**COMP4651 Term Project**

**Topic: DreamFrame - Serverless Image Generation and Management System Using AWS Lambda with OpenAI API and GitHub Pages**

**Name: HSU Siu Kit(20960426), KOK Siu Chung(20896338)**

## 1. Introduction

### Background:

Cloud computing has significantly transformed software development by offering scalable and efficient solutions tailored to diverse user needs. Among various architectures, serverless architecture has emerged as a powerful approach, enabling developers to focus on writing code without managing complex infrastructure. AWS Lambda is a leading serverless computing service that allows code execution in response to events without server provisioning, enhancing scalability, optimizing resource utilization, and reducing operational costs. The rise of generative AI has further integrated AI into applications, particularly in creative domains like image generation. By leveraging AI models from OpenAI, developers can create applications that convert user-defined prompts into unique visuals.

### Motivation:

This project, named DreamFrame, is driven by the increasing demand for AI-driven applications that foster creativity and streamline workflows. Many existing solutions struggle with scalability and user engagement due to their reliance on traditional server-based architectures. By adopting a serverless model, we aim to provide a robust platform for users to generate and manage images based on their creative inputs without concerns about underlying infrastructure. Additionally, our platform includes a prompt refinement function that enhances input quality before image generation, addressing common challenges users face in producing high-quality images. This project seeks to empower users—whether professional prompt engineers, artists seeking inspiration, or marketers needing quick visual content generation without any experience in writing a great prompt—by offering a seamless experience.

### Objective:

- Demonstrating the effectiveness of AWS Lambda as a serverless solution for handling API requests related to image generation.
- Implementing a prompt refinement function that enhances user input before it is processed by the OpenAI API.
- Exploring the integration of OpenAI's API for generating images based on refined user-defined prompts.
- Providing an intuitive web interface that facilitates prompt refinement and efficient image management.
- Target diverse audiences such as artists, marketers, content creators, and educators who can benefit from quick and creative image generation solutions.

## 2. Methodology

This section outlines the development of our serverless web application utilizing AWS Lambda and the OpenAI API for image generation. It covers the system architecture, front-end and back-end implementation, user workflow, and challenges encountered during development.

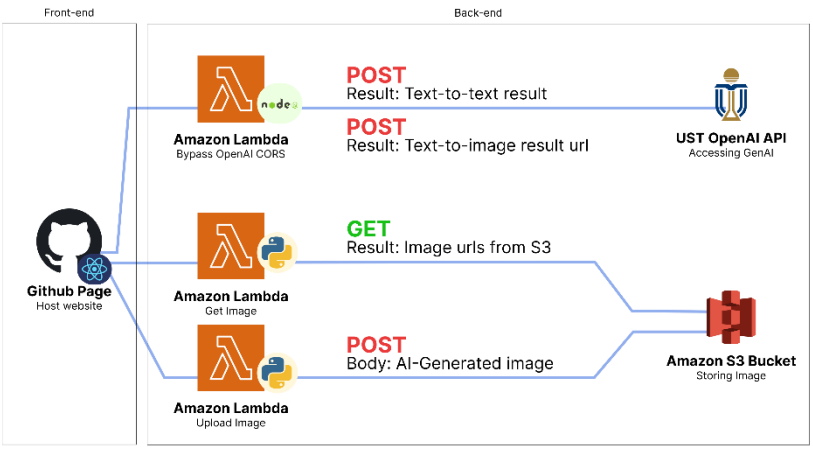
### System Architecture:

The architecture of our web application, DreamFrame, leverages serverless computing principles by integrating AWS Lambda and the OpenAI API for efficient image generation based on user-defined prompts. This design ensures scalability and minimizes infrastructure management.

(i) Components involved:

- **Front-end**: Hosted on GitHub Pages, the user interface allows users to input prompts and interact with the application.
- **API Gateway**: AWS API Gateway serves as an entry point for all user requests, routing them to the appropriate AWS Lambda functions.
- **AWS Lambda**: This serverless compute service executes backend logic in response to API requests, handling user input and managing image get and uploads to S3.
- **OpenAI API**: (DALL·E and GPT) This external service generates images based on refined prompts provided by users.
- **S3 Storage**: Amazon S3 is used for storing generated images, making them accessible via unique URLs.

