



munch-mate



MB/CSE 648 GPT Application

Professor Michael Hu

Cynthia Widjaja

Kavya Kulkarni

Nathaly Cobo Piza

SaiLakshman Garikapati

Sanmati Vikas

Yaoching Chi

Agenda

- Introduction
 - Project Proposal
 - Features
 - User Personas
- Design
 - Design Workflow
 - System Architecture
 - Code Snippets
- Product Demo
- Project Management
 - Project Charter
 - Gantt Chart
 - Weekly Task and Report
- Lesson Learned
- Areas for Improvement

Project Proposal

Providing cloud solution to host an intuitive virtual assistant for restaurant recommendations.



- Restaurants recommendation based on location
- Trending food spots suggestions
- Recommendations based on dietary requirements
- Instagram-worthy restaurants and dishes



User Personas



Health-conscious

Seek nutritious and locally-sourced meals

Tech-savvy

Prefer seamless dining experiences enabled by technology

Social media-savvy

Attracted to visually appealing dishes and actively engage in sharing their dining experiences online.

Eco-conscious

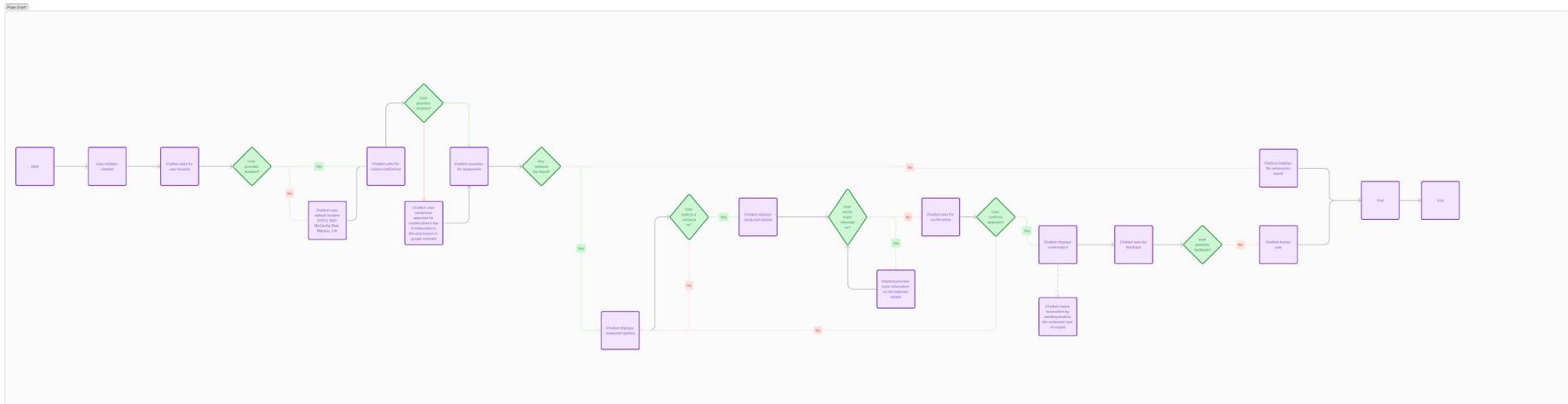
Favor sustainable dining options with practices like sourcing local ingredients, reducing food waste, and minimizing carbon footprints.

