implement report generation
6. Integration with Canvas
 - a. Initial integration with Canvas
 - b. Validate functionality with frontend, grading, and feedback upload.
7. Testing and Validation
 - a. Unit testing and acceptance testing

Metrics:

1. Chat Interactions
 - a. Accuracy of generated questions generated based on the rubric provided by the instructor.
 - i. The accuracy of questions generated shall be determined by the number of correct questions pertaining to the content provided by the instructor out of 10 questions generated.
 - b. The coherence of the follow up questions based on the response of the student. Categories include, but are not limited to “chat provides a probing question”, “chat changes topic”, “chat diverts off of topic”
2. Automated Grading System
 - a. Accuracy of grading”
 - i. The accuracy of the grading system will be measured by out of 4 interactions with a student with similar responses and follow up questions
 - b. Time taken to grade assessments
 - i. The metric of time taken to grade assessment shall be 80% faster than manual grading by the instructor.
3. Post-Assessment Report
 - a. Number of successful generation of post-assessment chat PDFs

- i. 95% success rate or higher when generating PDFs without errors or missing data
- 4. Integration with Canvas
 - a. Successful Uploads of Post-Assessment PDF
 - i. Must be 98% success rate without failures or errors
 - b. User Feedback on Canvas Integration
 - i. Percentage of users who find the Canvas Integration efficient
 - 1. 90% positive feedback from users

3.4 PROJECT TIMELINE/SCHEDULE

Semester 1

Task	Week (Semester 1)														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Requirements Engineering	X	X	X												
Research Large Language Models			X	X											
Research Data Frameworks			X	X											
Research Canvas LMS API			X	X	X	X	X	X							
Research Risk and Mitigations			X	X	X	X	X	X	X	X	X				

Infrastructure Setup			X	X											
Prototyping				X	X	X	X	X	X	X	X	X	X	X	X
Prompt Engineering				X	X	X	X	X	X	X	X	X	X	X	X

Figure 2. Gantt Chart for Semester 1

Task	Week (Semester 2)														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Create MySQL Database and Schema	X														
Complete Frontend Layout and Design	X	X	X												
Research and Implement OAuth2	X	X	X	X	X										
Create Question Generation Agent	X	X													
Create Question Generation Agent Tool		X	X	X	X										
Conduct Testing for Question Agent				X	X										
Create Conversational Agent				X	X										
Create Conversational Agent Tool					X	X	X	X							
Conduct Testing for Conversational Agent							X	X							
Create Student Evaluation Agent							X	X							
Create Student								X	X	X	X				

- ◆ There have been instances where users have been able to manipulate or allow the AI to disregard the original instructions or actions. This can be disregarding the question and allowing full points to users. This could be manipulating the AI into gaining a higher grade for the user.
- ◆ Mitigation: To mitigate this issue, we will refine our prompt engineering to disallow this under all circumstances. We will also use real-world scenarios to test our prompts and compare the difference between providing just the original instruction and providing the original instruction at every user interaction.

→ Cost of Tokens

- ◆ Risk factor: .7
- ◆ There is a concern that users could exponentially increase the cost of each interaction by increasing the amount of tokens or simply, words. This will increase cost and latency of the assessment.
- ◆ Mitigation: To mitigate this issue, we will implement a tokenizer counter using the Python library tiktoken. We will count and limit the number of tokens the user is able to produce per interaction. We will limit the total cost of the assessment to \$.25 which includes both input and output transactions to OpenAI. As of the last update to this document, the current rate of GPT 3.5 Turbo 1106 is \$.0010 for 1K tokens input and \$.0020 for 1K tokens output. We would distribute \$.10 for input and \$.15 for output, bringing 100K and 75K tokens respectfully.

→ FERPA/Student Information

- ◆ Risk factor: .5
- ◆ There is a concern that user data is being incorrectly stored per classification.
- ◆ Mitigation: To mitigate this issue, we have researched the fineprint of the regulations and standards put forth by FERPA and will be taking the necessary steps required to adhere to them. This includes, but is not limited to, encrypting data when available, using HTTPS requests to establish secure connections for transmitting data. We will also follow ISU's minimum security standards and guidance policy.

3.6 PERSONNEL EFFORT REQUIREMENTS

Considering additional time for meetings, brainstorming, iterations, debugging, and unexpected issues, a buffer of about 20% is added. The total estimated man-hours for the project would be around **215 hours**. With a buffer, this goes up to approximately 258 hours excluding preparation and practice for presentations.

Task	Estimated Hours
Requirements Engineering	80
Research Large Language Models	25
Research Data Frameworks	20
Research Canvas LMS API	30
Research Risk and Mitigations	45
Infrastructure Setup	25
Prototyping	120
Prompt Engineering	150

Task	Estimated Hours
Database Setup and Schema	10
Complete Frontend Design and Layout	30
Research and Implementation of OAuth2	60
Creating Agents	40

Creating Prompt Templates for Agents	220
External Canvas API Functionality (PDF Upload, Comment Submission)	10
Create Documentation	20
Total Integration Testing and Refinement	40

Task for Testing	Estimated Hours
Unit Testing	30
Integration Testing	20
System Testing	20
Endpoint Connection Testing	20
Regression Testing	30
Acceptance Testing	50

Milestone	Estimated Hours
Gathering Information	225
Prototyping	120
Prompt Engineering	190
Building Agents	245
Frontend/Backend	182

Testing	170
Total Hours	1,132

Figure 4. Personnel Work Hours Breakdown

3.7 OTHER RESOURCE REQUIREMENTS

The main resources when creating an application using any pretrained Artificial Intelligence is the cost to use the API. While there are free alternatives that we are considering for the project, there is a good chance that we decide on a paid API to use and therefore need to consider it. The most popular cost model for these API's is a 'by token' model which charges the user by how many tokens their prompt takes and how many tokens the API returns. A token is a word or phrase that the API is trained to understand, some words are multiple tokens and some phrases are only one token, it really depends. Being that the API also charges based on the amount of tokens in a return sequence, we are slightly at the mercy of the API, if the API returns a long response, we have to pay, meaning that the API is incentivized to send large responses.

Requirements for this project that aren't financial are mainly related to outside libraries and functionalities that we will be using. One of which is a framework that stores the current conversation with the AI API because the API has no other way of keeping a history of the conversation. This way the API knows what has already been said and can generate an appropriate response for the user based on everything it knows. Another resource is an API that links our project to Canvas. Canvas LMS offers an API that allows users to submit assignments and add comments and check grades and other varying tasks. This will be useful for us so that we can automatically submit the quiz/assignment for the user to make it a more streamlined process. Another potential resource that we are considering using is a backend. A backend could be useful to store current conversations for if we need them. These would likely only be needed if someone's internet checks out and they lose the conversation also so that people can't start a quiz, leave after seeing the question(s) and restart after quitting.

4 Design

4.1 Design Content

Our project utilizes OpenAI's capabilities to achieve the following tasks:

1. Question Generation from Rubric

The AI system interprets an instructor-made rubric to produce a question or prompt to evaluate student competency in the topic.

2. Analyze Student Response

The student engages in a conversation-style assessment with the AI providing their answer. The student will then justify or explain the reasoning and thinking behind their answer. The AI will then proceed to generate more questions, to probe deeper into the student's understanding, until it is satisfied with judging the student's skill.

3. Grading Student Response

The system will then evaluate the student's responses and contrast it to the given rubric to determine competency. It uses a holistic approach taking into consideration not just "what" but the "why" a student answers how they did.

4. Canvas LMS Integration

Both interactions and grades are logged and stored on Canvas. This allows transparency and allows instructors to review interactions when needed.

