In today's information-based society, e-mail is an important tool for interpersonal communication. People rely extensively on email not only for personal communication, but also for project management and collaboration.

Statistics show that in contemporary society, the use of e-mail is on the rise year after year [1]. Given the sheer volume of data being transmitted through emails, extracting key details accurately in these emails is a matter of great importance. Extracting important data such as meeting dates, pending tasks, and looming deadlines from this huge volume of emails can greatly impact an individual's work planning and time management.

1.3. Motivation and Project Idea

Faced with a constant influx of emails every day, people are finding it increasingly difficult to sift through and prioritize important content. An overabundance of emails tends to obscure important data such as appointment schedules, task and project due dates, and more. While reputable email platforms such as Gmail and Outlook offer user-friendly interfaces, they lack the advanced functionality to autonomously identify email content to categorize or extract important details. This oversight can lead to inefficiency, as users may miss opportunities or spend too much time reassembling information, thus compromising productivity. As a result, there has been a surge in demand for tools that can skillfully highlight and present key information from emails, and that can help individuals navigate their email-driven routines more effectively.

Project idea:

- To build up a program that can read emails contents from platforms like Gmail/Outlook etc.
- To extract information from emails.
- To automatically arrange those working deadlines/ meeting dates and merge them into a working schedule.
- To send out daily itinerary reminders to help users make work plans.
- To have ability to share meeting schedule details to social messaging applications (e.g., Whatsapp) with one click.

2. Methodologies

In this project, my goal was to develop a computer program to efficiently categorize and extract relevant information from Gmail emails. The main goal is to create a system that recognizes key details such as meeting times, tasks, and deadlines in email conversations, integrates them into the user's schedule while providing some schedule reminders, and additionally, I would like the program to have a sharing feature that allows for one click sharing of schedule details to social platforms.

In order to achieve this, I will be using a variety of technologies and programming methods. Gmail API will be the first step I need to touch, it provides the application with direct access to the email content. Gmail API ensures that the system gets the most up-to-date information from the user's emails. In order to grant access to email content and maintain secure authentication with OAuth 2.0, I must next learn how to utilize the Gmail API. In addition, the backend, which I will choose to develop using Flask, will be used to process email content, extract key calendar information, and manage database interactions. The front-end of my project application is planning to be built by using React, which can efficiently displays extracted email messages and schedules. I will integrated Flask-CORS for communication between the Flask backend and the React frontend. Going forward, I plan to add sharing functionality to the application that will allow users to share their schedules on social platforms with just one click. This future enhancement will bring more convenience to users and extend the utility of the app from personal scheduling to social sharing and collaboration.

In this project, I will also employ Natural Language Processing (NLP), which is useful for understanding and interpreting the content of Gmail emails. By using NLP techniques, I aim to develop algorithms capable of parsing email text to identify important information such as meeting times, tasks, and deadlines. This involves utilizing NLP libraries such as NLTK and spaCy to tokenize emails, extract entities, and classify content based on its relevance to the user's schedule. The extracted information will then be integrated into the user's calendar, ensuring that they are kept up-to-date with their commitments and deadlines. Through the application of NLP, the project aims to transform the way users interact with their emails, turning them into an organized and actionable resource for managing their schedules.

2.1. Gmail API

The Gmail API is a powerful tool provided by Google that allows developers to interact directly with Gmail accounts. By integrating the Gmail API into a computer program, a user's emails, labels, drafts and threads can be accessed and managed. By utilizing the Gmail API, we can obtain privacy authorization to parse email content, extract relevant information, and even automate tasks based on data received in emails. In conjunction with the Gmail API, I can attempt to write a computer program that provides a more organized and streamlined email management experience for users based on their specific needs. To set up and run an app that calls a Google Workspace API, down below are some procedures.

Figure2.1.1 The Google Could Project Main page

Incorporating the Gmail API into my project, I employed OAuth 2.0 authentication along with a distinct Client ID and Client Secret. This setup ensured secure and efficient management of user emails.

