

Automated email to customer message

**Using Generative AI concept,
openAI and chatGPT**

Prepared by: Kruti Dhyani

Table of content

1. Introduction
2. Design
3. Implementation
4. Test
5. Enhancement
6. Conclusion
7. Github repository Link:
8. Bibliography / References

Introduction

This project aims to streamline customer communication by automatically generating informative product-related emails, ensuring excellent customer support across various languages.

- **Automated Email Customer Support System:** This project centers around the automation of sending personalized emails to customers with product-related messages, leveraging ChatGPT's capabilities to create a web-based system.
- **Prerequisite:** It is recommended to first complete the "Customer Support System" project, where ChatGPT is used to develop a system for answering website-related questions, as this project builds upon that foundation.
- **Overview:** Imagine you are a customer service assistant at a major electronics store. The store's website offers various features, including language selection, an extensive product range categorized with detailed descriptions, and a web interface designed for this project.
- **Language Selection:** Customers can select their preferred language on the website, enhancing their experience.
- **Product Messaging:** The store's products are categorized into different segments, each with comprehensive descriptions. The objective of this project is to automate the process of sending customized product messages to customers via email.

Design

1

Generate a customer's comment

Input: Product's detailed description
[.jsonformat]
about the product
Output: 100 word comment

2

Generate email subject

Input: step:1 comment about product
output: chatGPT generates email subject
using inferring techniques

3

Generate summary based on comment

Input: comment + Language selection
output: Summary in english[summarization] +
translate in selected language [Transforming]

4

**Sentiment analysis of customer
comment**

Input: customer comment
Output: comment is positive or negative

5

**Generate an email to be sent to
customer**

Input: comment + summary + sentiment analysis
output: email with subject and content which
expanding the summary we get

Design view

Web interface which supports all the steps of the designing process. Like the below screenshot, which has language selection dropdown menu, automated message section, answer section and button which initiate the process.

Customer Support System_au/ x +

127.0.0.1:5000

11. Python Design... el Triggers in SQL Tu... 22 Design Patterns in... » | All Bookr

Choose the language of Email to be generated: Spanish

Question (Generated Comment):

I recently purchased the TechPro Ultrabook from this electronic product company and I am extremely satisfied with my purchase. The ultrabook is sleek and lightweight, making it perfect for everyday use. The 13.3-inch display provides a clear and vibrant viewing experience, and the 8GB RAM and 256GB SSD ensure smooth and fast performance. The Intel Core i5 processor is powerful enough to handle all my tasks efficiently. Additionally, the 1-year warranty gives me peace of mind. Overall, I highly recommend.

Answer (Generated Email):

Asunto: ¡Gracias por tus comentarios positivos sobre el TechPro Ultrabook!

Estimado cliente,

Nos complace enormemente escuchar tus comentarios y apreciamos tus palabras positivas. ¡Tu satisfacción es nuestra máxima prioridad!

Nos alegra saber que estás muy satisfecho con el

Implementation

- **Tools and TEchnology:** python Flask to run web applications, OpenAI secret key
- **Prerequisites:**
 - ◆ Create virtual environment or go to already created virtual environment using command:
workon <virtual environment name>
 - ◆ Download Flask and translate packages in your virtual environment. Using command:

```
bash: no job control in this shell
macs-MacBook-Pro:CS589 mac$ workon chatGPT
(chatgpt) macs-MacBook-Pro:CS589 mac$ pip install translate
Collecting translate
```

