

# AI Powered Quiz Generator

## **Team Members:**

Alex Franco, Alex Huang, Wissam Kazzazi,  
Sean Tran, Izak Vucharatavintara, Juan Pablo Zendejas

# Introduction

- We set out to create a product that would make it easier to study, and be prepared for exams.
- Our goal was to create a program that would allow you to upload course notes, and materials to create AI generated user tailored study guides.
- This project is important to all of us because we value our time and wanted to find a way to study more efficiently and effectively.



# Data and Methodology

## Data description and preprocessing

- Plain text, TXT, MD
- PDF, DOCX, DOC
- `allenai/olmOCR-2-7B-1025`: Image-to-text (OCR)

## AI related algorithms

- We chose `allenai/olmOCR-2-7B-1025` because it is good for processing different types of documents
- Will return clean, formatted text
- Equations and tables will be turned into LaTeX and HTML respectively
- Then it is given to `moonshotai/Kimi-K2-Instruct-0905` to generate the actual quiz

# Interfacing with Large Language Models

- Frontend requires **structured data**
- LLMs: prompt injection, malformed data
- Use JSON Schema to verify LLM output

Kimi K2 and other models...

Vercel AI SDK, Zod Schemas

```
1 ts createQuiz.ts 2 docx-files-rebase NULL-PORT.local 1 < 2025-12-01
createQuiz.ts > createQuiz
2 .....try {
3 .....// Call out to LLM to generate quiz.
81 .....const res = await generateObject({
1 .....model: huggingFace(model),
2 .....maxOutputTokens,
3 .....temperature,
4 .....schema: Quiz,
5 .....system: systemPrompt,
6 .....prompt: notes,
7 .....schemaName: "Quiz",
8 .....schemaDescription: "An object that represents a quiz for student
9 .....studying.",
9 .....});
#1 ts createQuiz.ts 2 ts quizSchema.ts 3 docx-files-rebase <ORT.local 1 < 2025-12-01
quizSchema.ts
3 import * as z from "zod";
4
5 export const Quiz = z.object({
6 .....title: z.string().min(1),
7 .....questions: z.array(z.object({
8 .....prompt: z.string().min(1).register(z.globalRegistry, {
9 .....description: "Multiple choice question prompt.",
10 .....}),
11 .....correctOption: z.string().min(1).register(z.globalRegistry, {
12 .....description: "The only correct answer for the multiple choice question.",
13 .....}),
14 .....incorrectOptions: z.array(z.string()).min(1).register(z.globalRegistry, {
15 .....description: "List of incorrect answers for the multiple choice question.",
16 .....}),
17 .....}).min(1).max(20),
18 .....}).register(z.globalRegistry, {
19 .....description: "An object that represents a quiz for student studying.",
20 .....});
21 export type Quiz = z.infer<typeof Quiz>;
```



# Experimental Results (DEMO)

Visualization of results (demo)

# Discussion

## Interpretation of Results

- Request parallelization -> fast output
- Quizzes are acceptable for quick study

## Future Ideas

- Quiz sharing
- Free-response

## Challenges encountered

- olmOCR requires specific image sizes
- LLMs are vulnerable to prompt injection
- User image prompts need to be processed
- Unsanitized input -> vulnerability



# Discussion



## Interpretation of Results

- Request parallelization -> fast output
- Quizzes are acceptable for quick study



## Challenges encountered

- olmOCR requires specific image sizes
- LLMs are vulnerable to prompt injection
- User image prompts need to be processed
- Unsanitized input -> vulnerability



## Future Ideas

- Quiz sharing
- Free-response

# Appendix - Code snippet

```
const documentResults = await Promise.all(
  ...files.filter(f => f instanceof File).map(extractTextFromDocument)
);
```

```
let res = await Promise.all(
  ...// For each page (1-indexed)
  ...Array(pdf.numPages).keys().map(n => n + 1)
  ...// convert page to image
  ...// .map(n => pageToImage(pdf, n))
  ...// convert image to text with olmOCR, limit parallelism
  ...// .map(p => p.then(b => limit(olmOCR, b)))
);
```

array map-reduce + functional programming ❤️



## Conclusion

We worked together and leveraged our strengths to do things like research, create visually and functionally solid UI.

Together we learned a lot about how to implement large language models.

We think that using AI to further education and empower students to become better students. As AI advances we believe it is going to be super important to continue building tools like this to help students use AI in a responsible manner that assists them in their learning.

# Appendix - Prompts

```
const systemPrompt = `
Attached is one page of a document that you must process.
Just return the plain text representation of this document as if you were
↳ reading it naturally. Convert equations to LaTeX and tables to HTML.
If there are any figures or charts, label them with the following markdown
↳ syntax ![Alt text describing the contents of the
↳ figure](page_startx_starty_width_height.png)
Return your output as markdown, with a front matter section on top specifying
↳ values for the primary_language, is_rotation_valid, rotation_correction,
↳ is_table, and is_diagram parameters.
`;
```

```
const systemPrompt = `
You are a backend assistant. You generate helpful, useful quizzes of an
↳ appropriate length to help the user study.
Your input is the user's class notes. Your output is a Quiz object. You must
↳ include 'Quiz' in the title.
`;
```



## Appendix - Git Repository

[github.com/nullobsi/cs450-ai-project](https://github.com/nullobsi/cs450-ai-project)