4.2 DESIGN COMPLEXITY

There are multiple components within our project that maintain a higher level of technical complexity

1. The design consists of components and subsystems that require a high level of prompt engineering in order to meet the technical requirements of the design
 - a. Agents: Agents will be used throughout the design. They have the capability to be optimized for conversational actions. Making decisions on grading and automatically sending rest api requests. Agents rely on prompts to provide instructions to the language model on how to act. Crafting effective prompts requires care to avoid hallucination while still generating high quality responses. Executing chains/LLMs and tools in the right order requires orchestration logic in the agent runtime. The developer needs to write the glue code to invoke the components correctly. Agents produce text which needs to be parsed into structured actions.
 - b. Agent Tools: In order to carry out their expected functions, Agent tools will be used to provide agents guidelines of when to use the tool. The tools need to be properly wrapped and formatted so the agent can understand how to

call them. This involves decorating them and converting them to OpenAI function format. The tools may need access to external resources like APIs and databases. The developer needs to handle authentication, rate limiting, caching, and other technical aspects of integrating with external systems.

- c. Prompt Engineering: Throughout the design there are multiple components of the design that interact with a Lanchain agent or utilize a prompt engineering template that is reliant on an OpenAI large language model.
 - d. Document onloading: A technical complexity within the design will be the onloading of information provided by the rubric and other documents and topics included for the student conversation. This is a critical component of the system as it is imperative for the design to correctly transfer the context of the information from the documents into the memory of the AI so it knows when and where to apply certain concepts and topics outlined by the instructor.
2. The problem scope contains multiple challenging requirements that match or exceed current solutions or industry standards.
 - a. Question Generation for Complex Topics
 - i. There are a wide variety of technical and general topics that are taught. This requires the AI to be well-informed in specific technical details along with comprehension of general subjects. If the AI does not know a subject, it develops the risk of being misinformed, irrelevant, or being too simple. It must develop questions that are meaningful and challenging.
 - b. Adaptative Answer Analysis
 - i. There could be many different answers that the student could choose from to answer a question. The AI must have high-quality training data to ensure it can understand the different answers provided by students. The AI needs to be sophisticated so that it can decipher or analyze the response rather than just the answer to the question.
 - c. Assessment Generation based on a Rubric
 - i. Based on a rubric presented by an instructor, the AI should be able to generate a framework for testing a student. It should be able to generate appropriate questions based on the given topics and thoroughly evaluate the student on each one. It should also be able to handle many different styles an instructor might apply for a

rubric. It should also be able to adjust to any further parameters an instructor adds in as well.

4.3 MODERN ENGINEERING TOOLS

OpenAI GPT: Large Language Model

Python: Designated programming language that has compatibility with OpenAI and APIs.

Canvas LMS API: Facilitation of all recorded student data including grades and responses.

GitHub: Version control for development environment.

Postman: Testing endpoints to ensure correct data is being delivered and received

Langchain: LLM framework responsible for agent components, chaining of responses, prompt templates and agent tools. Langchain's role is to be the framework that connects the user responses with the generated questions. It will also be responsible for evaluating a student's history.

Chainlit: Open-sourced Python Web Framework that packages React and socket.io along with OpenAI. Responsible for the interactive UI chat experience. It will provide the ability to integrate langchain agents tools and agents to add the chatbot experience while providing chat history to the backend.

4.4 Design Context

The societal need for this project addresses some of the shortcomings of traditional quizzes that include the focus on memorization and recall of facts, neglecting other important skills like critical thinking, problem solving, creativity and practical application of knowledge. Grading quizzes can be time-consuming for instructors particularly for open-ended questions or essay exams. Engaging students in conversation not only fosters a more interactive learning environment but also provides instructors with valuable insights into students' comprehension of course content. This is achieved by utilizing AI-generated questions for in-depth exploration of each student's understanding. This project is aimed for educational communities within the departments of a college or university that would like to conversationally benchmark their student's content application and critical thinking skills.

Area	Description	Examples
------	-------------	----------

<p>Public health, safety, and welfare</p>	<p>Our project aims to improve the quality of engagement for both students and instructors. It would help the wellbeing with a flexible form of a quiz that can offer students the chance to complete the conversation with the AI proctor at their own pace and on their own schedule. It can also reduce the stigma against association because some students may feel less anxious about answering questions from a bot than from a traditional quiz. It can benefit instructors by handling routine tasks such as grading and administrative tasks, freeing up instructors and teaching assistants' time to focus on more valuable aspects of teaching and providing support to the students. The data generated from the conversational interaction with the student can help instructors identify areas where students struggle and adapt their teaching methods accordingly.</p>	<ul style="list-style-type: none"> ● AI proctor allows for self-paced assessments. ● Reduction in anxiety levels for students during assessments. ● Instructors spend more time on student interaction rather than administrative tasks. ● Data from AI interactions is used to tailor teaching methods.
<p>Global, cultural, and social impact</p>	<p>This project aligns with the aspirations of educators who are striving for more advanced assessment techniques to gauge their students' capacity to apply the course materials they've absorbed during the semester. By implementing this solution, it has the potential to revolutionize how students engage with course content, shifting their understanding from mere recall and memorization to a more profound focus on application and critical thinking.</p>	<ul style="list-style-type: none"> ● Shift from traditional testing to conversational testing ● Encouragement of higher thinking when taking assessments
<p>Environmental impact</p>	<p>The environmental impact of this project would be one of resource and computer use per student. On a large scale with hundreds of students in a class, the amount of energy required is directly correlated to the processing cost to generate, examine, and categorize students for each quiz. On a grand scale it would contribute to the total energy use for resources associated with large language models.</p>	<ul style="list-style-type: none"> ● Energy usage tied to number of students and complex interactions with AI
<p>Economic impact</p>	<p>The economic impact of the implementation of solutions is the cost of operation to conversationally quiz the</p>	<ul style="list-style-type: none"> ● Cost Analysis for API calls during conversations,

	<p>student would have to be accounted to have multiple conversational assessments per semester. Costs like API usage for the conversation, analysis, and grading of the student as well as costs for hosting the front-end framework for multiple sessions to accommodate students has to be accounted for the expected cost per quiz. The cost associated would have to remain affordable for both the department purchasing the infrastructure and the students who will be paying for the conversational assessments.</p>	<p>analysis and grading</p> <ul style="list-style-type: none"> • Hosting equipment
--	--	---

Figure 5. Design Context Breakdown

4.5 PRIOR WORK/SOLUTIONS

A Similar Product:

★ Quizlet AI

The advantages of Quizlet AI is that it is capable of expanding on the content of the flashcards in the quizlet itself. Expansion of the content allows the AI to ask deeper questions and prompt users for examples or applications of the content in the flashcards. There are also different forms of interacting with the quiz including quizzing, asking deeper questions or just having an informational conversation with the AI to get familiar with the content.

The shortcomings of quizlet AI is that there is a word limit on the response and it can be an obstacle when trying to explain answers to deeper questions. Even though there's a feature to ask for a deeper understanding the character limit prevents it from being a natural conversation. Another limitation of the Quizlet AI is that there is a variability on the chance that the correct questions will be generated. If any content within the flashcards are obscure or new, the AI has a difficult time trying to figure out how to contextualize the information into questions or observing if an answer is correct.

4.6 DESIGN DECISIONS

❖ Large Language Model

- In the language model design decision we had to decide on which language model that we would use to control the generation of quiz questions and the evaluation of the interaction. The factors we used to come to the decision of the LLM that would meet our requirements was the combination of cost and cohesiveness. OpenAI provides a large variety of models that could fit the technical nature of the conversation and can be used interchangeably with other components in the system.