(ii) Interaction Flows:




The application supports multiple interaction flows, each facilitated by AWS Lambda functions:

- Image Generation:
  - Direct Prompt Use: When a user submits a prompt, it is sent to AWS API Gateway, which routes it to a Lambda function that called the UST OpenAI API. The image data generated is then returned to the front-end.
  - Prompt Refinement: Users can refine their prompts before generating images. This involves:
    - Sending the initial prompt to AWS API Gateway for refinement.
    - The refined prompt is then used to generate an image through the same process.
- Image Upload to S3: If a user chooses to upload an image, an instruction is sent to API Gateway, which routes the request to a Lambda function that handles the upload to Amazon S3.
- Community Creation: For retrieving all generated images, a request is sent to API Gateway, which directs it to a Lambda function that fetches the image from S3 and returns it to the front-end.

For detailed descriptions of user interactions within these flows, please refer to the **User Workflow** section.

**Front-End Development:**

The front end of our web application is hosted on GitHub Pages (github.io) and is designed for simplicity and ease of use. Built with React and TypeScript, it features an input prompt area where users can enter prompts to generate or refine text. The interface includes four key buttons: one to refine the text, one to generate an image based on the prompt, one to download the generated image, and another to upload images to Amazon S3. Additionally, there is a browse function that allows users to view community creations or manually upload their own images. The application leverages Daisy UI and TailwindCSS for styling, ensuring a responsive and visually appealing design. This straightforward design ensures a seamless user experience while facilitating

interaction with the application's core features.

**Back-End Development:**

The back end of our web application is built on a serverless architecture using AWS services, designed to efficiently handle user requests and manage data. The key components of the back end include:

- AWS API Gateway: Acts as the entry point for API requests. When a user submits a prompt for refinement or image generation, the request is routed through the API Gateway to a specific AWS Lambda function.
- AWS Lambda Functions(Python, Node.js): These functions handle different tasks:
  - Prompt Processing: If generative AI is to be used, the prompt is sent to a Lambda function that bypasses OpenAI CORS restrictions and communicates with the UST OpenAI API (for DALL·E and GPT) to generate refined prompt or an image. The response is then sent back to the front-end.
  - Image Uploads: If a user decides to upload an image, an instruction is sent to API Gateway, which routes it to another Lambda function that manages the upload process to Amazon S3.
  - Image Retrieval: Retrieving images follows a similar flow through API Gateway and Lambda functions.
- Amazon S3: This service is used for storing user-uploaded images and generated outputs, ensuring that images are easily accessible and manageable.
- Interaction Flow: When a user interacts with the application:
  - Submitting a prompt trigger an API call to AWS API Gateway.
  - The gateway routes this request to the appropriate Lambda function for processing.
  - The Lambda function either generates an image via the OpenAI API or handles image uploads/downloads as needed.
  - Responses are sent back to the front-end for display or further action.

**Prompt Refinement Function:**

The Prompt Refinement Function enhances user input before it is sent to the OpenAI API for image generation. By refining prompts, this function improves clarity and specificity, resulting in higher-quality generated images. Users can refine their prompts through an intuitive interface, allowing them to enhance their initial ideas without technical expertise. They can accept suggested refinements or modify them further based on their creative vision. The refinement process analyzes the initial prompt using predefined templates that guide users in formulating more descriptive requests for the LLM. For instance, the LLM is provided with a structured prompt that includes:

**Imagine you are an artist tasked with creating an AI-generated image. Please provide a detailed prompt in 100 words that includes**

**1. **A clear description of the object**: What is it? What are its key features?**

**2. **The setting or background**: Where is the object located? What is the environment like?**

**3. **Specific details**: Include colors, textures, lighting, and any other elements that enhance realism.**

**4. **Artistic style**: What kind of artistic approach should be taken (e.g., hyper-realistic, impressionistic)?**

**Your detailed prompt should be for the following object: **{text input}**.**

This structured approach ensures that the OpenAI API receives clear and descriptive prompts, enhancing its ability to generate relevant images based on user intent. By providing a framework for refining prompts using predefined templates, the application significantly improves image quality. In developing this refinement function, we applied principles of prompt engineering to bridge the gap between user language and system language. This involves translating colloquial user inputs into detailed and

specific prompts that align with generative AI capabilities. Techniques including role assignment and contextual framing establish a creative context for the LLM, while encouraging specificity and detailed descriptions ensure comprehensive outputs. Utilizing structured guidance and categorization helps the LLM focus on various aspects of the image, while providing artistic direction and clear instructions narrows down the output's focus. Additionally, encouraging elaborative responses invites richer detail in the generated content. This method ensures that well-structured prompts enhance relevance and quality, fostering a more engaging and productive creative process. Users are more likely to receive outputs that align with their expectations. The prompt refinement features not only streamline interactions but also empowers users to explore their creativity more effectively.

**User Workflow:**

This section outlines the user workflow for interacting with our web application, detailing each step from prompt input to image generation and management. The step-by-step user interaction process is as follows:

(i) Inputting Prompts: Users access the application via the GitHub Pages-hosted front-end and enter their desired prompt into a text input field.