- Flask command: `pip install flask`
- Or you can combine install:

```
Successfully installed cffi-1.15.1 ctranslate2-2.11.1 cmake-4.0.3 translate-3.0.1
(chatgpt) macs-MacBook-Pro:CS589 mac$ pip install flask
Requirement already satisfied: flask in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (2.3.3)
Requirement already satisfied: Werkzeug>=2.3.7 in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (from flask) (2.3.7)
Requirement already satisfied: Jinja2>=3.1.2 in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (from flask) (3.1.2)
Requirement already satisfied: itsdangerous>=2.1.2 in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (from flask) (2.1.2)
Requirement already satisfied: click>=8.1.3 in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (from flask) (8.1.7)
Requirement already satisfied: blinker>=1.6.2 in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (from flask) (1.6.2)
Requirement already satisfied: MarkupSafe>=2.0 in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (from Jinja2>=3.1.2->flask) (2.1.3)
(chatgpt) macs-MacBook-Pro:CS589 mac$ pip install Flask openai translate
Requirement already satisfied: Flask in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (2.3.3)
Requirement already satisfied: openai in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (0.28.0)
Requirement already satisfied: translate in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (3.6.1)
Requirement already satisfied: Werkzeug>=2.3.7 in /Users/mac/.virtualenvs/chatgpt/lib/python3.10/site-packages (from Flask) (2.3.7)
```

- Create .env file in your project folder and set the variable `OPENAI_API_KEY= "your key"` and `ORG_ID = "your ID"`

Implementation: import libraries, data & set key

```
← → CS589
app_json.py 2 ×
emailToCustomer_automated > app_json.py > ...
1  # Works perfectly but slow without usinf Json file. Data is stored in .py file
2  import openai
3  import os
4  import time
5  from flask import Flask, render_template, request
6  from dotenv import load_dotenv
7  #from products_data import products_data
8  import json
9  import random
10
11  |
12  app = Flask(__name__)
13
14  # Load product data from product.json
15  with open('products_data.json', 'r') as json_file:
16  |   products_data = json.load(json_file)
17
18  load_dotenv()
19  openai.api_key = os.getenv("OPENAI_API_KEY")
20  if openai.api_key is None:
21  |   raise ValueError("API key not found in environment variables")
22
```

```
emailToCustomer_automated > {} products_data.json > ...
{
  "TechPro Ultrabook": {
    "name": "TechPro Ultrabook",
    "category": "Computers and Laptops",
    "brand": "TechPro",
    "model_number": "TP-UB100",
    "warranty": "1 year",
    "rating": 4.5,
    "features": ["13.3-inch display", "8GB RAM",
                "256GB SSD", "Intel Core i5 processor"],
    "description": "A sleek and lightweight ultrabook for everyday use.",
    "price": 799.99
  },
  "BlueWave Gaming Laptop": {
    "name": "BlueWave Gaming Laptop",
    "category": "Computers and Laptops",
    "brand": "BlueWave",
    "model_number": "BW-GL200",
    "warranty": "2 years",
    "rating": 4.7,
    "features": ["15.6-inch display", "16GB RAM",
                "512GB SSD", "NVIDIA GeForce RTX 3060"],
    "description": "A high-performance gaming laptop for an immersive experience.",
    "price": 1199.99
  },
  "PowerLite Convertible": {
    "name": "PowerLite Convertible",
    "category": "Computers and Laptops",
    "brand": "PowerLite",
    "model_number": "PL-CV300",
    "warranty": "1 year",
    "rating": 4.3,
    "features": ["14-inch touchscreen", "8GB RAM",
                "256GB SSD", "360-degree hinge"],
    "description": "A versatile convertible laptop with a responsive touchscreen.",
    "price": 699.99
  },
}
```

Implementation: Step wise

Step:1=>

```
# Step 1: Generate a Customer's Comment
def generate_customer_comment(products_data):
    prompt=f"""
    Assume that you are a customer to an electronic product company.
    Write a 100-word only comment about the products delimited by triple backticks in its own language.
    Products: ```{products_data}```
    """
    response = get_completion_with_rate_limit(prompt)
    return response
```

Step:2 =>

```
# Step 2: Generate Email Subject
def generate_email_subject(comment):
    prompt=f"""
    Assuming that you provide customer support for an electronic product company.
    Based on the customer comment delimited in triple backticks, suggest a short email subject to respond to the customer.
    Comment= ```{comment}```
    """
    response = get_completion_with_rate_limit(prompt)
    return response
```