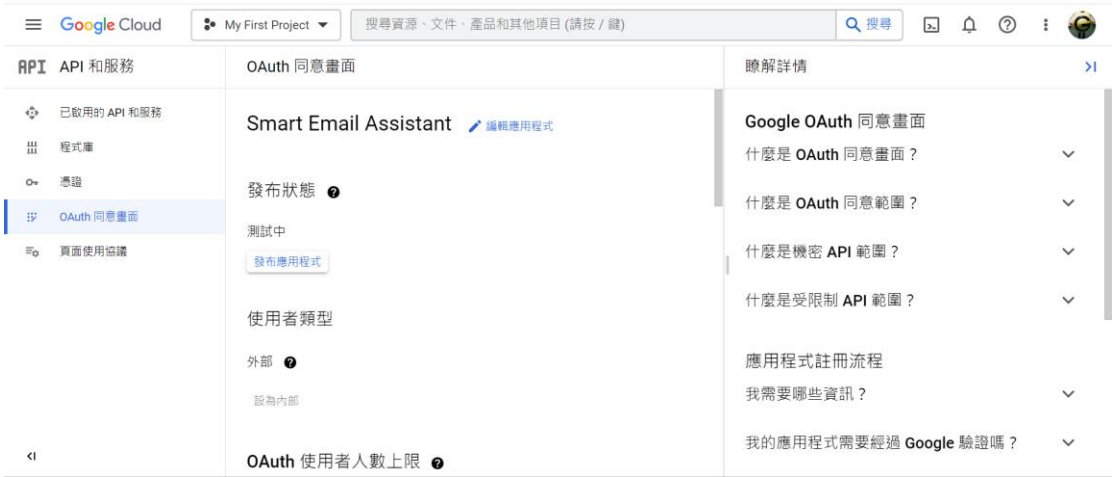


Figure2.1.2 The page that to employ OAuth 2.0 authentication

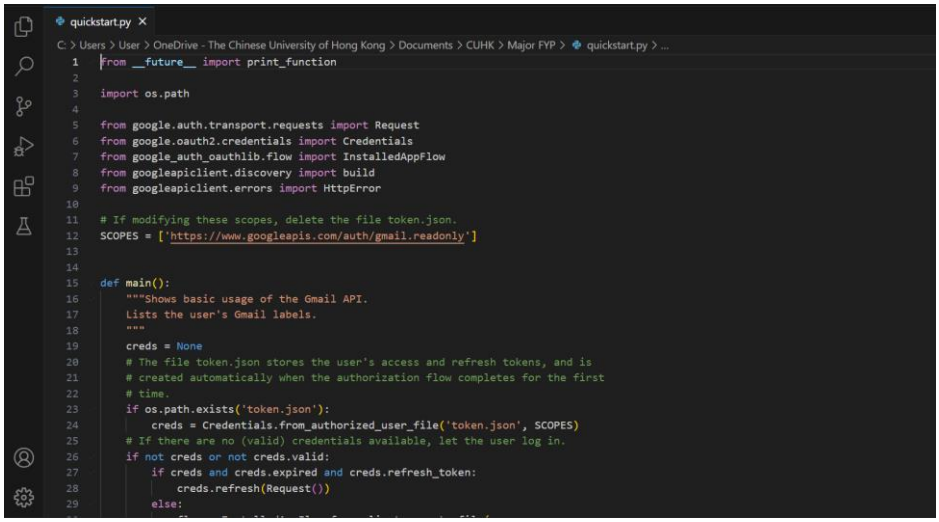


Figure2.1.3 The starting python code that to call Gmail API.

2.2. Flask

Flask is a powerful Python web framework for building web applications that is well suited for small projects due to its simplicity and flexibility. Flask provides functionality that lets me interact with Gmail API and React.