Experience-focused

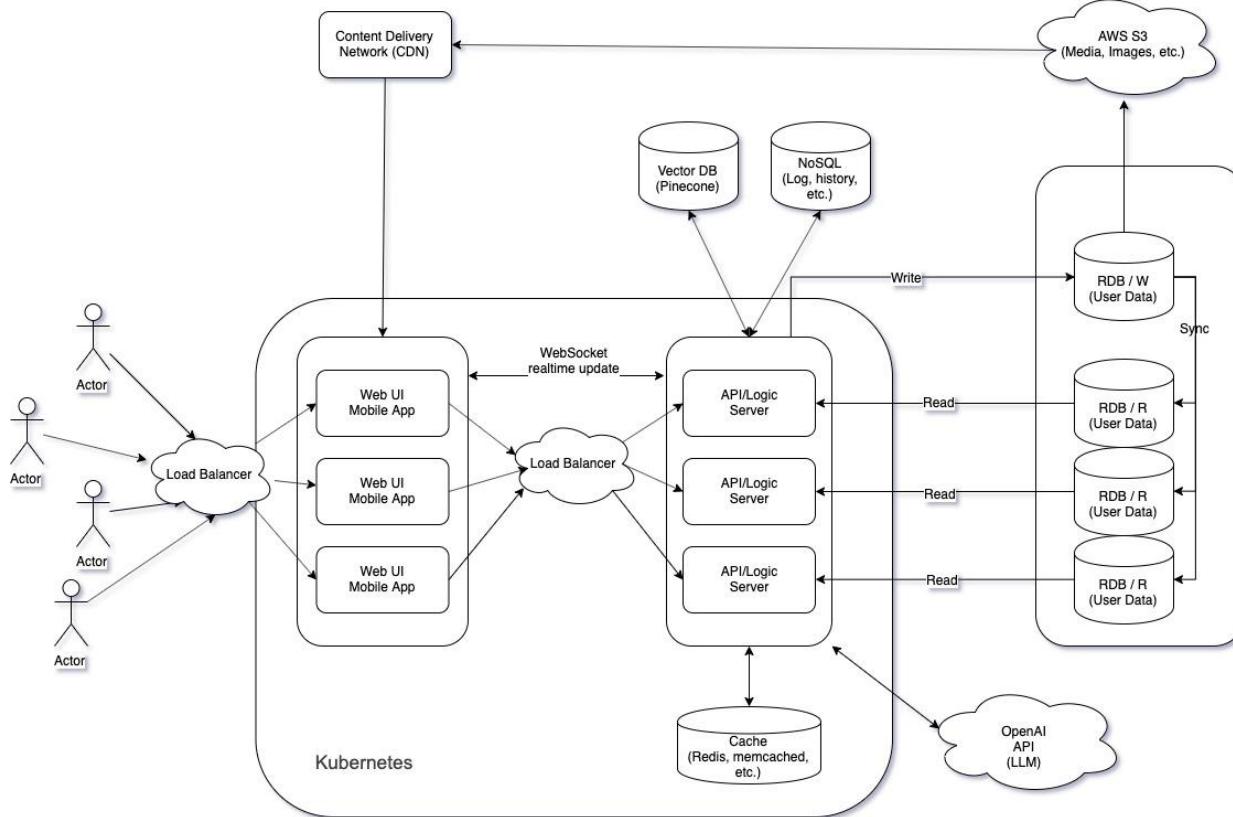
Value unique dining experiences with inventive menus, and thematic decorations

Design Workflow

Content [link](#) to Google Drive



System Architecture



Web UI Frontend (React.js) EC2

API/Logic Server Backend (Node.js) EC2

Created by Yaoching Chi

Code Snippets

MUNCH-MATE

outline

- (e) openai_model
- (e) embedding_model
- (e) model_temperature
- (e) pinecone_key
- (e) pinecone_namespace
- (e) pinecone_index
- (e) OpenAI
- (e) RESTAURANTS
- (e) PineconeEvents

