

Ejercicio LLM

El ejercicio propuesto consiste en hacer un “lector de recetas” que nos ayude a hacer una lista de compras con su costo estimado, estimación de calorías y sugerencias de vino para maridar.

El funcionamiento es el siguiente:

- 1- Lectura de la(s) receta(s) en formato json. Nota: La app permite leer múltiples recetas, para que nos ayude a concentrar los ingredientes para hacer una única lista de compras.
- 2- Una vez leídas las recetas, se generan listas con el nombre de la receta, ingredientes y preparación
- 3- Con la lista de ingredientes de cada receta, se genera el primer prompt que pide que se agrupen y genere una tabla con “Nombre del ingrediente” y “cantidad”

Nota: aquí fue importante afinar el prompt para asegurarse de que no se incorporen etiquetas adicionales al resultado, así como limpiar el resultado en caso de que se hayan agregado comillas o “json” al inicio de cada cadena

- 4- Con el resultado anterior, se genera un nuevo prompt para solicitar que se haga una estimación de calorías por cada uno de los ingredientes
Nota: Como mejora en el futuro, se debería agregar la cantidad de personas para poder hacer un cálculo más aproximado.
- 5- Con resultado anterior, se genera un nuevo prompt para solicitar que se busque el costo aproximado de cada ingrediente en pesos MXN
- 6- Con los resultados anteriores, genero tablas tipo reporte que serán usadas para mostrar resultados y para hacer una sumatoria de “Costo Total” y “Calorías”
- 7- Adicionalmente, con los nombres de las recetas, hacemos un prompt adicional que nos permita solicitar a OpenIA sugerencias de vinos para maridajes. Dicho resultado también se convierte a tablas de reporte para ser utilizados en una respuesta final.
- 8- Finalmente, se genera un texto de resumen que nos indica los nombres de los platillos, la lista de compras desglosada en ingredientes, cantidad. Se agrega el total del costo aproximado y calorías.

Muchas felicidades!!
A continuación veremos los detalles para tu evento

Platillos a servir:
===
,Sopes Mexicanos,Sopa de tortilla,Fish & Chips
===

Lista de Compras:

	nombre	cantidad \
0	masa de maíz	2 tazas
1	agua	1/2 taza
2	sal	al gusto
3	Aceite para freír	al gusto
4	frijoles refritos	1 taza
5	carne deshebrada (pollo, res o cerdo)	1 taza
6	lechuga picada	1/2 taza
7	crema ácida	150 ml
8	queso fresco desmoronado	150 g
9	Salsa	al gusto
10	tortillas de maíz	8
11	jitomates maduros	4
12	chiles pasilla secos	2
13	dientes de ajo	2
14	cebolla	1/4
15	caldo de pollo	1 litro
16	epazote	1 ramita
17	aguacate en cubos	1
18	Pimienta	al gusto
19	filetes de pescado blanco (merluza, bacalao o ...	4
20	harina de trigo	1 taza
21	cerveza fría	1 taza
22	polvo para hornear	1 cucharadita
23	huevo	1
24	papas grandes	4
25	Vinagre de malta (opcional, para servir)	al gusto
26	Limón en gajos (para acompañar)	1 gusto

Costo Estimado:

**362 MXN

Calorias del Menu:

**4999

===

Sugerencias de Vinos:

===

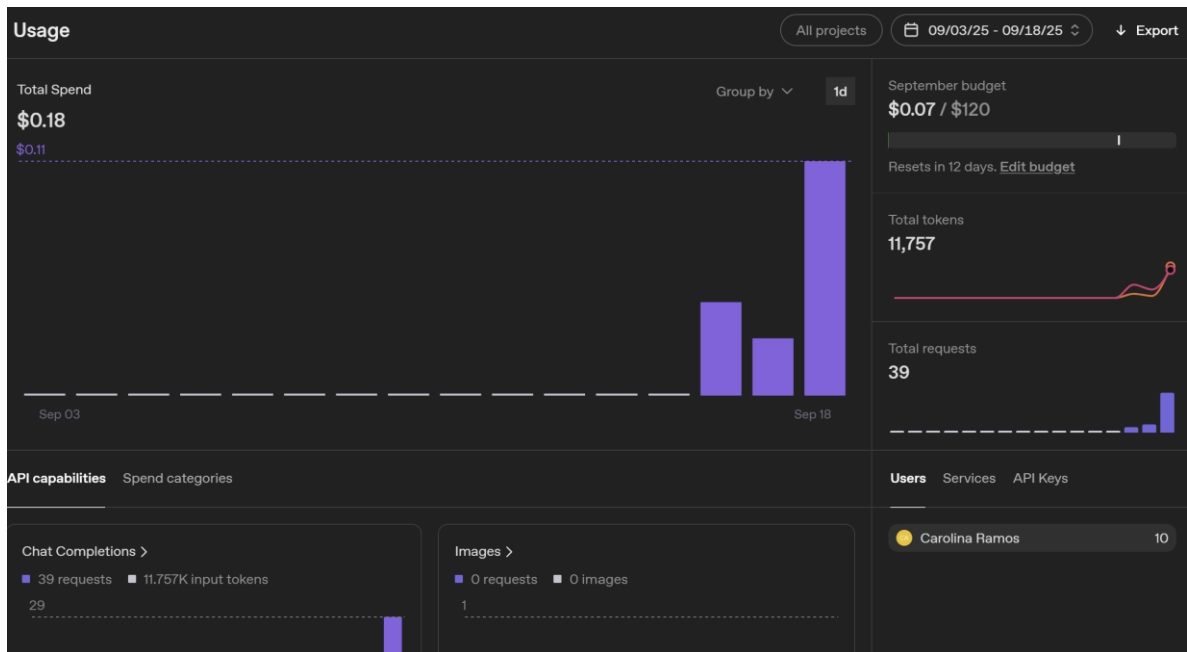
	Nombre del Vino	Tipo de UVA
0	Viña Esmeralda	Moscatel
1	Marqués de Riscal Rueda	Verdejo
2	Kim Crawford	Sauvignon Blanc

===

- 9- Como mejoras adicionales, se debería solicitar la estimación de tiempo y número de comensales, así mismo, podríamos hacer un prompt para solicitar ideas de presentación.

Pasos previos y herramientas utilizadas:

- Desarrollé el código en COLAB
- Utilice OPEN IA



Cada ejercicio se lleva 4 request de aprox 1,200 tokens en total

ANEXO CODIGO

<https://colab.research.google.com/drive/1q2dXXPd7dqH-pVuCHONO4g6bR0KgvSe8?usp=sharing>

NUEVO CODIGO LECTURA RECETAS


```
[188]
✓ 0 s

from openai import OpenAI
from google.colab import userdata

# Access your OpenAI API key from Colab's secrets manager
api_key = userdata.get('OPENAI_API_KEY')

client = OpenAI(api_key=api_key)

print(client.base_url)
```

 <https://api.openai.com/v1/>

```
[189]
✓ 0 s

import json
import glob
import os

directorio = "/content/"
```

```
[190]
✓ 0 s

def leer_recetas_desde_archivos(directorio):
    recetas = []
    for archivo in glob.glob(os.path.join(directorio, "*.json")):
        with open(archivo, "r", encoding="utf-8") as f:
            receta = json.load(f)
            recetas.append(receta)
    return recetas
```

```
[191]
✓ 0 s

# Llamada a la función
todas_las_recetas = leer_recetas_desde_archivos(directorio)

# Mostrar un resumen de lo que se leyó
for receta in todas_las_recetas:
    print(f"|| {receta['Nombre']}")
    print(f"    Ingredientes: {len(receta['Ingredientes'])}")
    print(f"    Pasos: {len(receta['Preparación'])}\n")
```