(ii) Prompt Refinement: Users have the option to refine their prompts for better results. If they choose to refine, they can click the “Magic Prompt” button. The prompt is sent to AWS API Gateway, which routes it to a Lambda function that processes and returns a refined version. Users can review the refined prompt before proceeding.

(iii) Generating Images: After entering or refining a prompt, users click the "Generate Image" button. The prompt (either original or refined) is sent to AWS API Gateway, which routes it to a Lambda function that calls the UST OpenAI API. The generated image is then returned to the front-end and displayed to the user.

(iv) Uploading Images to S3: Users can upload images by clicking an "Upload" button. An instruction is sent from the front-end to AWS API Gateway, which routes it to a Lambda function that handles the upload process to Amazon S3.

(v) Community Creations: Users can view previously generated or uploaded images by navigating to “Browse.” A request is sent to AWS API Gateway, directing it to a Lambda function that retrieves images from S3 and displays them on the front-end.

**Users’ Experience (UX):**

User experience (UX) is a fundamental aspect of our web application, significantly influencing user interaction and overall satisfaction. A well-designed UX encourages creativity and engagement, making it easier for users to achieve their goals.

(i) Intuitive Interface Design: The application features a clean, user-friendly interface that simplifies navigation. Users can easily input prompts, generate images, and manage their galleries without confusion.

(ii) Prompt Refinement Guidance: The prompt refinement process is straightforward, providing users with clear instructions and feedback on enhancing their prompts for better results.

**Challenges and Solutions:**

The first challenge was finding a suitable serverless service to meet our application needs. We addressed this by utilizing AWS Lambda, which allows us to run code without provisioning or managing servers, providing a scalable and efficient solution for our back-end processes.

The second challenge involved the complexity of using API keys to interact with the OpenAI API. Making requests requires careful handling of API keys and managing responses from multiple APIs. To streamline this process, we implemented a structured flow where AWS Lambda functions to handle API requests and responses, simplifying interactions and ensuring secure management of API keys.

## 3. Insights

**Outcome and Impact:**

The web application has been successfully deployed, implementing core functionalities such as serverless architecture with AWS Lambda, prompt refinement, and image generation using the OpenAI API. Initial testing indicated that users found the interface intuitive and appreciated the ability to refine their prompts before generating images. Feedback highlighted that this feature significantly improved the quality of generated images. We hope this DreamFrame contributes to the growing intersection of AI and creative processes by providing a useful tool that simplifies image generation and enhances user creativity. As more individuals and businesses seek efficient ways to create visual content, applications like ours can streamline workflows across various industries, from marketing to education.

**Lesson:**

Through this project, our team gained valuable insights into both technical and collaborative aspects of software development. A key lesson was the importance of clear communication within our small team, allowing us to effectively split the workload according to each member's strengths. We also learned to prioritize user experience by considering different users' needs in our application. On the technical side, we developed a deeper understanding of AWS (especially Lambda), serverless functions, generative AI, and API integration. We tailored prompts to suit various user types and learned the best practices for integrating web services using AWS and APIs.

**Potential Extension:**

- User Accounts: Implemental user accounts to allow personalized experience, including saving favorite prompt and picture or making the generated picture upload to a private bucket
- Collaboration Features: Adding functionalities that allow multiple users to collaborate on image generation projects could foster community engagement.
- Flexibility of Customized AI Experiences: Allowing users to change generative AI's parameter (such as temperature and token amount) or changing the pre-set prompt
- Data Analytics or Statistic: In the "Community Creations" section, we can add the prompt used to generate the picture. Also, we can add some statistics and perform some data analysis on the prompt used or the picture generated.

## 4. Summary

This report presents DreamFrame, a serverless web application that utilizes AWS Lambda and the OpenAI API, enabling users to generate images from refined prompts. It highlights the integration of serverless architecture with generative AI, enhancing user experience through prompt refinement that leads to improved image quality. Users found the application intuitive, empowering artists and marketers alike to streamline their workflows. Future enhancements may include additional AI features, user accounts, collaboration tools, and analytics to further expand its impact on creative processes.

## 5. Github URL

**Deployed Website:**

Github Classroom: (No permission)

Backup: https://willthree123.github.io/4651-web

**Source Code:**

Classroom: https://github.com/COMP4651-24fall/project-group-21-dreamframe

Backup: https://github.com/willthree123/4651-web

## 6. Reference

Acknowledgment: Portions of this document were edited for conciseness, precision, and grammatical accuracy using OpenAI's ChatGPT language model. The AI tool assisted in refining the language and enhancing the overall clarity of the text.