OpenAIEvents

- constructor
- apiKey
 - getRestaurants*
 - makeReservation*
 - knowledgeBase*

createConversation

- completion
- err

embeddingQuery

- res
- err

functionCall

getRestaurants

makeReservation

knowledgeBase

- pc
- queryMsg

xq

res

contexts

prompt

delimiter

count

limit

role

content

unknown>

You, 9 minutes ago | 1 author (You)

```
> /*-
```

```
7
```

```
8 const {openai_model, embedding_model, model_temperature, pinecone_key, pinecone_namespace, pinecone_index} = require('../config');
```

```
9 const OpenAI = require('openai');
```

```
10 const {RESTAURANTS} = require('../constant/constants');
```

```
11 const PineconeEvents = require('./pineconeEvents');
```

```
12
```

```
You, 38 minutes ago | 1 author (You)
```

```
13 class OpenAIEvents {
```

```
14   constructor(apiKey) {
```

```
15     this.apiKey = apiKey;
```

```
16     this.openai = OpenAI({apiKey});
```

```
17     this.model = openai_model;
```

```
18     this.embeddingModel = embedding_model;
```

```
19     this.temperature = model_temperature;
```

```
20     this.availableFunctions = {
```

```
21       ...
```

```
22     }
```

```
23   }
```

```
24
```

```
25 }
```

```
26
```

```
27
```

```
28 async createConversation (messages, tools = null, tool_choice = null) {
```

```
29   try {
```

```
30     const completion = await this.openai.chat.completions.create({
```

```
31       messages, ... You, 7 days ago - updated add api key setting and verify api key -
```

```
32       model: this.model,
```

```
33       temperature: this.temperature,
```

```
34       tools: tools,
```

```
35       tool_choices: tool_choice
```

```
36     });
```

```
37     return completion
```

```
38   } catch (err) {
```

```
39     console.error(err);
40   }
41 }
```

```
42 > async embeddingQuery (query) { -
```

```
43   const response = await this.openai.embeddings.create({
44     query,
45   });
46   return response;
47 }
```

```
48 > functionCall (responseMsg) { -
```

```
49   const payload = {
50     role: "assistant",
51     content: responseMsg,
52   };
53   return payload;
54 }
```

```
55 > getRestaurants ((location, type)) { -
```

```
56   const response = await this.openai.chat.completions.create({
57     model: "gpt-3.5-turbo",
58     messages: [
59       {role: "user", content: `I want to eat ${type} food in ${location}.`},
60     ],
61     temperature: 0.5,
62   });
63   return response.choices[0].message.content;
64 }
```

```
65 > makeReservation ((restaurantName, date, time, partySize)) { -
```

```
66   const response = await this.openai.chat.completions.create({
67     model: "gpt-3.5-turbo",
68     messages: [
69       {role: "user", content: `I want to make a reservation at ${restaurantName} on ${date} at ${time} for ${partySize} people.`},
70     ],
71     temperature: 0.5,
72   });
73   return response.choices[0].message.content;
74 }
```

```
75 > async knowledgeBase ({location, cuisine}) { -
```

```
76   const response = await this.openai.chat.completions.create({
77     model: "gpt-3.5-turbo",
78     messages: [
79       {role: "user", content: `I want to know about ${cuisine} food in ${location}`},
80     ],
81     temperature: 0.5,
82   });
83   return response.choices[0].message.content;
84 }
```

```
85 > PineconeEvents () { -
```

```
86   const pineconeEvents = new PineconeEvents();
87   pineconeEvents.start();
88 }
```

```
89 > PineconeEvents () { -
```

```
90   const pineconeEvents = new PineconeEvents();
91   pineconeEvents.start();
92 }
```

```
93 > PineconeEvents () { -
```

```
94   const pineconeEvents = new PineconeEvents();
95   pineconeEvents.start();
96 }
```

```
97 > PineconeEvents () { -
```

```
98   const pineconeEvents = new PineconeEvents();
99   pineconeEvents.start();
100 }
```

```
101 > PineconeEvents () { -
```

```
102   const pineconeEvents = new PineconeEvents();