Step:3=>

```
# Step 3: Generate Customer Comment Summary
def summarize_comment(comment):
    prompt=f"""
    Assuming that you provide customer support for an electronic product company.
    Provide a concise summary in 50 words of the following customer comment delimited in triple backticks. Comment: ```{comment}```
    """
    response = get_completion_with_rate_limit(prompt)
    return response

def translate_summary_with_chatgpt(language, summary):
    prompt= f"""
    Translate the following summary delimited by triple backticks to the language delimited by <>.
    Language:```{language}```
    Summary:<{summary}>
    """
    response = get_completion_with_rate_limit(prompt)
    return response
```


Implementation: Step wise

Step:4=>

```
# Step 4: Analyze Customer Comment Sentiment
def analyze_sentiment(comment):
    prompt=f"""
    Assuming that you provide customer support for an electronic product company.
    What is the sentiment of the comment delimited in triple backticks? Is it positive or negative?
    Comment: ```{comment}```
    """
    max_tokens=10
    response = get_completion_with_rate_limit(prompt)
    sentiment = response.lower()
    if "positive" in sentiment:
        return "positive"
    elif "negative" in sentiment:
        return "negative"
    else:
        return "neutral"
```

Step:5 =>

```
# Step 5: Generate Customer Email
def generate_customer_email(summary, sentiment, email_subject, language):
    if sentiment == "positive":
        response_text = "We're thrilled to hear your feedback and appreciate your positive words. Your satisfaction is our top priority!"
    elif sentiment == "negative":
        response_text = "We're truly sorry to hear about your experience. Your feedback is crucial, and we'll strive to address your concerns."
    else:
        response_text = "Thank you for your feedback! We're always looking to improve, and your insights are valuable."
    prompt = f"""
    Assuming that you provide customer support for an electronic product company.
    Given the specified parameters below:
    - Comment summary enclosed in backticks (`{summary}`)
    - Our response text enclosed in triple quotes (`\"{response_text}\"`)
    - Translate the Email subject enclosed in angle brackets (<{email_subject}>) to language \"{language}\"
    Write a complete email responding to the customer's comment using the language \"{language}\".
    """
    response = get_completion_with_rate_limit(prompt)
    return response
```

Implementation: main

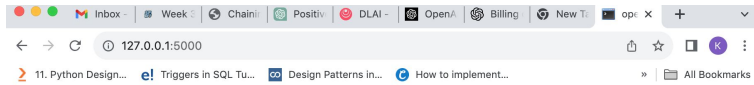
```
@app.route('/', methods=['GET', 'POST'])
def index():
    answer = ""
    comment = generate_customer_comment(products_data)
    print("A customer comment has been generated.")
    if request.method == 'POST':
        language = request.form.get('language') # Fetch language input from the webpage
        print(f"Selected language: {language}")
        answer = process_comment_to_email(comment, language)
    return render_template('index.html', question=comment, answer=answer)

def process_comment_to_email(comment, language):
    # Step 2: Generate Email Subject
    email_subject = generate_email_subject(comment)
    print(f"An email subject is generated from the customer's comment.")
    # Step 3: Generate Customer Comment Summary
    summary = summarize_comment(comment)
    print("A Summary is generated from the customer comment.")
    translated_summary = translate_summary_with_chatgpt(language, summary)
    print("The summary has been translated to the requested language.")
    # Step 4: Analyze Customer Comment Sentiment
    comment_sentiment = analyze_sentiment(comment)
    print(f"Sentiment of the comment is detected as: {comment_sentiment}")
    # Step 5: Generate Customer Email
    email_content = generate_customer_email(translated_summary, comment_sentiment, email_subject, language)
    print("A customer email has been generated.")
    return email_content

if __name__ == '__main__':
    app.run(debug=True)
```

Error and Error Solution

While running the app it Throws 'RateLimitError' which is not the coding error but openAI do not allow you to run continuously so following is error screenshot and I added line of code to my file to solve the error.