- ❖ Prompt Template and Agent framework
 - Designing a prompt template is a crucial decision that our team took into consideration. We needed to create a structured format for the chat model to generate questions that were relevant and coherent based on a provided rubric. The agent framework will need to ensure the generation of questions are valid and useful and will handle these interactions. These both work hand-and-hand to provide context of multi-turn interactions.
- ❖ Frontend Framework
 - Important to provide a useful interface to interact with both the backend and the chat model. There are factors like real-time communication, user experience, and ease of use. That is why we made the decision to use Chainlit as it provides a simple framework that can facilitate the communication of technical topics while carrying a coherent conversation with the student.

4.7 PROPOSED DESIGN

Our team decided on OpenAI. In regards to prompt engineering, we have been testing and developing a prompt through ChatGPT. There have been instances where the student could obtain an answer from discussing the question with ChatGPT. We are currently assessing the risks and mitigation techniques in regards to this

For the frontend component of our design, we have found a frontend framework that doubles as a web development framework, ChainLit. Through testing we found that we can change the source code of the framework if we want to customize our own UI changes. We also found the capability to integrate langchain components and OpenAI's language model into the framework because of the framework's capacity to create functions around a chat experience.

4.7.1 Design o (Initial Design)

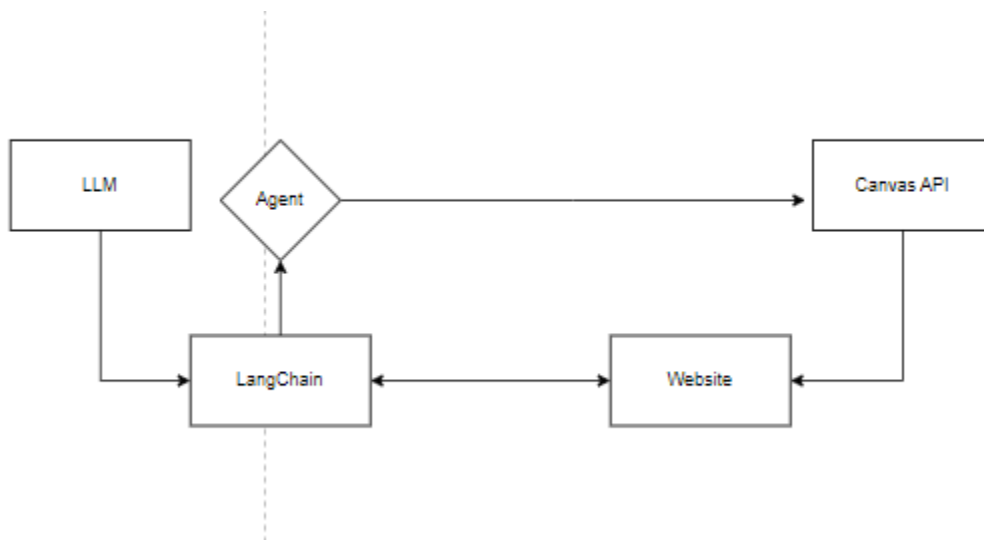


Figure 6. Proposed Design Flow Diagram

```
Below is an instruction that describes a task. You are an instructor for a class. Do not include introductions or conversations. Do not include any explanation.  
  
### Instruction:  
Create a quiz to test a student's understanding of MIPS assembly instructions. Please provide a complete question prompt where it tests the basic skills of using registers by testing the knowledge of only addition and subtraction commands in MIPS . The question should be clear and structured, requesting the hexadecimal values for each register after each instruction.
```

Figure 7. Design o Prompt Template

In this depiction of prompt engineering, the prompt is primed by setting the role, labeling the instructions and lists constraints for the responses. Within the instruction, the prompt declares the format expected for return and the topic that should be covered with additional information on how the questions should be structured and any other specific content.

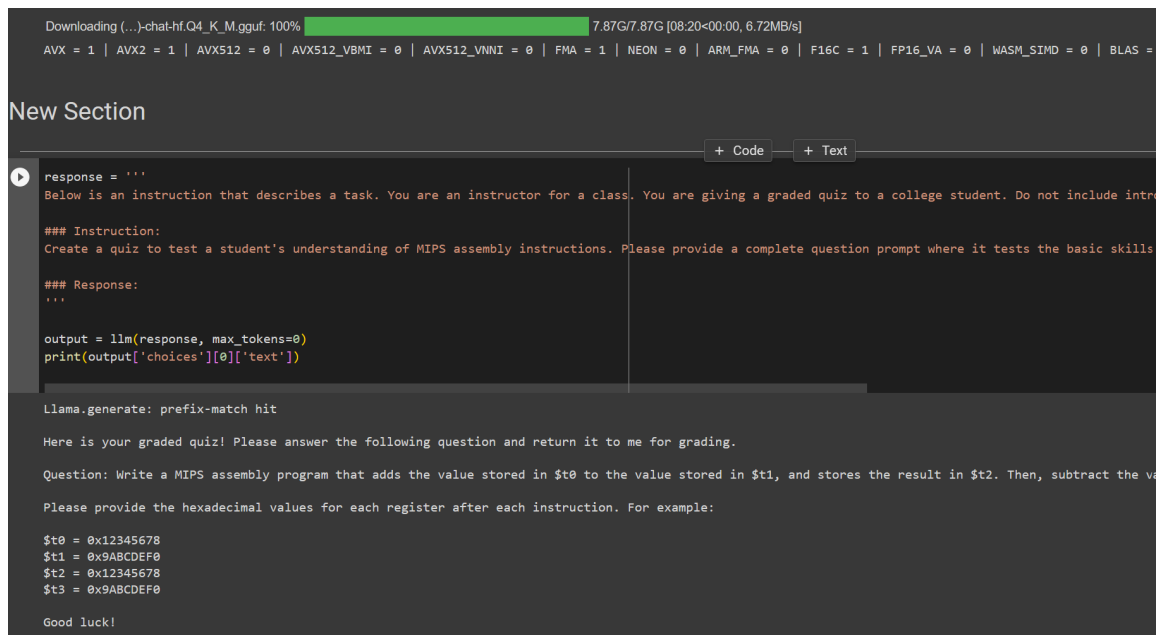
4.7.2 Design Visual and Description

In the initial design there were components that we knew that would be important to meet the requirements for this project. It was critical that the Canvas API was the source of the rubric that contained the values of each criteria outlined by the instructor and as it allowed the instructor a simple way of informing the system of the criteria of the assignment instead of manually uploading the PDF into canvas. The destination of the grades after the assignment was graded was also important as well. Canvas was the

required destination of the grades of the student and sending the grades to canvas would prevent potential FERPA violations since any data of the student would be attributed by the canvasID instead of their personal information. The web interface component is another component required as it is the interface that allows instructors to link the assignments of the class to the student, outline the topics for the assessment and view potential assessment questions generated by the AI. At the time, it was unclear to divide between the instructor interface and the chat component. That's why they were combined into one component. The langchain component was crucial in this design as it was the component that controlled the content generated from the LLM. The Agent was also a component that was needed as it would use its reasoning capabilities to create judgements on the performance of students in the chat. The results of the students interaction would then be sent to canvas to be posted in the grade book under the students canvasID.