103   pineconeEvents.start();
104 }
```

```
105 > PineconeEvents () { -
```

```
106   const pineconeEvents = new PineconeEvents();
107   pineconeEvents.start();
108 }
```

```
109 > PineconeEvents () { -
```

```
110   const pineconeEvents = new PineconeEvents();
111   pineconeEvents.start();
112 }
```

```
113 > PineconeEvents () { -
```

```
114   const pineconeEvents = new PineconeEvents();
115   pineconeEvents.start();
116 }
```

```
117 > PineconeEvents () { -
```

```
118   const pineconeEvents = new PineconeEvents();
119   pineconeEvents.start();
120 }
```

```
121 > PineconeEvents () { -
```

```
122   const pineconeEvents = new PineconeEvents();
123   pineconeEvents.start();
124 }
```

```
125 > PineconeEvents () { -
```

```
126   const pineconeEvents = new PineconeEvents();
127   pineconeEvents.start();
128 }
```

```
You, yesterday | author (You)
1  // ...
2
3  const { Pinecone } = require('@pinecone-database/pinecone')
4
5  You, yesterday | author (You)
6  class PineconeEvents {
7    constructor({ apiKey, dbNamespace, dbIndex }) {
8      this.apiKey = apiKey;
9      this.pc = new Pinecone({ apiKey });
10     this.dbIndex = dbIndex;
11     this.dbNamespace = dbNamespace;
12     this.index = this.pc.index(dbIndex);
13   }
14
15   async queryDB ({query, topK}) {
16     const res = await this.index.namespace(this.dbNamespace).query({
17       topK,
18       vector: query,
19       includeValues: true,
20       includeMetadata: true,
21     });
22     return res;
23   }
24
25   // ...
26
27   You, yesterday + update: add pinecone database in backend.
28 }
29
30 module.exports = PineconeEvents
```

```
> MUNCH-MATE
  > You're leaving a gap! Edit now!
    ✓ OUTLINE
      (e) express
      (e) router
      (e) authMiddleware
      (e) openaiMiddleware
      (e) Conversation
      (e) TOOLS
      (S) SYSTIENTS
      (S) PERSONAS
      > (e) gen_sys_msg
      > (e) gen_assist_msg
      > (e) gen_user_msg
      > (e) handleFunctionCall
      > (S) router.get('/conversations') callback...
      > (S) router.get('/conversation/:id') callback...
      > (S) router.post('/init') callback
      > (S) router.post('/msg') callback
      (e) <unknown>

  1 > /*-
  2   |
  3   |
  4   |
  5   |
  6   |
  7   |
  8   const express = require('express');
  9   const router = express.Router();
 10  const authMiddleware = require('../middleware/auth');
 11  const openaiMiddleware = require('../middleware/openai');
 12  const Conversation = require('../models/conversation')
 13  const {TOOLS, SYS, CONTENTS, PERSONAS} = require('../constant/constants');
 14  const {openai, responseMsg} = require('../constant/constants');
 15  const gen_sys_msg = (msg) => ({ role: "system", content: msg })
 16  const gen_assist_msg = (msg, isFunctionCall=false) => ({ role: "assistant", content: msg, isFunctionCall })
 17  const gen_user_msg = (msg) => ({ role: "user", content: msg })
 18  const handleFunctionCall = async (openai, responseMsg) => {
 19    |
 20    |
 21    |
 22    |
 23    |
 24    |
 25    |
 26    |
 27    |
 28    |
 29    |
 30    |
 31    |
 32    |
 33    |
 34    |
 35    |
 36    |
 37    |
 38    |
 39    |
 40    |
 41    |
 42    |
 43    |
 44    |
 45    |
 46    |
 47    |
 48    |
 49    |
 50    |
 51    |
 52    |
 53    |
 54    |
 55    |
 56    |
 57    |
 58    |
 59    |
 60    |
 61    |
 62    |
 63    |
 64    |
 65    |
 66    |
 67    |
 68    |
 69    |
 70    |
 71    |
 72    |
 73    |
 74    |
 75    |
 76    |
 77    |
 78    |
 79    |
 80    |
 81    |
 82    |
 83    |
 84    |
 85    |
 86    |
 87    |
 88    |
 89    |
 90    |
 91    |
 92    |
 93    |
 94    |
 95    |
 96    |
 97    |
 98    |
 99    |
 100   |
 101   |
 102   |
 103   |
 104   |
 105   |
 106   |
 107   |
 108   |
 109   |
 110   |
 111   |
 112   |
 113   |
 114   |
 115   |
 116   |
 117   |
 118   |
 119   |
 120   |
 121   |
```