AGRUPACION RECETAS

```
[241]
✓ 0 s

for receta in todas_las_recetas:
    print(f"|| {receta['Nombre']}")
    print(f"    Ingredientes: {len(receta['Ingredientes'])}")
    ingredientes = ingredientes + receta['Ingredientes']
    print(receta['Ingredientes'])
    ingredientesTemp = ingredientesTemp + str(receta['Ingredientes'])
    Nombre_Recetas = Nombre_Recetas + ", " + receta['Nombre']
    ingredientes=ingredientesTemp

print(ingredientes) # lista de strings
print(Nombre_Recetas) # lista de strings

|| Sopas Mexicanas
Ingredientes: 10
['2 tazas de masa de maiz', '1/2 taza de agua', '1/2 cucharadita de sal', 'Aceite para freir', '1 taza de frijoles refritos', '1 taza de carne deshebrada (pollo, res o cerdo)']
|| Sopa de tortilla
Ingredientes: 13
['8 tortillas de maiz', '4 jitomates maduros', '2 chiles pasilla secos', '2 dientes de ajo', '1/4 de cebolla', '1 litro de caldo de pollo', '1 ramita de epazote', '1 aguacate']
|| Fish & Chips
Ingredientes: 11
['4 filetes de pescado blanco (merluza, bacalao o similar)', '1 taza de harina de trigo', '1 taza de cerveza fria', '1 cucharadita de polvo para hornear', '1 huevo', 'Sal al gusto']
|| Sopas Mexicanas,Sopa de tortilla,Fish & Chips
```

```

promptR1 = f"""

Dada la siguiente lista de ingredientes de una receta, genera un JSON con las tuplas de:

* La nombre del ingrediente
* Cantidad

===
{ingredientes}
===

NOTA IMPORTANTE: Devuelve sólo el JSON sin texto adicional, preámbulos o las tildels ``json
ya que se usará directamente en python. Tampoco agregues "ingredientes" como parte del JSON

"".strip()

print(promptR1)

Dada la siguiente lista de ingredientes de una receta, genera un JSON con las tuplas de:

* La nombre del ingrediente
* Cantidad

===
['2 tazas de masa de maiz', '1/2 taza de agua', '1/2 cucharadita de sal', 'Aceite para freir', '1 taza de frijoles refritos', '1 taza de carne deshebrada (pollo, res o cerdo)']
===

NOTA IMPORTANTE: Devuelve sólo el JSON sin texto adicional, preámbulos o las tildels ``json
ya que se usará directamente en python. Tampoco agregues "ingredientes" como parte del JSON

responseR1 = client.responses.create(
    model="gpt-4o",
    input=promptR1
)

print(responseR1.output_text)

```

```

promptR2 = f"""

Dada la siguiente lista de ingredientes de una receta, busca el aproximado de contenido calorico de cada uno y genera un JSON con las tuplas de:

* La nombre del ingrediente
* Cantidad
* Calorias Aproximadas

===
{responseR1.output_text}
===

NOTA IMPORTANTE: Devuelve sólo el JSON sin texto adicional, preámbulos o las tildels ``json
ya que se usará directamente en python. Tampoco agregues "ingredientes" como parte del JSON

"".strip()

print(promptR2)

Dada la siguiente lista de ingredientes de una receta, busca el aproximado de contenido calorico de cada uno y genera un JSON con las tuplas de:

* La nombre del ingrediente
* Cantidad
* Calorias Aproximadas

===
[
  {"nombre": "masa de maiz", "cantidad": "2 tazas"},
  {"nombre": "agua", "cantidad": "1/2 taza"},
  {"nombre": "sal", "cantidad": "1/2 cucharadita"},
  {"nombre": "Aceite para freir", "cantidad": "al gusto"},
  {"nombre": "frijoles refritos", "cantidad": "1 taza"},
  {"nombre": "carne deshebrada (pollo, res o cerdo)", "cantidad": "1 taza"},
  {"nombre": "lechuga picada", "cantidad": "1/2 taza"},
  {"nombre": "crema ácida", "cantidad": "1/2 taza"},
  {"nombre": "queso fresco desmoronado", "cantidad": "1/2 taza"},
  {"nombre": "Salsa", "cantidad": "al gusto"},
  {"nombre": "tortillas de maiz", "cantidad": "8"},
  {"nombre": "limón", "cantidad": "1 limón"},
  {"nombre": "cilantro", "cantidad": "1/2 taza"}
]

```

```

responseR2 = client.responses.create(
    model="gpt-4o",
    input=promptR2
)

```

```

print(responseR2.output_text)