2.3. React

React is a popular JavaScript library for building user interfaces, React was developed by Facebook and is known for its efficiency and flexibility. It allows developers to create web

applications that update data without having to reload the page. React components are easy to manage and make it easy to develop applications, making it a popular choice for modern web development. Based on the usefulness of react, I will also choose to use react in this project.

2.4. The combined system of Flask and React

Flask and React can work effectively together in full-stack web development. In my project, Flask serves as the backend, handling server-side logic and data processing, and managing database interactions, while React is used to build the frontend, focusing on creating dynamic and responsive user interfaces. Data is exchanged between Flask and React via an API, usually in JSON format. With this combination, the work can be separated, with Flask handling the background processes and React focusing on the user experience.

In my project, Flask-CORS is used to communicate between the Flask backend and the React frontend. By implementing Flask-CORS, I am able to provide controlled access to resources on the Flask server from web pages hosted on different domains. This is important for the interaction between the React-based user interface and the Flask server, ensuring smooth data exchange and an integrated application experience.

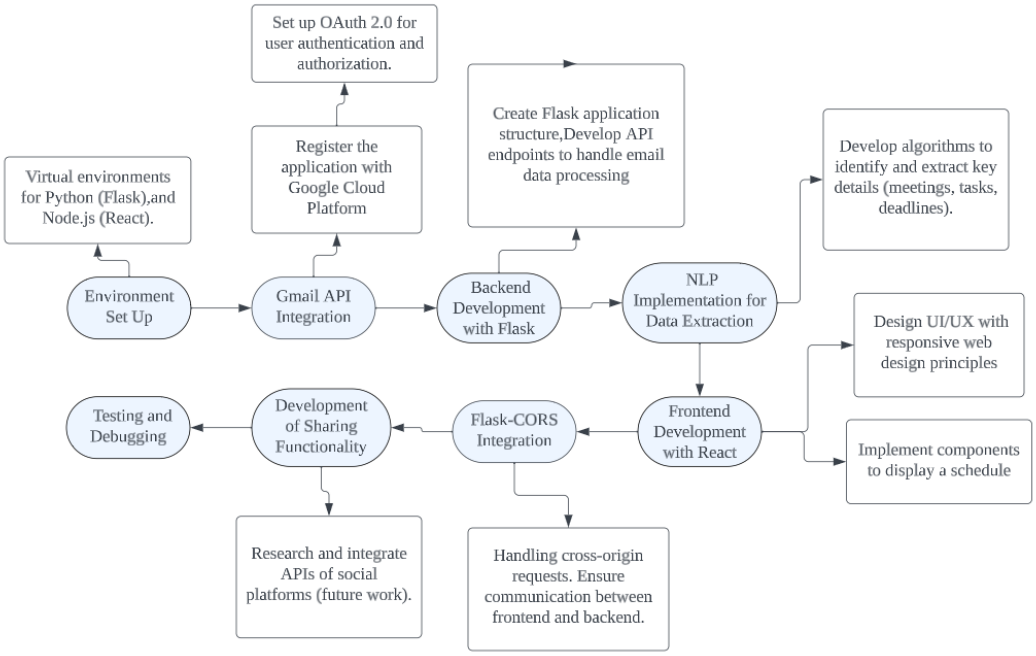
2.5. The Hugging Face Transformers library

The Hugging Face Transformers library is an open-source for natural language processing NLP tasks. It provides thousands of pre-trained models to perform tasks on texts such as classification, information extraction, question answering, summarization, translation, and more. These models are highly efficient at processing and understanding large volumes of text and will be very suitable for the project. By fine-tuning it using a pre-trained model, I am able to effectively recognize key details such as meeting times, tasks and deadlines. The library's NLP capabilities help accurately parse and understand the content of emails, which is important for integrating this information into a user's schedule.

2.6. Email Dataset

In my project, some open-source datasets will be used for training and fine-tuning my natural language processing models. It enables the models to learn the nuances of email language, understand specific contexts, and accurately extract key features such as dates, times, and action items. By using on the dataset, the models can offer personalized email categorization and information extraction, enhancing the overall functionality of the system.