4.7.3 Functionality

Currently our design uses open-source large language models consisting of LLama 2 with 7 billion parameters or the fine-tuned versions of Llama 2 like CodeUp, which handles programming languages specifically. This is run on a local server using the Python port of llama.cpp which allows us to run large language models on a local server. Instead of using cloud computing or massive resources normally found in hosting these services, we can use the power of the CPU of our single server.



```
Downloading (...)chat-hf.Q4_K_M.gguf: 100% 7.87G/7.87G [08:20<00:00, 6.72MB/s]
AVX = 1 | AVX2 = 1 | AVX512 = 0 | AVX512_VBMI = 0 | AVX512_VNNI = 0 | FMA = 1 | NEON = 0 | ARM_FMA = 0 | F16C = 1 | FP16_VA = 0 | WASM_SIMD = 0 | BLAS =

New Section

+ Code + Text

response = ''
Below is an instruction that describes a task. You are an instructor for a class. You are giving a graded quiz to a college student. Do not include intr

### Instruction:
Create a quiz to test a student's understanding of MIPS assembly instructions. Please provide a complete question prompt where it tests the basic skills

### Response:
'''

output = llm(response, max_tokens=0)
print(output['choices'][0]['text'])

Llama.generate: prefix-match hit

Here is your graded quiz! Please answer the following question and return it to me for grading.

Question: Write a MIPS assembly program that adds the value stored in $t0 to the value stored in $t1, and stores the result in $t2. Then, subtract the v

Please provide the hexadecimal values for each register after each instruction. For example:

$t0 = 0x12345678
$t1 = 0x9ABCDEF0
$t2 = 0x12345678
$t3 = 0x9ABCDEF0

Good luck!
```

Figure 8. Running localized LLM on Google Colab

In Figure 8, we ran a Large Language Model without incurring API costs. We were able to query instructions for testing with this model and test different prompts. The above

screenshot shows a simple example of how prompt engineering and the LLM ran locally would output.

4.7.4 Design 1 (Design Iteration)

The updates in the matured design lies within the backend components. There was a consensus that OpenAI's GPT models would be more feasible in both speed and quality. There we have added specificity of the agents and their different tasks. The question generation Agent will be provided agent tools that provides it the ability to generate questions given the rubric and content expected to be generated on the quiz. This agent will primarily provide questions for the conversational agent who would then present the questions to the student. The Answer evaluation agent will be provided agent tools that will enable the agent to evaluate the history of a conversation and to determine if the student meets the criteria outlined in the upload of the rubric from the instructor. It is also tasked with sending the rest request back to canvas containing a pdf file of the conversation as well as the grades from the evaluation of the student. The conversational agent is tasked in creating a conversational experience for the student. Not only will it be provided questions that the student must answer but it will also develop follow up questions based on the responses of the student. The flask server utilized by Chainlit is used to communicate with the OpenAI API and the Canvas API.

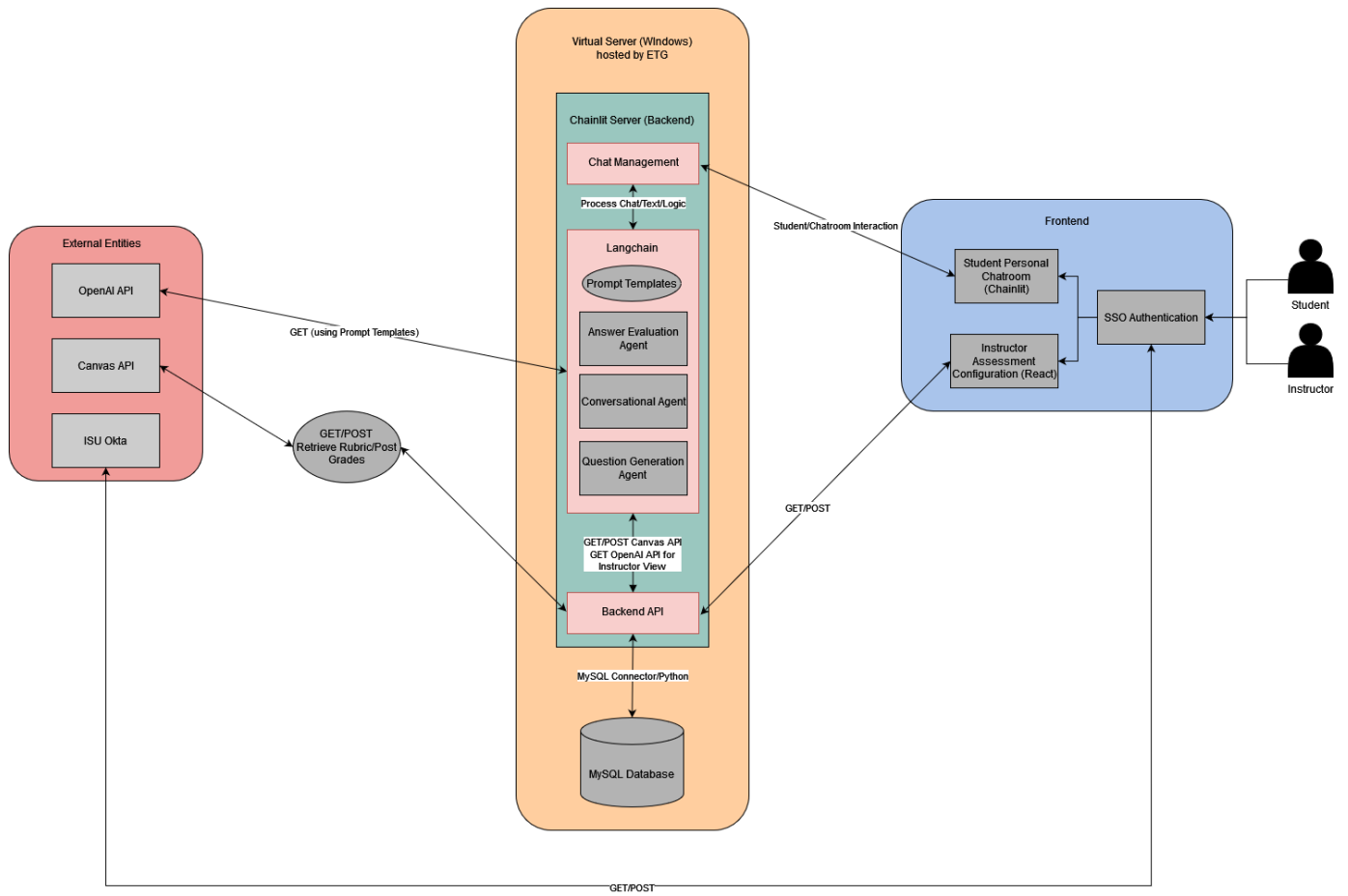
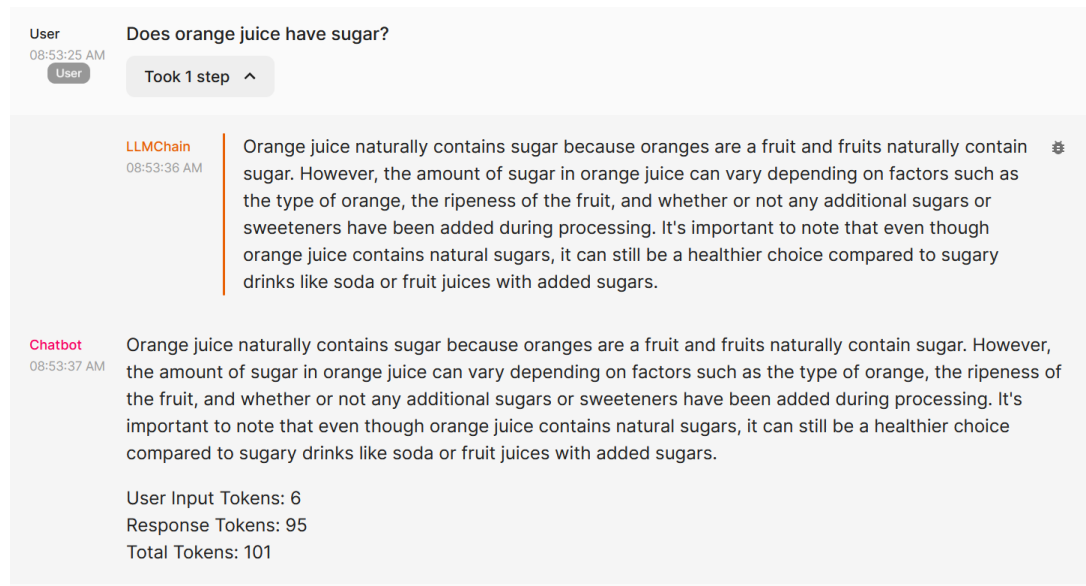


