**SDD Testing Document**

**1. Survey Feedback**

*Objective:* To gather user feedback on the application's functionality, usability, and overall user experience.

**Survey Questions:**

1. On a scale of 1-5, how would you rate the usability of the application?
   * The most common response was people choosing to rate this application a 3 out of 5. As it does work as intended, however, it is not visually appealing as well as being slow to generate questions.
2. Were there any functionalities that did not work as expected?
   * Sometimes the program ran into a random error.
   * The program sometimes generated questions that did not make sense.
3. Describe any difficulties you faced while using the application.
   * It was hard to read what was going on when the application began to spit out information as the logs from the previous questions got in the way, disrupting the clarity of the application.
   * The application began to be difficult to use because it was not clearly laid out and the font size was too small.
   * Sometimes the application would have an error randomly and you would have to restart the application.
4. What improvements would you suggest?
   * Making it more visually appealing and easier to read for everyone.
   * Make the application work faster.
   * Have a workaround for the errors, maybe start the application on its own but enter the same prompt as before.
   * Fewer bugs
5. Any other comments or feedback?
   * It would be really cool for teachers to be able to utilise this tool in larger classrooms where teachers aren’t always able to provide explanations and give out questions.
   * This application with a little more design features and creativity could work really well in primary schools to keep students engaged.
   * Would it be possible to create this for any other subjects? Maybe a series.

**Summary of Survey Feedback:**

* **Positive Feedback:**
  + Functionality: The application works as intended, with users generally finding it functional for the intended purpose.
  + Educational Potential: The application is seen to have potential in the classroom, especially for larger classrooms where personalized attention from teachers may be limited.
  + Flexibility: Users are interested in the possibility of the application being expanded to cover different subjects.
* **Areas of Improvement:**
  + Usability and Design: The most common suggestion is to improve the visual appeal and readability of the application, as users find it currently lacking in these areas.
  + Performance: Users experience the application as slow and would prefer a faster response time when generating questions.
  + Error Handling: There is a need for better error management, with suggestions for the application to auto-restart or remember the last prompt after an error occurs.
  + Bug Fixes: Users would like to see fewer bugs and issues during the operation of the application.
  + Functionality: Some reported problems with random errors and nonsensical question generation, indicating a need for refinement in how the application processes inputs.

**2. Desk Check**

*Objective:* An internal review of the application's functionality by manually working through its features.

**Feature:** Generate and Present Maths Questions

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | **Expected Output** | **Actual Output** | **Pass/Fail** |
| Algebra (for the topic) | A multiple-choice question based on algebra. | Expected Output | Pass |
| Help | A helpful hint for the current question | Expected Output | Pass |
| User provides incorrect answer | That's not correct. Try again, or type 'help' for a hint. | Expected Output | Pass |
| User provides correct answer | Correct! Well done. | Expected Output | Pass |
| “Exit” | “Goodbye! Have a great day.” Then program terminates | Expected Output | Pass |

**Summary of Desk Check Findings:**

* **Total Features Checked:** 5
* **Pass:**
  + Algebra (for the topic): The application correctly generated a multiple-choice question based on algebra.
  + Help: The application provided a helpful hint for the current question.
  + User provides incorrect answer: The application responded correctly by prompting the user to try again or ask for a hint.
  + User provides correct answer: The application acknowledged the correct answer with a congratulatory message.
  + "Exit": The application appropriately bid farewell to the user and terminated the program as expected.
* **Fail:**
  + None

All features that were desk-checked have passed according to the expected outcomes outlined in the provided format. This indicates that for the inputs tested, the application behaved as intended and met the functionality requirements.

**3. Reflection and Evaluation**

*Objective:* To provide a holistic view of the testing phase, lessons learned, and future recommendations.