2.7. Workflow



3. Results

3.1. Sample Email

Here is an example email that has been used to test the program, The content is generated by AI, which contains some information that need to be recognized and extracted, like the name of the event and the date/time, there are also some additional useless content. (Information that need to be extract is highlighted)

Subject: Meeting Confirmation: Project Alpha Review - November 30, 2023
Dear Team Members,
I hope this message finds you well. I am writing to confirm our upcoming meeting and provide you with all the necessary details:
Meeting Topic: Comprehensive Review of Project Alpha
Date: Wednesday, November 30, 2023
Time: 10:00 AM to 12:00 PM EST
Venue: Conference Room B, Level 3, Central Office Building / Zoom Link [Zoom Link for Remote Attendees]
Agenda:
Overview of Project Progress by Project Manager
Financial Review by Finance Lead
Feedback Session and Q&A
Discussion of Next Steps and Milestones
Preparation: Please review the Project Alpha progress report and the financial summary document sent earlier. Come prepared with any questions or feedback you might have.
Additional Notes: A light breakfast will be provided in the meeting room from 9:30 AM. Please inform me of any dietary restrictions.
I believe this meeting will be a valuable opportunity to collectively assess our progress and plan for the upcoming phases of Project Alpha. Your insights and expertise in your respective areas are critical for the success of this project.
Please confirm your attendance by EOD November 27, 2023, and let me know if you have any specific points you would like to discuss or add to the agenda.

Looking forward to a productive meeting. Thank you for your dedication and collaboration.
Best regards,
John Doe Project Coordinator

3.2. The Website page

While developing the front-end of the application, I used the DevExtreme React Scheduler component from the DevExtreme React UI component libraries[2], which was very helpful in creating a dynamic, user-friendly interface for email scheduling, and also served as a reference for my project on how to create a user-friendly interface that would show off the results of my work for the time being.

Like a normal calendar, this web application has three methods of displaying the schedule, including day/week/year. The date, time and event name successfully extracted from the email will be displayed here.

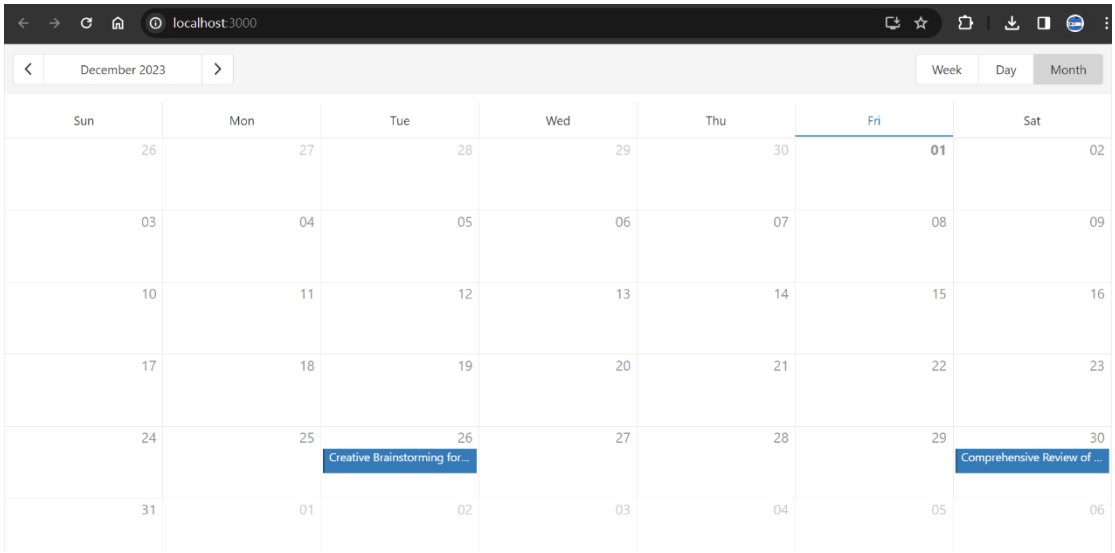


Figure3.2.1. Monthly mode of the schedule.

