YRIKKA Take-Home Challenge Report

Algorithms and Frameworks

As per the challenge's requirements, React was used for the application's frontend, while a Flask server was set up to handle requests to the OpenAI API.

This project uses Next.js as its React framework, which was chosen for its superior developer experience and built-in optimizations (for example, the `<Image />` component which optimizes image loading). TypeScript was used to enforce type safety during development. Styling the application was accomplished with Tailwind CSS and the shadcn/ui component library for its flexibility with utility classes.

The Flask app uses Blueprints for modularity, helping to structure the codebase and provide room for more routes and services to be easily added. The `openai` package for Python was used to conveniently make requests to the OpenAI API.

Use of AI Tools/Assistances

Generative AI tools were leveraged during the development process to speed up the development process. Below are the tools used and their roles in the creation of the project:

- GitHub Copilot - inline code editing and chat from IDE
- ChatGPT - for exploring different ways to approach the program structure and other high-level tasks (was also particularly helpful for abstracting away different components into different files)
- Perplexity - for answering questions related to OpenAI API specifications
- Supermaven - inline text completions

Future Improvements

The user experience of the chat interface could be greatly improved by introducing a workflow to iteratively edit images. This kind of feedback loop could start with an image provided by the user or one generated from a prompt, then edit and refine the current image by feeding in new prompts with the most recent image generation. The challenges of implementing this lie with the way image data is provided. I ran into CORS issues while trying to download the image contents from the returned URL, and I was unable to figure out how to send a base64 image response back to the frontend. As the OpenAI API requires an image file in order to perform image edits, I was unable to edit images returned from the image generation process.

The image editing mechanism could also be improved on. Due to the rather constrained limitations of DALL-E's image editing capabilities, a mask is required in order to inpaint the image. For sake of simplicity, a square in the middle of the image was used as the mask for all image edits - however, the user should also have the option to choose masked region so that any part of the image could be edited.

The sidebar was created with the hopes of supporting multiple chats in the future, where each chat is used to generate a new image.