Figure 9. Proposed Design 1 Flow Diagram

4.7.5 Design Visual and Description



The screenshot displays a chat interface with the following elements:

- User:** "Does orange juice have sugar?" (08:53:25 AM)
- LLMChain:** "Orange juice naturally contains sugar because oranges are a fruit and fruits naturally contain sugar. However, the amount of sugar in orange juice can vary depending on factors such as the type of orange, the ripeness of the fruit, and whether or not any additional sugars or sweeteners have been added during processing. It's important to note that even though orange juice contains natural sugars, it can still be a healthier choice compared to sugary drinks like soda or fruit juices with added sugars." (08:53:36 AM)
- Chatbot:** "Orange juice naturally contains sugar because oranges are a fruit and fruits naturally contain sugar. However, the amount of sugar in orange juice can vary depending on factors such as the type of orange, the ripeness of the fruit, and whether or not any additional sugars or sweeteners have been added during processing. It's important to note that even though orange juice contains natural sugars, it can still be a healthier choice compared to sugary drinks like soda or fruit juices with added sugars." (08:53:37 AM)
- Token Usage:**
 - User Input Tokens: 6
 - Response Tokens: 95
 - Total Tokens: 101

Figure 10. Example Interaction using Chainlit Framework

In this depiction of the chat interaction there is depth in the interaction between the question and the response between the user and in the AI. For testing purposes we also wanted to add the ability to view Token usage as there is a difference in cost between the user input tokens and the response tokens from the api. In the application of our design we aim to integrate the conversation depth in addition to the generation of the quiz questions so that we can develop a cohesive conversation with the student so that the history can then be recorded and analyzed to evaluate the answers of the student.

```
Below is an instruction that describes a task. You are an instructor for a class. You are giving a graded quiz to students. You are using recall and analyze learning methods to test these students. They first must recall about the prompt. When responding to their reply, you must have them analyze on why they chose that answer. Do not include introductions or conversations. Do not include any explanations. You are the professor for a Computer Organization and Assembly Level Programming class teaching at the college-level. Do not give any hints. Do not give comments. I will respond with the student response and you will evaluate my response for correctness according to the rubric that is provided. After evaluating on correctness, please have the student analyze on why they chose the answer, in this case, ask me the analyze prompt and I will take the role of the student. You will then judge the correctness and follow the rubric. Output the total score at the end of our conversation. The rubric as follows:  
  
1 point - Has minimal understanding of MIPS programming  
2 points - Has general knowledge of MIPS programming and demonstrates proficiency on MIPS  
3 points - Has mastery knowledge of MIPS programming and can perform high level tasks on MIPS.  
  
### Instruction:  
Create a prompt about basic MIPS implementation that tests the general use or knowledge of MIPS basic instructions.
```

Figure 11. Design 1 Prompt Template

In Figure 11, we were able to mature our MIPS prompt from Design 0. During testing different prompt configurations and keywords, our team noticed we were receiving too much information (AI introduction, “here’s a prompt”) or out right giving an answer or hints. We implemented and reminded the AI to not under any circumstances do this and it seemed to fair well.

Instructor View - Configure New Assessment

Course: DO NOT USE ▾

Assignment: Blank Assignment For Testing ▾

Topics:

Please ensure you are as descriptive as possible when crafting the topics. The more descriptive you are, the better the prompt selection will be.

Preview of Four Generated Prompts:

Prompt 1: What is the purpose of the program counter (PC) register in MIPS programming?

Prompt 2: In MIPS, what is the purpose of the register \$t0?

Prompt 3: How many general-purpose registers are available in a typical MIPS architecture?

Prompt 4: What is the purpose of the branch instruction in MIPS programming?

Text-Format Rubric:

Rubric for MIPS Programming:

Criterion: MIPS Programming

- Details: This criterion measures the student's understanding of basic register use in MIPS programming.

- Rating: Full Marks

- Points: 5.0

- Rating: Minimal

- Points: 3.0

- Rating: No Marks

- Points: 0.0

Criterion: Computer Arithmetic

- Details: This criterion evaluates the student's ability to perform two's complement, addition, and carry in computer arithmetic.

- Rating: Full Marks

- Points: 5.0

- Rating: Minimal

- Points: 3.0

- Rating: No Marks

- Points: 0.0

Figure 12. Instructor View Prototype

Figure 12 shows the prototype of the Instructor View which was added to the new design. This showcases the interoperability of OpenAI API, instead of a locally hosted Large Language Model and Canvas' API. This differs from Design 0 where we ran an abstract backend to a localized LLM.

4.8 Technology Considerations

LLM Selection and Evaluation

Attempted Solution:

- We initially explored the Llama 2 by Meta due to transparency and open-source of the complete project. However, when testing the specific needs of the CPR E 381 course, its performance in coding-related tasks fell short, particularly with the MIPS programming language.

Strengths and Weaknesses:

- Llama 2's 70B parameter model showed promise but lacked the fine-tuned coding ability required for our objectives. Open-source was a competing factor in this case.
- The alternative LLM, Code Up, although refined for programming tasks, similarly did not meet our expectations for MIPS language proficiency.

Chosen Solution:

- After extensive testing, OpenAI's GPT-3.5 emerged as the superior choice, which crafted more precise and relevant results, with a negative being a paid service. The trade-off for its higher responsiveness and quality output was the cost incurred per token usage.

Programming Language Decision:

Python vs. Java:

- Despite our proficiency in Java, Python was chosen for its widespread adoption in the AI/ML community and the availability of OpenAI-specific libraries.
- **Trade-off:** Opting for Python meant embracing its rapid prototyping and strong AI/ML support over Java's performance and static typing benefits.

Data Framework Selection:

LlamaIndex Vs Langchain

In order to develop consistent responses from the LLM a data framework is needed to interact with the API. The data framework should be able to send pre-constructed prompts that outline the requirements and the constraints of the responses from the API. Another requirement for the data framework is the use of agents to make decisions. Throughout the design there are components that need to make decisions on their own in order to perform the correct function. An example of this would be the grading of a student's assessment. Given the student's conversation with the AI, the Agent should be able to review the conversation and accredit the student points based on the rubric provided by the instructor. In making this decision we inspected two data frameworks: LangChain and LlamaIndex. While Llama index provided a simpler encapsulation and abstraction of chatbots that can interact with the API it was lacking in equitable documentation rendering it difficult to apply to projects on a larger scale. Llama index also had restricted abilities with the tools possible for the agents to use. On the contrary, Langchain had developed stronger documentation on the possibilities of its framework

- We favored Langchain over LlamaIndex because of Langchain's ease of customization. Langchain provides the ability to create custom tools for the agents that will be using a reasoning engine powered by the LLM.
- **Strengths:** Langchain also allowed the ability to switch out LLM allowing us to change the intelligence of the responses based on the needs of an agent. Langchain also provides templates which are a collection of easily deployable reference architectures for a wide variety of tasks.
- **Weaknesses:** In comparison to LlamaIndex simple design, Langchain has a complex level of abstraction requiring a deeper understanding in order to get the most out of its tools.

Frontend Framework and Tools

Chainlit:

- Chainlit was the uncontested choice for its ease of setup and its compatibility with our React-based frontend requirements.
- **Strength:** It will provide the front-end web component of the application. It already has a developed chat interface using webhooks powered by flask and has stable integrations for langchain and openAI
- **Weakness:** As it is the only open-source python package we found that could provide the UI interactions of the chat while also providing integrations for langchain and OpenAI api, any additional UI requirements would require the altering of the chainlit source code in order to be reflected in the chat UI.