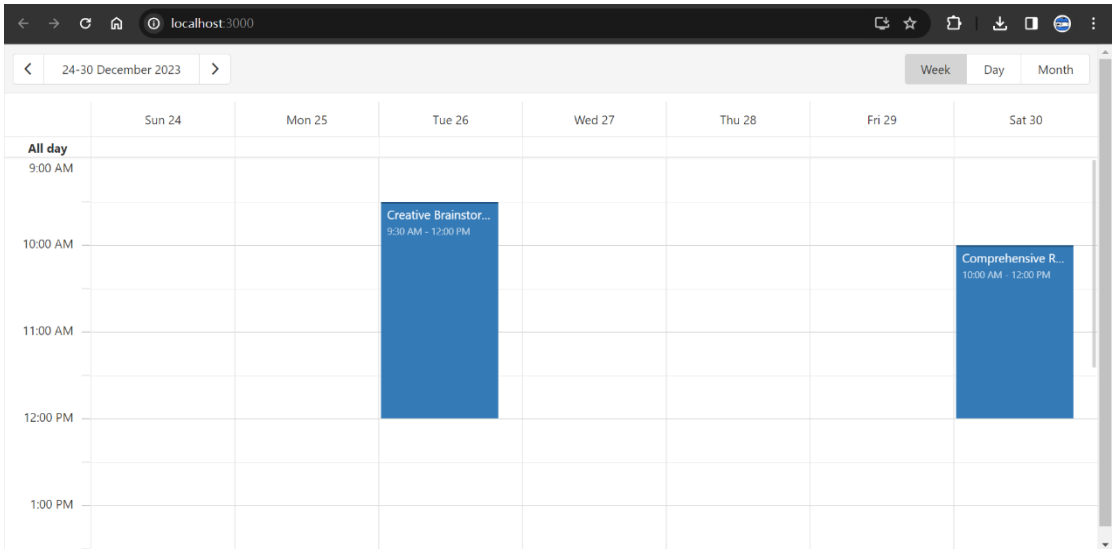


Figure3.2.2. Weekly mode of the schedule.