RateLimitError

openai.error.RateLimitError: Rate limit reached for default-gpt-3.5-turbo in organization org-WoEf03htip07Ed0Mwd5X1M1w on requests per min. Limit: 3 / min. Please try again in 20s. Contact us through our help center at help.openai.com if you continue to have issues. Please add a payment method to your account to increase your rate limit. Visit <https://platform.openai.com/account/billing> to add a payment method.

Traceback (most recent call last)

```
File "/Users/mac/virtualenvs/chatgpt/lib/python3.10/site-packages/lask/app.py", line 2213, in __call__
    return self.wsgi_app(environ, start_response)

File "/Users/mac/virtualenvs/chatgpt/lib/python3.10/site-packages/lask/app.py", line 2193, in wsgi_app
    response = self.handle_exception(e)

File "/Users/mac/virtualenvs/chatgpt/lib/python3.10/site-packages/lask/app.py", line 2190, in wsgi_app
    response = self.full_dispatch_request()

File "/Users/mac/virtualenvs/chatgpt/lib/python3.10/site-packages/lask/app.py", line 1486, in full_dispatch_request
    rv = self.handle_user_exception(e)

File "/Users/mac/virtualenvs/chatgpt/lib/python3.10/site-packages/lask/app.py", line 1484, in full_dispatch_request
    rv = self.dispatch_request()

File "/Users/mac/virtualenvs/chatgpt/lib/python3.10/site-packages/lask/app.py", line 1469, in dispatch_request
    return self.ensure_sync(self.view_functions[rule.endpoint])(**view_args)

File "/Users/mac/rutli/CS589/emailToCustomer_automated/app.py", line 127, in index
    answer = process_comment_to_email(comment, language)

File "/Users/mac/rutli/CS589/emailToCustomer_automated/app.py", line 137, in process_comment_to_email
```

```
# Define a rate limiting mechanism
RATE_LIMIT = 3 # Requests per minute
RATE_LIMIT_PERIOD = 60 # 60 seconds in a minute

# Define a timestamp variable to track the last request time
last_request_time = 0

def get_completion_with_rate_limit(prompt):
    global last_request_time

    # Calculate the time since the last request
    current_time = time.time()
    time_since_last_request = current_time - last_request_time

    # Check if the rate limit has been reached, and if so, wait
    if time_since_last_request < RATE_LIMIT_PERIOD:
        time_to_wait = RATE_LIMIT_PERIOD - time_since_last_request
        time.sleep(time_to_wait)

    # Make the API request
    messages = [{"role": "user", "content": prompt}]
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=messages,
        temperature=0,
    )

    last_request_time = time.time()

    return response.choices[0].message["content"]
# response = openai.ChatCompletion.create(
```

Executing project

To run the code write `python3 <filename>.py` and server is running on port 5000.

Open browser and type <http://127.0.0.1:5000> to use the project.

```
PROBLEMS 4 OUTPUT TERMINAL PORTS POSTMAN CONSOLE DEBUG CONSOLE bash - emailToCu
(chatgpt) macs-MacBook-Pro:emailToCustomer_automated mac$ python3 app1.py
  * Serving Flask app 'app1'
  * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
  * Running on http://127.0.0.1:5000
Press CTRL+C to quit
  * Restarting with stat
  * Debugger is active!
  * Debugger PIN: 760-560-365
A customer comment has been generated.
127.0.0.1 - - [04/Oct/2023 00:18:48] "GET / HTTP/1.1" 200 -
A customer comment has been generated.
Selected language: en
An email subject is generated from the customer's comment.
A Summary is generated from the customer comment.
The summary has been translated to the requested language.
Sentiment of the comment is detected as: positive
A customer email has been generated.
127.0.0.1 - - [04/Oct/2023 00:25:05] "POST / HTTP/1.1" 200 -
□
```

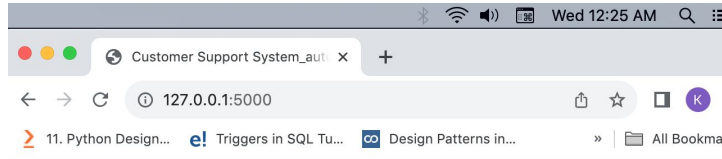
Test Cases

Expected output:

ID	Question	Answer
1	English	English
2	English	Non-English
3	Non-English	English
4	Non-English	Non-English
		<p>Note:</p> <ul style="list-style-type: none">▪ ChatGPT can infer the language used in Question and then generate the Answer with the same language

Test cases output for all combinations:

1) English to English



Choose the language of Email to be generated: English

Question (Generated Comment):

I recently purchased the TechPro Ultrabook from this electronic product company and I am extremely satisfied with my purchase. The ultrabook is sleek and lightweight, making it perfect for everyday use. The 13.3-inch display provides a clear and vibrant viewing experience, and the 8GB RAM and 256GB SSD ensure smooth and fast performance. The Intel Core i5 processor is powerful enough to handle all my tasks efficiently. Additionally, the 1-year warranty gives me peace of mind. Overall, I highly recommend

Answer (Generated Email):

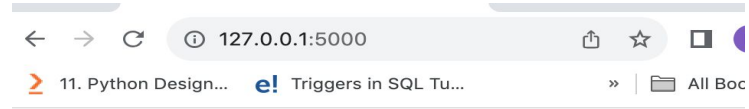
Subject: Thank you for your positive feedback on the TechPro Ultrabook!

Dear [Customer's Name],

Thank you for taking the time to share your positive feedback regarding your recent purchase of the TechPro Ultrabook. We are delighted to hear that you are highly satisfied with its elegant design, lightweight nature, clear 13.3-inch display, fast

Process

2)spanish to spanish



Choose the language of Email to be generated: Spanish

Question (Generated Comment):

TechPro Ultrabook: Un ultrabook élégant et léger pour une utilisation quotidienne.
BlueWave Gaming Laptop: Un ordinateur portable de jeu performant pour une expérience immersive.
PowerLite Convertible: Un ordinateur portable convertible polyvalent avec un écran tactile réactif.
TechPro Desktop: Un ordinateur de bureau puissant pour le travail et les loisirs.
BlueWave Chromebook: Un Chromebook compact et

Answer (Generated Email):

Estimado cliente,

¡Gracias por su mensaje! Nos alegra escuchar sus comentarios y apreciamos sus palabras positivas. ¡Su satisfacción es nuestra prioridad!

Hemos recibido su comentario sobre nuestros productos electrónicos y nos complace proporcionarle más detalles sobre cada uno de ellos. A continuación, encontrará una descripción de cada

Process

Test cases output for all combinations:

3) English to Spanish

The screenshot shows a web browser window with the title "Customer Support System_auti". The address bar shows "127.0.0.1:5000". The page has a header with navigation links: "11. Python Design...", "e! Triggers in SQL Tu...", "Design Patterns in...", and "All Book". Below the header, there is a dropdown menu labeled "Choose the language of Email to be generated:" with "Spanish" selected. The main content area is divided into two sections: "Question (Generated Comment):" and "Answer (Generated Email):". The "Question" section contains a text box with the following text: "I recently purchased the TechPro Ultrabook from this electronic product company and I am extremely satisfied with my purchase. The ultrabook is sleek and lightweight, making it perfect for everyday use. The 13.3-inch display provides a clear and vibrant viewing experience, and the 8GB RAM and 256GB SSD ensure smooth and fast performance. The Intel Core i5 processor is powerful enough to handle all my tasks efficiently. Additionally, the 1-year warranty gives me peace of mind. Overall, I highly recommend". The "Answer" section contains a text box with the following text: "Asunto: ¡Gracias por tus comentarios positivos sobre el TechPro Ultrabook! Estimado cliente, Nos complace enormemente escuchar tus comentarios y apreciamos tus palabras positivas. ¡Tu satisfacción es nuestra máxima prioridad! Nos alegra saber que estás muy satisfecho con el". Below the text boxes, there is a "Process" button.