**Reflection:**

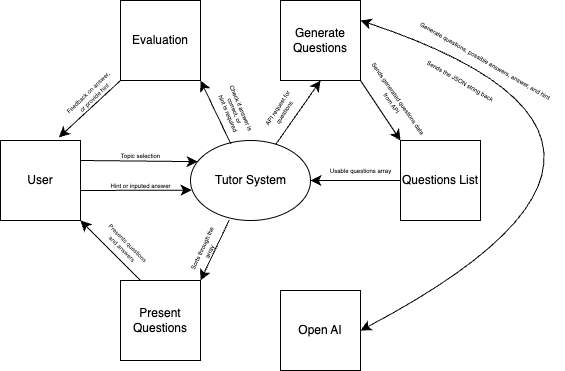
* **What Went Well:**
  + Actually getting ChatGPT API to provide me with proper questions related to the requested topic.
  + Getting ChatGPT to create a JSON line that was an array containing questions, possible answers, as well as a correct answer and an explanation as to why the correct answer is correct.
  + Creating a help button that prompts the user to the next step in solving the maths problem but not telling them the answer, sometimes.
* **Challenges Encountered:**
  + Figuring out how to get the hint working, because I thought I would have to engage with the API again, however, I found a workaround and just made it create a hint when generating the question.
  + Another difficult challenge I came across was getting the possible answers to be displayed in an effective way, while also changing questions at the same time.

**Evaluation:**

* **Effectiveness of Testing:**
  + The testing phase was a crucial part of this project as it helped me narrow down and focus on what needed to be fixed, such as the error workarounds to keep the program running, as well as the bugs that the API made when generating questions. However, in this section, I did have to use the help of ChatGPT to solve this issue as my solution kept resulting in errors. So with the help of ChatGPT and YouTube videos, I got the error issue resolved. Additionally, the testing phase helped me realise what other potential features I could add such as the hint option.
* **Recommendations for Future Testing:**
  + If I were to recreate this program again, I would do it in JavaScript and have it on an online platform that users can access from their browsers. Additionally, this would help fix the design problems that users were talking about in the survey feedback as I could try and make it more visually appealing and accessible when answering questions, displaying questions, and providing effective help. I believe this type of program could have further potential in other areas too which is something that the testing phase had made me passionate about.

**Data flow diagram**

<https://drive.google.com/file/d/1-qJ5nfbVpJ6JCkEuPgZybtQNv7R_Z3vz/view?usp=sharing> (if you need better quality just ask me and I can show you on my laptop)



**Social and ethical issues.**

**Accessibility and Equity**

Ensuring equitable access to education technology that utilises AI systems is one of the most significant issues. Although AI can potentially customise and tailor learning to individual students' needs, learning styles, and abilities, there is a risk that it could extend other inequalities even further. For example, students in underfunded regions or schools that have limited access to technology or funds to purchase AI education systems might not have the same access to other more developed regions. This disparity can lead to a widening educational gap across the world, where some students benefit from this new personalised learning, while others are left to the traditional ways which may disbenefit them in the future. Additionally, there could be a concern raised where students depend on AI education so much that they could completely wipe out the need for school teachers and even schools themselves. If students can learn all of this at home with the assistance from AI this could lead to impacting the emotional and social development of students.

**Data Privacy and Security**

Large amounts of data are required for AI systems to function effectively, this will include student information like learning patterns, personal preferences, and potentially biometric data. This brings forth quotes about how this data is stored, who has access to it, and how student's data is securely protected from data breaches. Additionally, the ethical use of this data is also a concern. For example, if a student's data is used to make decisions about their educational paths, there can be a misuse of data through commercial selling of data or surveillance from other nations. These can all lead to an infringement on students’ privacy rights.

**Bias and Fairness**

AI systems are still not free from the biases of their creators. The creators' biases could potentially leak into their education software, leading to unintended discrimination. For example, if an AI system is trained on one demographic of students, it has a chance of not performing as well or being as effective for students outside of that demographic group. This could lead to unequal education experiences and outcomes, as the pre-held bias is instilled into these students' education. Therefore, ensuring that AI education systems are as unbiased and fair as possible is a significant challenge and issue that must be addressed.