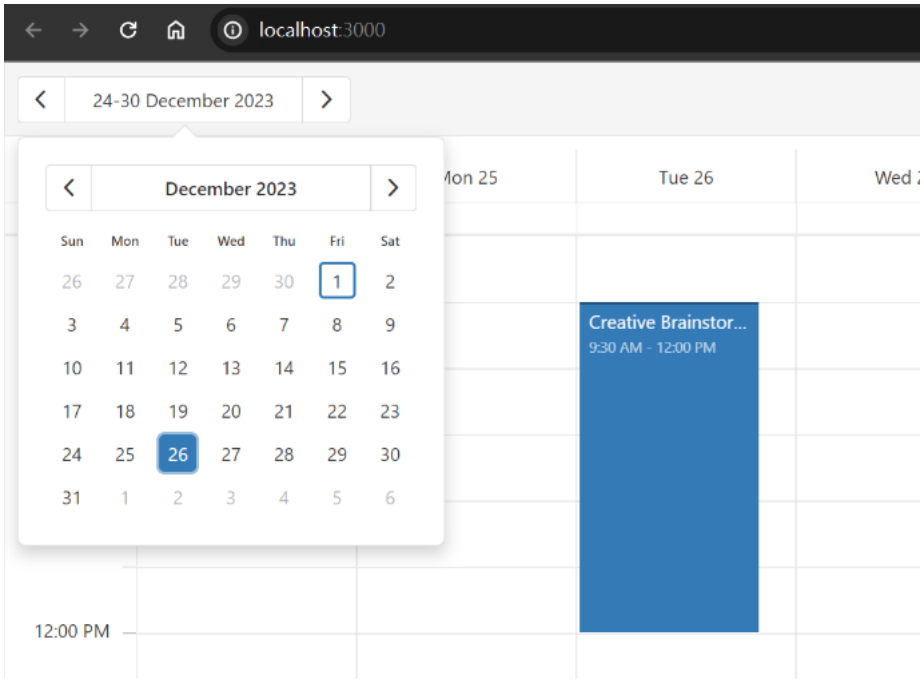


Figure3.2.3. Date selection function

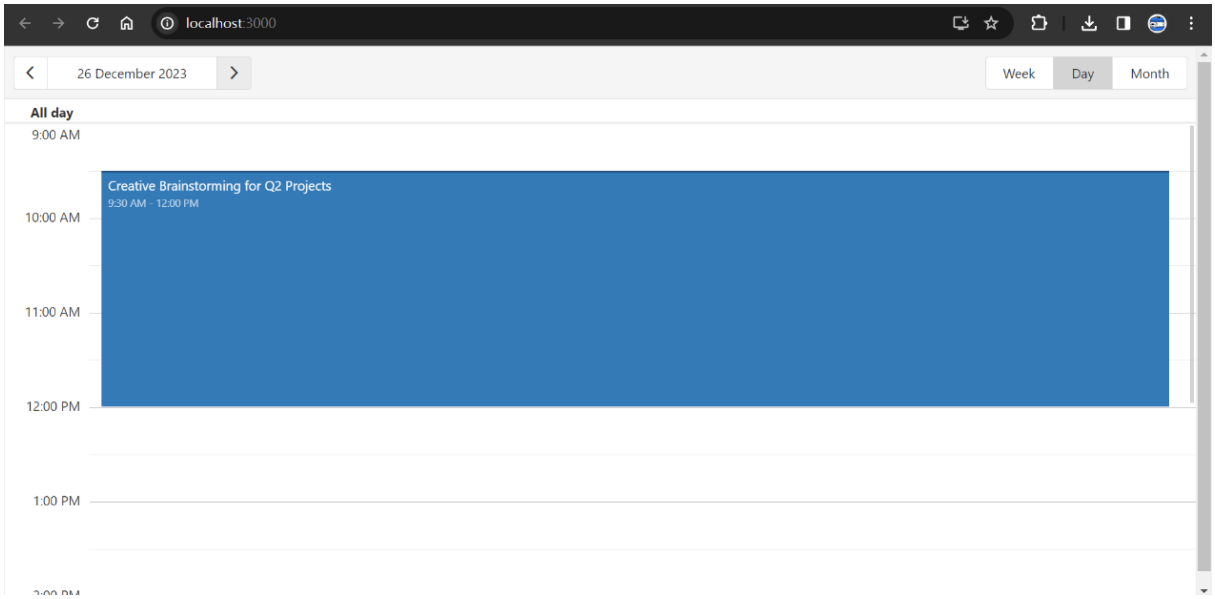


Figure3.2.4. Daily mode of the schedule.

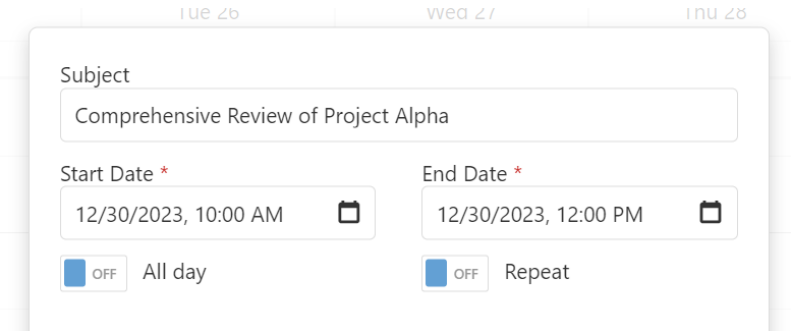


Figure3.2.5. The extracted Information details shown in the page.