Database Selection

MySQL vs. MongoDB:

- We favored MySQL over MongoDB for its relational structure and our existing expertise with SQL.
- **Strength:** MySQL supports our structured data and relational data management needs effectively.
- **Trade-off:** In choosing MySQL, we passed on MongoDB's document-oriented model, which might have offered performance advantages with unstructured data and scalability.

4.9 Design Analysis

We did not implement our design in Semester 1. We were able to create prototypes that proved to be successful and were a great proof of concept.

5 Testing

5.1 UNIT TESTING

The components of the system that will be tested would be the questions generated by the question generation agent. The answer evaluation agent will be tested to ensure that the agent is correctly using the rubric to evaluate the students answers. The conversation agent will be tested to ensure that the agent can have a cohesive conversation with the student.

For all agent testing, we plan to implement a testing agent that will be given the output of the question, evaluation, and conversation agent and will be given the description of the agent and asked if the output of the agent meets the requirements outlined in each agent.

Manual Testing for OpenAI's Output: Since the assessment and grading responses are conducted by OpenAI, which is unpredictable, we will need to manually review the quality of the prompts. We will never be able to ensure functionality, but by thoroughly testing the program, we can get close

5.2 INTERFACE TESTING

The two interfaces within our design would be the chat interface with the student and the quiz linking and rubric upload and assessment configuration for the instructor. The composition of the chat interface being tested would be creating unit tests based on the planned interactions within the chat such as chat initialization, question generation, and follow-up question generation. The assessment configuration component will be tested by verifying that the quiz id and the id of the quiz the backend receives are the same. There will also be rubric and assessment configuration tests to assess that all the configurations that the instructor outlines will be reflected and propagated throughout the generation components of the system.

5.3 INTEGRATION TESTING

One of the critical integration paths in our design is the automated interactive assessment generation. It is the section of our design that will allow instructors to specify topics, optional assessment settings, and rubric selection. Testing approach includes developing automated test cases that simulates scenarios of instructors specifying topics, assessment settings, and rubric selections. This test should incorporate a diverse range of topics to ensure robustness. The next step in the process is verifying database integrity to ensure the correct configurations were chosen and saved. The second testing approach we would also need to conduct is manual testing (eyeballing). This involves having instructors or clients manually review the prompts generated by OpenAI for accuracy and appropriateness. They should also verify if the difficulty level and topics pertain to the assessment.

Testing rubric retrieval and accuracy is another critical path of the system as it will ensure that rubrics are being retrieved on the backend and can be broken down by the AI and applied to the students response. It will be tested by using PyUnit to ensure that values credited to the student are values outlined in the provided rubric.

Another critical system path to test is the upload of grades into canvas. Testing the components involved with uploading grades into canvas is critical as it is the primary location of the results of the interactive assessment. Grade upload will be tested by using PyUnit to test the REST responses from the canvas API in comparison to the POST request to upload the grades into Canvas.

5.4 SYSTEM TESTING

Our system level testing strategy lies in the testing of multiple components in the system ensuring that requirements for components are met and generated properly so that they can work in conjunction with other components of the system. This includes unit testing the Chainlit functionality of the chatbot ensuring that users cannot maliciously control the actions of the chat bot. Database functionality is required for the unit as it will ensure CRUD in the mySQL database. API calls are another component to conduct unit testing to ensure proper responses from the Canvas API and OpenAI API. Interface testing will be conducted by ensuring the bidirectional communication between the chat interface on the client end. Within Integration testing, the workflow from creating the assessment to completing the assessment will be tested to verify that all of the system components are meeting its requirements. As it is difficult to test AI-Generated content because of its non deterministic nature, a manual review is needed to validate the questions provided to the student. This would involve instructors to verify the appropriateness of the output. To mitigate against potential risks, there will be tests created to simulate a premature exit of the assessment as well as tests to ensure that the planned mitigation for malicious user responses is working correctly.

1. Unit Testing
 - a. Chainlit Functionality: Testing basic functionality of chatroom
 - b. Database Functionality: Verifying CRUD in the mySQL database
 - c. API Calls: Testing proper responses from Canvas API and OpenAI API..
2. Interface Testing
 - a. Frontend-Backend Interaction: Ensure communication between client interface to chatbot and chatbot response to client interface.
 - b. Backend-Database Interaction: Ensure communication between backend and database.
 - c. API Integration: Check to ensure that APIs are integrated on the backend.
3. Integration Testing
 - a. Workflow Integration: Testing the workflow from creating the assessment to completing the assessment that involves all system components

- b. Database Integration and Canvas Integration: Ensuring that data that needs to be stored and retrieved are correctly stored
 - c. Full API Integration: Testing API interactions including Canvas and OpenAI
- 4. Manual Review for AI-Generated Content
 - a. Manual review is essential since we cannot predict every output. This process involves instructors or subject matter experts to determine the appropriateness of the output.
- 5. Covering Edge Cases
 - a. Simulating Premature exit of the assessment
 - b. Malicious user responses.
 - c. Allow CPR E 381 students to end-to-end test the system for a simulated assessment

5.5 REGRESSION TESTING

Our Regression testing strategy starts with the automating regression testing of the critical components of the design. For the student interface the Chainlit chat interface will be tested to ensure real time communication from the student is correctly sent to the OpenAI API and followed by a follow up question. The database operations will be tested by ensuring that user login information is correctly stored and retrieved from the database. Prematurely closed chat logs and instructor rubrics will also be tested to verify that they are correctly injected into the question generation prompt. We will conduct this testing by using a continuous integration pipeline to run a full suite of automated tests that would include CRUD operations on the mysql database, chatroom functionality and automated tests that simulate responses that would trigger a malicious user flag. Utilizing version control and code reviews will ensure that peer reviews can preemptively catch errors before being pushed. For the AI generated content, prompt testing will be used to compare prompts to ensure that they are of similar difficulty. It will be ensured that prompts do not exceed the complexity outlined by the instructor configuration.

- 1. Automated Regression Testing
 - a. Critical Features
 - i. Chainlit chat interface to ensure real time communication
 - ii. Database operations/CRUD, DB includes:
 - 1. Tables containing student, instructor, and admin information. Information includes assessments assigned to student and conversations that are still needed
 - 2. Assessment configuration including course, rubric, assignment, and other configurations
 - iii. API Interactions
 - b. Tests are based on original system requirements
- 2. Continuous Integration (CI) Pipeline
 - a. Run full suite of automated tests

- b. Specific tests would include CRUD operations on mySQL database, chatroom functionality, API interactions
 - c. Write automated tests that simulate responses that would trigger a malicious user flag
 3. Version Control and Code Review
 - a. Ensure peer reviews to help catch errors before being pushed.
 - b. Using GitLab
 4. Manual testing of AI-generated prompts
 - a. Compare prompts to ensure that they are of similar difficulty
 - b. Make sure prompts don't exceed the information provided when making assessments
 5. Updating Test Cases to reflect progress

5.6 ACCEPTANCE TESTING

We will demonstrate that the design requirements are being met by the inclusion of edge cases in our acceptance testing. By conducting user testing with the intention of finding potential risks in potential responses from users and finding ways to mitigate them. User testing will also allow us to qualify the responses from the generated questions and ensure that the questions are restrained to the scope of the content and the assessment configurations. Reliability and Usability are both requirements that will be the focal point of the testing because it will enable us to spot any potential pitfalls in the generated questions and responses. Specifically with the consistency of the generated question and the responses and the prevention of students being able to ask . As well as the ability for users to have a coherent conversation with the chatbot to ensure that the student is able to respond to the questions generated by the AI.