A screenshot of a web application showing a list of Italian restaurants in Milpitas. At the top, a message box says "Do we have the Italian food in Milpitas?" with a user icon. Below it, there's a section for "Curry Pizza House" with details like cuisine (Italian), phone number (408) 946-2300, and address (209 N Capitol Ave, Milpitas, CA 95035). Another section for "Pizza Guys" follows, with similar details. A third section for "Premier Pizza" is partially visible. At the bottom, a message asks if users want more information or assistance with reservations. A text input field at the bottom left says "Set prompt here...".



Product Demo



A grid of images illustrating different customer segments:

- Tech-Savvy:** A man and a woman sitting at a table, looking at a smartphone.
- Social Media-Savvy:** A man holding up a smartphone with a QR code displayed.
- Health-Conscious:** A table full of healthy food items like salads and smoothies.
- Eco-Conscious:** A woman sitting at a table with a straw bag.
- Experience-Focused:** A man wearing a VR headset while eating.

At the top, a search bar asks "What's the style you're looking for?"

A Touch of Project Management



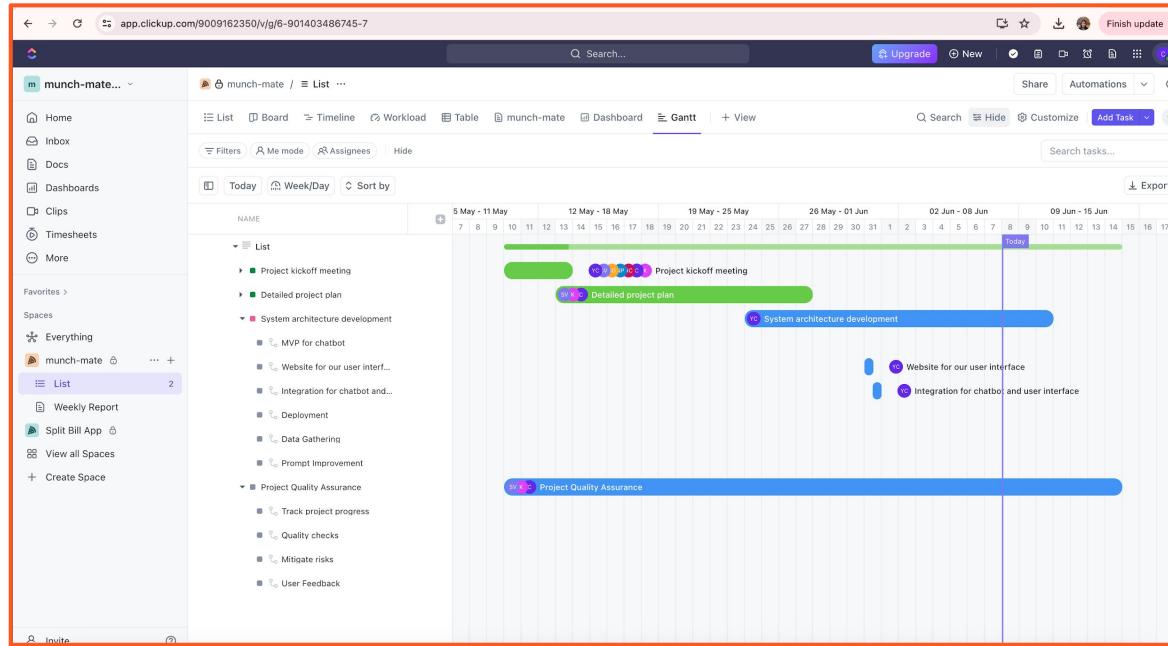
Project Charter

Content [link](#) to Google Drive or [Project Charter](#) link on Click Up

munch-mate	
Project Charter	
1. General Project Information	
Project Name	munch-mate
Executive Sponsors	Prof. Michael Hu
Department Sponsor	California Science and Technology University
Impact of Project	To enhance dining experience for users by providing personalized restaurant recommendations while boosting visibility and business for local eateries in Milpitas.

Gantt Chart

Link to [Gantt Chart](#) on Click Up



Weekly Task and Report

Link to [Weekly Report](#) on Click Up

The screenshot shows a Click Up card for a 'Weekly Report' under 'Week 5'. The card has a title 'Week 5' and a subtitle 'Monday (9:30 - 10:30pm)'. It contains an 'Agenda' section with the following tasks:

- 1 Prepare for project presentation (presenter?) 10mins
- 2 Architectural design (Sai, YaoChing, Cynthia)
- 3 Pinecone- function call for vectorize the data (Cynthia)
=> see colab file
(restaurant_list: sanjose, SantaClara and milpitas)
- 4 User personas updates and add the image (Nathaly, Vikas, Sai, Kavya)?
=> for reference, please see shared drive > documentation > [Excel GPT for Foodie](#)
g > Tiers and lifestyles
Cynthia create 1 User Persona for sample (please see shared drive > documentation> 1 Health Conscious Guest
User personas images, link [here](#)

The screenshot shows a Click Up 'Board' view for the 'munch-mate' project. The board has columns for 'OPEN', 'IN PROGRESS', and 'CLOSED'. Tasks include:

- OPEN**: Project Quality Assurance, System architecture development, Project Quality Assurance, Project Quality Assurance, Project Quality Assurance, Project Quality Assurance, Project Quality Assurance.
- IN PROGRESS**: System architecture development, Project kickoff meeting.
- CLOSED**: Detailed project plan, Project kick off meeting.

Lesson Learned

Effective teamwork and communication were crucial for keeping the munch-mate project on track and resolving issues promptly.

Team alignment, prioritization, firm decision making for progressive deliverables

Bigest lesson learnt would be understanding the use of GPT application which lead to greater involvement in building a project from scratch with a utmost team collaboration and also got a exposure to understand how to program with the help of ChatGPT.



The importance of having a complete source of truth, especially considering that there are specific areas where GPT may not completely cover the data. For example, in our project, it was crucial to create our own data set related to restaurants, addresses, emails , etc. and vectorize them so that our chatbot was reliable and accurate on its answers.

Finally I learned how to code the AI programing and can be integrated to web page.

Building our own AI Chatbot! Team work, Presentation within 5 minutes, Python, Agile Project Management

Areas for Improvement

Enhancing user feedback collection through methods like in-app surveys to better understand and meet user needs in the future.



Area to improve can be implementation of some more function such as voice to text option and google images reference chain link so that customers can get a better understanding of the ambiance before they proceeded to the reservation.

ability to provide accurate information, map and direction, listen for your request, connect to booking/payment gateway

A point of improvement would be to include more functions to make it a more complete chatbot. For example, you could feature restaurant reviews and give recommendations based on them, offer recipe ideas, etc.

The possibilities are endless. With more time, we could implement new and improved ideas to our bot like voice to text, maps, making reservations, UI improvisations, and much more. Once launched publicly, focus on user feedback and enhancements.

The course is too short that I can't make a "final product" for our team to integrate more functions, hope it could be a 2 terms course that we can keep learning how to make our chatbot perfectly

Thank you!

Cynthia Widjaja	Kavya Kulkarni	Nathaly Cobo Piza	SaiLakshman Garikapati	Sanmati Vikas	Yaoching Chi
			