```

```

[
  {"nombre": "masa de maiz", "cantidad": "2 tazas", "calorias": 900},
  {"nombre": "agua", "cantidad": "1/2 taza", "calorias": 0},
  {"nombre": "sal", "cantidad": "1/2 cucharadita", "calorias": 0},
  {"nombre": "Aceite para freir", "cantidad": "al gusto", "calorias": 120},
  {"nombre": "frijoles refritos", "cantidad": "1 taza", "calorias": 240},
  {"nombre": "carne deshebrada (pollo, res o cerdo)", "cantidad": "1 taza", "calorias": 250},
  {"nombre": "lechuga picada", "cantidad": "1/2 taza", "calorias": 5},
  {"nombre": "crema ácida", "cantidad": "1/2 taza", "calorias": 200},
  {"nombre": "queso fresco desmoronado", "cantidad": "1/2 taza", "calorias": 200},
  {"nombre": "Salsa", "cantidad": "al gusto", "calorias": 20},
  {"nombre": "tortillas de maiz", "cantidad": "8", "calorias": 480},
  {"nombre": "jitomates maduros", "cantidad": "4", "calorias": 100},
  {"nombre": "chiles pasilla secos", "cantidad": "2", "calorias": 50},
  {"nombre": "dientes de ajo", "cantidad": "2", "calorias": 10},
  {"nombre": "cebolla", "cantidad": "1/4", "calorias": 15},
  {"nombre": "caldo de pollo", "cantidad": "1 litro", "calorias": 60},
  {"nombre": "epazote", "cantidad": "1 ramita", "calorias": 2},
  {"nombre": "aguacate en cubos", "cantidad": "1", "calorias": 240},
  {"nombre": "queso fresco desmoronado", "cantidad": "100 g", "calorias": 250},
  {"nombre": "crema ácida", "cantidad": "100 ml", "calorias": 190},
  {"nombre": "Aceite vegetal para freir", "cantidad": "al gusto", "calorias": 120},
  {"nombre": "Sal", "cantidad": "al gusto", "calorias": 0},
  {"nombre": "Pimienta", "cantidad": "al gusto", "calorias": 0},
  {"nombre": "filetes de pescado blanco (merluza, bacalao o similar)", "cantidad": "4", "calorias": 400},
  {"nombre": "harina de trigo", "cantidad": "1 taza", "calorias": 400},
  {"nombre": "cerveza fria", "cantidad": "1 taza", "calorias": 150}.

```

```

promptR3 = f"""

```

```

Dada la siguiente lista de ingredientes agregados de varias recetas, agrupalos repetidos y genera una sola lista de compras con costo en pesos MXN aproximado y genera el res

```

```

* La nombre del ingrediente
* Cantidad
* calorias
* Costo aproximado

```

```

===
{responseR2.output_text}
===

```

```

NOTA IMPORTANTE: Devuelve sólo el JSON sin texto adicional, preámbulos o las tildes ```json
ya que se usará directamente en python. Tampoco incluyas o nombres el resultado como lista de compras o ingredientes, manten la estructura de solo "ingredientes" con las ti

```

```

"".strip()

```

```

print(promptR3)

```

```

Dada la siguiente lista de ingredientes agregados de varias recetas, agrupalos repetidos y genera una sola lista de compras con costo en pesos MXN aproximado y genera el resu

```

```

* La nombre del ingrediente
* Cantidad
* calorias
* Costo aproximado

```

```

===
[
  {"nombre": "masa de maiz", "cantidad": "2 tazas", "calorias": 900},
  {"nombre": "agua", "cantidad": "1/2 taza", "calorias": 0},
  {"nombre": "sal", "cantidad": "1/2 cucharadita", "calorias": 0},
  {"nombre": "Aceite para freir", "cantidad": "al gusto", "calorias": 120},
  {"nombre": "frijoles refritos", "cantidad": "1 taza", "calorias": 240},
  {"nombre": "carne deshebrada (pollo, res o cerdo)", "cantidad": "1 taza", "calorias": 250},
  {"nombre": "lechuga picada", "cantidad": "1/2 taza", "calorias": 5},
  {"nombre": "crema ácida", "cantidad": "1/2 taza", "calorias": 200},
  {"nombre": "queso fresco desmoronado", "cantidad": "1/2 taza", "calorias": 200},
  {"nombre": "Salsa", "cantidad": "al gusto", "calorias": 20},
  {"nombre": "tortillas de maiz", "cantidad": "8", "calorias": 480},
  {"nombre": "jitomates maduros", "cantidad": "4", "calorias": 100},
  {"nombre": "chiles pasilla secos", "cantidad": "2", "calorias": 50},
  {"nombre": "dientes de ajo", "cantidad": "2", "calorias": 10},
  {"nombre": "cebolla", "cantidad": "1/4", "calorias": 15},
  {"nombre": "caldo de pollo", "cantidad": "1 litro", "calorias": 60},
  {"nombre": "epazote", "cantidad": "1 ramita", "calorias": 2},
  {"nombre": "aguacate en cubos", "cantidad": "1", "calorias": 240},
  {"nombre": "queso fresco desmoronado", "cantidad": "100 g", "calorias": 250},
  {"nombre": "crema ácida", "cantidad": "100 ml", "calorias": 190},
  {"nombre": "Aceite vegetal para freir", "cantidad": "al gusto", "calorias": 120},
  {"nombre": "Sal", "cantidad": "al gusto", "calorias": 0},
  {"nombre": "Pimienta", "cantidad": "al gusto", "calorias": 0},
  {"nombre": "filetes de pescado blanco (merluza, bacalao o similar)", "cantidad": "4", "calorias": 400},
  {"nombre": "harina de trigo", "cantidad": "1 taza", "calorias": 400},
  {"nombre": "cerveza fria", "cantidad": "1 taza", "calorias": 150}.