The **instructor view** is also a component of the system that will require the involvement of the client as we want to validate that the instructor is able to comprehend the conversation between the student and the AI and confirm that the questions generated by the AI fall within the scope outlined by the assessment configuration, this entails questions covering what's outlined by the course material and professor specifications as well as being a difficulty that matches what the professor wants.

5.7 SECURITY TESTING (IF APPLICABLE)

It will be important for us to ensure that each student is entitled to their own attempt and that their conversation with the bot is safe from other people viewing it without access. We will be using OAuth 2.0 to authenticate users. We will automate the tests by using open-source Python package requests-oauthlib which allows us to process requests to OAuth providers. We can manually test the end-to-end endpoints to verify if users are successfully authenticated. We can also use automated testing using pre-generated

tokens.

5.8 RESULTS

The testing that we have conducted is limited to integration of some of our modules and system testing. By having an early prototype to conduct tests on, we have been able to integrate some of our module together, for example getting the Canvas LMS to both work with our code and to integrate it with OpenAI to get more precise responses from OpenAI. We've also conducted some interface testing. We're not complete with our interfaces but we have created a basic interface which has the general design and concept that will be eventually used. Due to the fact that we are early in the implementation process, we have yet to start or complete any tests of automation or regression.

6 Implementation

The preliminary implementation that we have completed includes a prototype of the chat interface that includes basic prompt engineering using the question generation agent using Chainlit and a connection to the OpenAI LLM. We also have a prototyped instructor view that connects to the Canvas API and can pull a text-based rubric.

7 Professionalism

This discussion is with respect to the paper titled "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment", *International Journal of Engineering Education* Vol. 28, No. 2, pp. 416-424, 2012

7.1 AREAS OF RESPONSIBILITY

Our group consists of Software Engineering majors so it is only natural we follow the SE Code of Ethics.

Area of Responsibility	Personal Definition	Addressed by SE COE	Comparison to NSPE
Work Competence	The ability to effectively carry out work efficiently and honestly.	Principle 3.04. Ensure that they are qualified for any project on which they work or propose to work by an appropriate combination of education and training, and experience.	NSPE only covers areas of competence that you are performing services. SE COE ensures qualification through education and experience.

Financial Responsibility	The ability to handle financial resources responsibility and realistically.	Principle 3.09. Ensure realistic quantitative estimates of cost, scheduling, personnel, quality and outcomes on any project on which they work or propose to work and provide an uncertainty assessment of these estimates.	NSPE directs individuals to be a trusted agent while SE COE implies having realistic goals for cost and project.
Communication Honesty	The ability to be truthful and honest during communication.	Principle 1.06. Be fair and avoid deception in all statements, particularly public ones, concerning software or related documents, methods and tools.	Both codes command honesty but SE COE focuses more on software deception issues.
Health, Safety, Well-Being	The ability to ensure the safety of the health and well-being of all individuals.	Principle 1.03. Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment. The ultimate effect of the work should be to the public good.	NSPE covers the whole public while SE COE relates to the safety and privacy of software.
Property Ownership	The ability to respect intellectual and physical property rights.	Principle 2.06. Identify, document, collect evidence and report to the client or the employer promptly if, in their opinion, a project is likely to fail, to prove too expensive, to violate intellectual property law, or otherwise to be problematic.	NSPE directs individuals to be a trusted agent while SE COE explicitly states to document and report abuse of intellectual property law.
Sustainability	The ability to produce sustainable products and consider environmental impacts.	Principle 3.03. Identify, define and address ethical, economic, cultural, legal and environmental issues related to work projects.	NSPE does not mention sustainability. SE COE directly references economic, social, and legal aspects of environmental issues.

Social Responsibility	The ability to responsibly act for society as a whole.	Principle 1.02. Moderate the interests of the software engineer, the employer, the client and the users with the public good.	NSPE directs individuals to conduct themselves honorably and promote services that benefit society. SE COE focuses on the balance of stakeholders interest with the public good in software development.
------------------------------	--	---	--

Figure 13. Area of Responsibility Interpretation Breakdown

7.2 PROJECT SPECIFIC PROFESSIONAL RESPONSIBILITY AREAS

Work Competence: This area definitely applies to our project’s professional context. A team’s relative skill in the area of a project will always be important for carrying out a task. It is likely recommended to have skilled individuals who can contribute to the project but it is also important to expand the expertise of the individuals otherwise the project is likely too simple. It’s important for both the stakeholders and the users that they can trust the team to make a working product that is conducive to the initial specifications and made in an efficient manner. We rank as medium for this task. Due to circumstances such as the novelty of artificial intelligence, and our inexperience with this, combined with the fact that we are still college kids that are learning, we don’t rank as high. On the other hand, we do bring a lot of diverse knowledge into the project as well as some experience and knowledge with artificial intelligence, which puts us at medium.

Financial Responsibility: Many software projects suffer the same fate of going well over budget, and while this is especially pertinent to the realm of artificial intelligence assisting with human tasks, one could argue that the risk is relatively low compared to the potential of this new invention. For our group specifically, funding has been simple and straightforward. OpenAI uses a rate based on tokens to charge users. Financially is probably where we differ the most from an actual software project as we’re not having to pay for labor, constantly have stakeholder meetings and pitches for funding, but we still have to consider things such as how many tokens we’re using. We rank medium for this again not because of anything wrong we’ve done but because of the lack of financial responsibilities that we have.

Communication Honesty: Due to the complex relationship that technology has always had with the general public, this was always going to be very important. While it’s still young in the technology game, it’s likely that AI will come with all of the distrust and hatred that all of its technological predecessors had before. The most relevant topic that has been

popping in and out of the mainstream has to do with how 'open' these open source projects truly are. Whether companies are responsible for sharing what their neural network was trained on is a controversial topic and many see it as a way of not being honest in communication. This is massive for our professional context and especially so because we will be working with instructors. We have not disclosed any information about our project to the public so we rank as N/A for this topic for the time being. We have discussed many codes and regulations that we will likely have to adhere to when actually implementing this project so it is still very much important to our context.

Health, Safety, and well-being: This area is not nearly as important to our area as the last few have. The biggest aspect in this project is to do with privacy. Because softwares that helps to assist learning is typically dealing with sensitive student information, it is very important to be extra cautious so that nothing of theirs gets leaked. While softwares like ours aren't actually protected under the FERPA regulations, it is important to us that we treat the information we receive from the student as if we were, this includes not saving information on our backend when we don't need to and using secure methods of transferring information. Although overall, this isn't a large area in the context of our project, it's still important. We haven't implemented much of the code that will eventually handle student and instructor information but we have had many good discussions about what are plan is and how we can confidently protect information. So we rank high in this area.

Property Ownership: This area once again is not the most important area to our professional context. Intellectual property will always be important to preserve and to make sure we're not infringing on anything that we don't have rights to. It's no more important to us than any other project. Because we're using existing software solutions such as OpenAI, ChainLit, and LangChain, this does have some impact. But because the rules and regulations for using these softwares is clear and spelled out for us, this is less of an issue for us than most projects. Our team ranks as N/A for this area because we are not likely to infringe on anyone's intellectual properties and we definitely haven't yet.

Sustainability: While some software projects have to consider the environmental impact of their work, such as cryptocurrencies, our professional context has very little to do with sustainability. We rank N/A for this, we're not producing anything or have to worry about any natural resources to create and distribute our product.

Social Responsibility: This has been and will likely forever be the most important are for a software project. This will be discussed in more detail in section 7.3 but it is both

incredibly important and incredibly hard to eliminate bias from software ‘algorithms’. This is especially important for our professional context because we’re making a software that will subjectively grade students. It will be important to try and minimize the bias towards different styles of writing, how many details students use, and other small things. But this task is almost impossible because the neural network was trained on something, and like humans, some biases are really hard to notice or remove. That being said. We have really considered this area in our software project and we have plans to mitigate a software bias showing up. In the context of our project, we rank as high for how we’re performing this. There were always going to be some discrepancies but so is there with human graders, we’re just aiming to be at the same level or better.