**Program - Documentation**

**Generate\_questions function:**

* + This function interacts with the ChatGPT API to generate a list of maths questions based on the given topic inputted by the user. It sends a prewritten response to the API and gets back a list of questions, multiple-choice options, a correct answer, and a hint. Finally, this function checks if the response from the API is in the requested JSON format.

**Present\_question function:**

* + Once the list of questions is generated and checked by the generate\_questions function, this function aims to present each of the questions and the possible answers to the user. The function then creates a loop that lets the user ask for a hint. If they do not ask for a hint and guess an answer, the function then evaluates the user’s input. It will then provide feedback, once the question is correct, it will move on to the next question until there are no more questions left.

**Main function:**

* + This function is where the two previous functions get called upon. The function is in a while true loop that will repeat unless the user types exit. In this loop, the user will be asked to input the subject they want to work on. Once selected the generate\_questions function is called, followed by the present\_questions function to generate and show the questions. In this function, there is a failsafe where if no questions can be generated, or an error occurs, the user will be informed and asked to try again.

**Program – Actual Code**

Fair warning that you may have to retry the topics again and again until it works. The API isn’t the greatest in generating the same thing every time.

Some ones that consistently work are: “simple addition”, “simple multiplication”, and “fractions”

import openai

import json

#set the api

api\_key = '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'

#chatgpt message variable

chatgpt = []

#start the openai api client

openai.api\_key = api\_key

def generate\_questions(maths\_topic):

response = openai.ChatCompletion.create(

model="gpt-3.5-turbo",

messages=[

{"role": "system", "content": f"Create a list of 3 multiple-choice math questions on the topic of {maths\_topic}. For each question, provide a question text, a list of four options labeled a), b), c), and d), a correct\_answer that is one of these letters ('a', 'b', 'c', 'd'), and a hint to help solve the question if needed. Return the questions as a JSON array of objects."},

{"role": "user", "content": maths\_topic}

],

)

content = response['choices'][0]['message']['content'].strip()

try:

questions\_list = json.loads(content)

return questions\_list

except json.JSONDecodeError:

print("Invalid JSON content:", content)

return None

def present\_questions(questions\_list):

for question\_info in questions\_list:

# check if 'question' key exists

if 'question' not in question\_info:

print("Tutor: Error - The key 'question' does not exist in the question info.")

print(f"Tutor: The available keys are: {list(question\_info.keys())}") # Print the available keys for debugging.

continue # skip this question and move to the next

# assuming the keys exist, print the question

print(f"Tutor: {question\_info['question']}")

# check if 'options' key exists

if 'options' in question\_info and isinstance(question\_info['options'], dict):

for key, value in sorted(question\_info['options'].items()):

print(f"Tutor: {key}) {value}")

else:

print("Tutor: Error - The key 'options' is missing or is not a dictionary.")

continue

# start loop for the user to answer or ask for help

while True:

user\_input = input("You: ").strip().lower()

# If the user asks for help

if user\_input == 'help':

hint = question\_info.get('hint', 'There is no hint for this question.')

print(f"Tutor: {hint}")

continue

# check if the users answer is correct

correct\_answer = question\_info['correct\_answer'].lower()

if user\_input == correct\_answer:

print("Tutor: Correct! Well done.")

break

else:

print("Tutor: That's not correct. Try again, or type 'help' for a hint.")

def main():

while True:

print("Tutor: What subject would you like to work on today? (Type 'exit' to end)")

user\_subject = input("You: ").strip().lower()

if user\_subject == 'exit':

print("Tutor: Goodbye! Have a great day.")

break

questions\_list = generate\_questions(user\_subject)

if questions\_list:

present\_questions(questions\_list)

else:

print("Tutor: I'm sorry, I couldn't generate questions for that subject. Please try again or choose a different topic.")

main()