```

```

promptR4 = f"""
Dada la siguiente lista de recetas, sugiere 3 vinos de mesa para hacer un maridaje, sugiriendo el nombre comercial y tipo de uva. Para el resultado genera un JSON con las tu
* Nombre del Vino
* Tipo de UVA

===
{Nombre_Recetas}
===

NOTA IMPORTANTE: Devuelve sólo el JSON sin texto adicional, preámbulos o las tildes ```json
ya que se usará directamente en python. No agregues etiquetas adicionales como parte del JSON, como "Sugerencias de Vinos", ingredientes, nombre de la receta o ningún otro.

"".strip()

print(promptR4)

```

```

Dada la siguiente lista de recetas, sugiere 3 vinos de mesa para hacer un maridaje, sugiriendo el nombre comercial y tipo de uva. Para el resultado genera un JSON con las tu
* Nombre del Vino
* Tipo de UVA

===
,Sopes Mexicanos,Sopa de tortilla,Fish & Chips,Sopes Mexicanos,Sopa de tortilla,Fish & Chips,Sopes Mexicanos,Sopa de tortilla,Fish & Chips
===

NOTA IMPORTANTE: Devuelve sólo el JSON sin texto adicional, preámbulos o las tildes ```json
ya que se usará directamente en python. No agregues etiquetas adicionales como parte del JSON, como "Sugerencias de Vinos", ingredientes, nombre de la receta o ningún otro.

```

```

responseR4 = client.responses.create(
    model="gpt-4o",
    input=promptR4
)

print(responseR4.output_text)

```

```

import json
import pandas

reportVinos = pandas.DataFrame(json.loads(responseR4.output_text), columns=["Nombre del Vino", "Tipo de UVA"])
reportVinos

```

	Nombre del Vino	Tipo de UVA
0	Viña Esmeralda	Moscatel
1	Marqués de Riscal Rueda	Verdejo
2	Kim Crawford	Sauvignon Blanc

```
#reportIngredientes = pandas.DataFrame(json.loads(respuesta3))
reportIngredientes = pandas.DataFrame(json.loads(respuesta3), columns=["nombre", "cantidad", "calorias", "costo"])

reportIngredientes
```

	nombre	cantidad	calorias	costo
0	masa de maíz	2 tazas	900	15
1	agua	1/2 taza	0	0
2	sal	al gusto	0	5
3	Aceite para freír	al gusto	240	20
4	frijoles refritos	1 taza	240	12
5	carne deshebrada (pollo, res o cerdo)	1 taza	250	30
6	lechuga picada	1/2 taza	5	5
7	crema ácida	150 ml	390	20
8	queso fresco desmoronado	150 g	450	25
9	Salsa	al gusto	20	10
10	tortillas de maíz	8	480	12
11	jitomates maduros	4	100	10
12	chiles pasilla secos	2	50	8
13	dientes de ajo	2	10	4
14	cebolla	1/4	15	3
15	caldo de pollo	1 litro	60	18
16	epazote	1 ramita	2	3
17	aguacate en cubos	1	240	20
18	Limón	al gusto	0	5


```
total_calorias = reportIngredientes['calorias'].sum()
print(f"Total calorias: {total_calorias}")

total_costo = reportIngredientes['costo'].sum()
print(f"Total costo estimado: {total_costo} MXN")
```

Total calorias: 4999
Total costo estimado: 362 MXN

RESULTADO

```
textoR1 = f"""

Muchas felicidades!!
A continuación veremos los detalles para tu evento

Platillos a servir:
===
{Nombre_Recetas}
===

Lista de Compras:

{reportIngredientes}

Costo Estimado:
~*{total_costo} MXN

Calorías del Menu:
~*{total_calorias}

===

Sugerencias de Vinos:
===
{reportVinos}
===
```



ANEXO DIAGRAMA