7.3 MOST APPLICABLE PROFESSIONAL RESPONSIBILITY AREA

The most important personal responsibility area as mentioned in section 7.2 is the social responsibility area. Due to increasing interest in topics such as social media ‘algorithms’ and what each artificial intelligence network has been trained on, it’s easy to be wary of technology now. Some people worry that as technology takes over more and more of what we do, biases of the past will resurface because the models are still being trained on old data. Well what does this mean for us? Because we are using artificial intelligence to both grade and guide an assessment, it’s important that no student is graded unfairly because they write in an unconventional way or use shorter/longer sentences than another student. It’s really hard to identify these biases because it is likely that we share some of them with the AI. Aside from biases, a learning platform will affect the way people learn and the way they study. Many students will treat assignments differently based on what platform they are on. Some applications are known for giving answers to students if they need which makes it easier to not pay attention. Other applications give the student unlimited attempts which also affects how they learn. Others create unwanted stress in the student by constantly having to worry about whether they are cheating. These issues aren’t just limited to software solutions, learning and evaluation of learning all come with their own issues. While we know our application will inevitably come with issues of its own, we aim to try and create a positive environment for the users. It’s important to uphold the decency of educational institutions by trying our best to create an assessment environment that isn’t stressful for the student but also thoroughly assesses what the student knows about the given topic.

8 Closing Material

8.1 DISCUSSION

The result of our project appears to be a feasible software-based assessment environment that is expected to meet all of the requirements outlined. Based on just the prototype we have developed, we have integrated the Canvas API, allowing for back-and-forth communication. We have developed a model to automate the assessment grading process, the cost of each exam being three dollars per assessment per student will be hard-capped, and enforced based on the token counting system, there will be a pdf document of the chat history for the instructor to review should there be discourse between the grade given by the software and what the student thinks they deserve. The instructor's view also meets our requirements, where it is easy for instructors to view, use, and create assessments. Following all that, we have complied with FERPA as implemented under Iowa State University policies.

8.2 CONCLUSION

The overall goal of our project is to design and develop an interactive artificial intelligence-based assessment environment. This would provide instructors with a way to streamline the assessment process with an environment that automatically assesses, grades, and submits assessments to Canvas.

Our plan of action is to use all of the resources we have described, including but not limited to; ChainLit, LangChain, OpenAI's GPT LLM, and Canvas API, to further develop software based on our prototypes. We will start with the prototypes that we have already worked on and improve upon them by connecting all of the components using different kinds of REST and API requests, fine-tuning the question evaluation agent and assessment grading agent to provide accurate feedback and responses using specific prompt-engineering and rubric evaluation, and eventually using real-life (volunteer) students to test our project in a semi-controlled environment.

We will start by connecting the prototypes that we have created and creating a fully functioning machine where an instructor can create an assessment, assign students to it, have the students take the assessment, and send the grade and record of the assessment to Canvas. After this, we will move into fine-tuning the different agents being used in the design, specifically the Question Generation agent, the Answer Evaluation Agent, and the Assessment Grading agent. These will need to be adjusted using prompt engineering to ensure that all of the requirements of our project are being met and the project is secure.

We will then start to use real student testing, as well as the various other types of testing we have implemented to further adjust our project to improve its capabilities.

8.3 REFERENCES

[1]“Get started | Langchain,” *python.langchain.com*.
https://python.langchain.com/docs/get_started

[2]“Overview,” *Chainlit*. <https://docs.chainlit.io/get-started/overview> (accessed Nov. 28, 2023).

8.4 APPENDICES

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar data that does not directly pertain to the problem but helps support it, include it here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc., PCB testing issues etc., Software bugs etc.

8.4.1 TEAM CONTRACT

Team Members:

- 1) Akpobari Godpower_____ 2) Alex Vongphandy_____
- 3)Abram Demo_____ 4) Drake Rippey_____
- 5) _____ 6) _____
- 7) _____ 8) _____

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
 - a. TA meetings: Mondays (Weekly) 3:30pm
 - b. Client/Submitter Meeting: Fridays (Weekly) 1:10pm
2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
 - a. Communication amongst team members: Discord
 - b. Issues and scheduling: Git
 - c. Weekly meeting with client: In-Person
 - d. Weekly meeting with TA: Webex
3. Decision-making policy (e.g., consensus, majority vote): Consensus, we will present pros and cons of each idea until we all come to an agreement
4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived): Alex will keep meeting minutes and an alternate person will be selected during the meeting if he is absent. Meeting minutes will be logged on an agreed Google Doc.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings: Full attendance at every meeting, arriving on time to each meeting. Full participation is required at every meeting. An understanding of what will be discussed at each meeting and what each team member needs to present is also expected

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

Responsibilities are delegated every week after every weekly meeting with client. Timeline and deadlines are to be adhered to unless the team is notified that timeline or deadline will not be reached and an explanation

3. Expected level of communication with other team members: Communication on work completion, questions and for any emergency situations. Communicate often with team members. We expect team members to respond to other team members questions/emergencies within 24 hours of original comment, preferably sooner.

4. Expected level of commitment to team decisions and tasks: Full commitment to team decisions and tasks. If a team member disagrees with a team decision or cannot fulfill a task, they will need to explain the reasoning through communication channels. The team will have a consensus on remediation.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction,

individual component design, testing, etc.):

- a. Team Leader/Team organization - Akpobari G
- b. Testing - Drake R
- c. Server Management - Alex Vongphandy
- d. Frontend Development Design - Abram Demo

2. Strategies for supporting and guiding the work of all team members:

3. Strategies for recognizing the contributions of all team members:

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.

- a. Akpobari Godpower
- b. Abram Demo - Front end developer for summer internship, Java programming and other languages, Experience using Git and other Agile management tools
- c. Alex Vongphandy - Server/Network Management, Java Programming
- d. Drake Rippey - Basic Java Programming combined with experience in back-end development and lots of customer service experience.

2. Strategies for encouraging and support contributions and ideas from all team members: During meetings if someone has encountered a blocker and needs help, time will be taken to understand the problem and develop possible solutions towards the problem.

3. Procedures for identifying and resolving collaboration or inclusion issues: Contribution recognition will be reflected in the documentation and will be discussed during meetings so that other team members are aware if they have any questions for the contributor for team understanding.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

- Proof of concept design that meets requirements of the client.
- Find solutions around potential constraints and present them to the client.
- Make design decisions based on research and testing to ensure projects are on

2. Strategies for planning and assigning individual and team work: We will try to create an open environment where everyone's opinions are able to be presented. We will encourage contributions from team members by reaching out for help early in the development process and by helping each other out especially if we may have more expertise in an area.

3. Strategies for keeping on task: By having weekly meetings where we will discuss what we have achieved in the prior week as well as discussing our future plans, we will constantly have a workload

to keep us busy throughout the semester. We will try and prevent stalled work by timeboxing our processes and by reaching out for help often in the development process and by helping out other team members when we are able. Documentation to prevent information loss. A Kanban board to keep track of tasks that need to be completed, completed tasks and tasks placed in backlog

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

Any infractions of the obligations will first be discussed amongst team members to see if there is an alternative solution to prevent further infractions.

2. What will your team do if the infractions continue?

Contacting first the TA, than the instructor if need be to let the correct people know that they are not doing work and should not be given credit for something they are not contributing to.

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1) _____ Drake Rippey _____ DATE __09/08/23_____

2) _____ Alex Vongphandy _____ DATE __09/08/23_____

3) _____ Akpobari Godpower _____ DATE __09/08/23_____

4) _____ Abram Demo _____ DATE __09/08/23_____