**Link for video and code: https://drive.google.com/drive/folders/1KdTRiMr62x-y1eW8ViaqUTZthTwg9zrq?usp=drive\_link**

**What is working:**

Our platform incorporates several impactful features designed to enhance the learning experience for LIGN 101 students. Firstly, students have the flexibility to choose their workload by inputting the number of questions they wish to review. We ensure that the learning process is neither overwhelming nor underwhelming.

Secondly, the core effectiveness of our platform lies in its ability to personalize the difficulty level of questions. The difficulty level is dynamically adjusted, providing students with personalized questions. If a student performs well on previous questions, they will encounter more challenging ones. Conversely, if a student answers a question incorrectly, they will receive a lower difficulty level question.

Thirdly, student answers can be evaluated by a specific Assistant API with access to class materials such as slides, lecture transcripts, and problem sets. This Assistant API not only generates relevant questions but also provides precise evaluations of student responses. This capability ensures that feedback and learning materials are always relevant and up-to-date. Additionally, we strive to offer encouragement to students, whether or not they perform well.

Lastly, at the end of each practice session (if the student chooses not to continue with more questions), they will receive a summary that includes the questions they answered that day, along with a review of questions marked incorrect including those that the student has no idea how to answer, and the concepts they need to revisit.

In conclusion, our platform combines personalized learning, intelligent feedback, and flexible learning options for LIGN 101 students.

**What is not working:**

While our platform demonstrates some progress in enhancing the learning experience for LIGN 101 students, there are still areas where it falls short or fails to meet our objectives.

Firstly, the Assistant API we utilized does not permit users to modify resource files. This limitation means that our application can only support the materials initially uploaded, preventing instructors from automatically making further changes to update the questions for course content. Consequently, our application restricts the range of content available for students to learn or practice.

Additionally, users are unable to choose to revise or practice a specific section of the course content. This limitation leads to a challenge for students seeking to focus on particular concepts or materials they are deficient in, hindering their ability to learn and improve.

Moreover, students are not able to keep a copy of the summary section, which means that they cannot easily go over the questions after they have closed a session. Similarly, their working progress (i.e., level of difficulty) can only be updated within each session. Every time students run the function, the level of difficulty of problems will start from level-3, which is not friendly to those who do not want to waste time on easier questions. In fact, we are somehow doubtful on how well openAI can evaluate the levels of the corresponding questions. In practice, although there are some differences among questions of different levels, there is no clear distinction among those of similar levels. We believe that this problem does not only rise in our project but may apply to a much broader range of applications. Therefore, to design better algorithms of evaluation, whether through openAI or by ourselves, is critical to a more advanced and accurate version of our assessment system.

To address these issues, we can make further potential modifications:

1) Create multiple Assistant APIs, each containing different materials from specific chapters of the course content. By segmenting the course resources, we can invoke the relevant Assistant API with its imported files to generate corresponding questions based on the user's input.

2) Utilize other Python packages capable of extracting text from PDFs, Powerpoints, or other document formats that users may import. Subsequently, upload the extracted text to the API. This approach significantly expands the scope of course materials available for generating questions, excluding the limitation of file types to be imported in the API. This modification also allows students to choose and focus on the materials they need to improve.

**What we learned:**

The most important takeaway from this project is how to use the GPT API. GPT’s feature of "Assistant API", which has features including instructions, files, and functions, is like a new approach to programming, if we consider GPT as a programming language with highly flexible syntax and more functions. Despite the non-reproducibility and randomness of results, we fine-tuned the model to make it more predictable. For instance, by specifying the desired question format in the instruction and instructing GPT to incorporate specific phrases in its responses, we have managed to have greater control over the generated content. This opens up a new method for dynamic and customized interactions with the GPT model.

Furthermore, although we do not have a UI at this time of the project, we nevertheless have learned more about UI design by reading instructions and watching videos of related tutorials. We attempted to build one via multiple platforms, including PyQt, Django, and Flask. We hope that as we enhance our knowledge through future practice, we may ultimately build our more user-friendly web application for real-life usage. We noticed that the areas of programming and AI are worth exploring, and we will not hesitate to explore them deeper in future career and study.