Customer Support System_auti x +

127.0.0.1:5000

11. Python Design... e! Triggers in SQL Tu... Design Patterns in... » All Book

Choose the language of Email to be generated: Spanish

Question (Generated Comment):

I recently purchased the TechPro Ultrabook from this electronic product company and I am extremely satisfied with my purchase. The ultrabook is sleek and lightweight, making it perfect for everyday use. The 13.3-inch display provides a clear and vibrant viewing experience, and the 8GB RAM and 256GB SSD ensure smooth and fast performance. The Intel Core i5 processor is powerful enough to handle all my tasks efficiently. Additionally, the 1-year warranty gives me peace of mind. Overall, I highly recommend

Answer (Generated Email):

Asunto: ¡Gracias por tus comentarios positivos sobre el TechPro Ultrabook!

Estimado cliente,

Nos complace enormemente escuchar tus comentarios y apreciamos tus palabras positivas. ¡Tu satisfacción es nuestra máxima prioridad!

Nos alegra saber que estás muy satisfecho con el

Process

4) spanish to english

The screenshot shows a web browser window with the title "Customer Support System_auti". The address bar shows "127.0.0.1:5000". The page has a header with navigation links: "11. Python Design...", "e! Triggers in SQL Tu...", "Design Patterns in...", and "All Book". Below the header, there is a dropdown menu labeled "Choose the language of Email to be generated:" with "English" selected. The main content area is divided into two sections: "Question (Generated Comment):" and "Answer (Generated Email):". The "Question" section contains a text box with the following text: "Les produits de cette entreprise sont vraiment incroyables! J'ai récemment acheté le TechPro Ultrabook et je suis très satisfait de ses performances. C'est un ultrabook élégant et léger, idéal pour une utilisation quotidienne. De plus, le BlueWave Gaming Laptop offre une expérience immersive de jeu avec sa puissance de jeu élevée. Le PowerLite Convertible est également très polyvalent avec son écran tactile réactif. Je suis également impressionné par la qualité des smartphones,". The "Answer" section contains a text box with the following text: "Subject: Thank you for your wonderful review of our products! Dear [Customer's Name], We hope this email finds you well. We wanted to personally express our gratitude for your recent comment about our electronic products. Your satisfaction means the world to us and we're delighted to hear that you're extremely happy with". Below the text boxes, there is a "Process" button.

Customer Support System_auti x +

127.0.0.1:5000

11. Python Design... e! Triggers in SQL Tu... Design Patterns in... » All Book

Choose the language of Email to be generated: English

Question (Generated Comment):

Les produits de cette entreprise sont vraiment incroyables! J'ai récemment acheté le TechPro Ultrabook et je suis très satisfait de ses performances. C'est un ultrabook élégant et léger, idéal pour une utilisation quotidienne. De plus, le BlueWave Gaming Laptop offre une expérience immersive de jeu avec sa puissance de jeu élevée. Le PowerLite Convertible est également très polyvalent avec son écran tactile réactif. Je suis également impressionné par la qualité des smartphones,

Answer (Generated Email):

Subject: Thank you for your wonderful review of our products!

Dear [Customer's Name],

We hope this email finds you well. We wanted to personally express our gratitude for your recent comment about our electronic products. Your satisfaction means the world to us and we're delighted to hear that you're extremely happy with

Process

Enhancements

1. Works as a translator app when person do not know the word or sentence meaning.
2. Integrate with customer Relationship management to maintain a comprehensive record of customer interactions, enabling better tracking and follow-up.
3. Integrate a chatbot alongside ChatGPT to provide immediate responses to common customer queries.
4. Implement real-time analytics to monitor customer interactions and sentiments. This data can be used to further improve the support system by identifying trends and areas where customers need assistance the most.

Conclusion

The Automated Email Customer Support System utilizes ChatGPT to automate personalized product email messages for enhanced customer support. With multi-language support, sentiment analysis, and future enhancements, it promises to elevate customer interactions and satisfaction. This project marks a significant step towards efficient, data-driven, and customer-centric support within a large electronics store.

Github Links:

<https://github.com/DKruti/Machine-Learning/tree/master/Generative%20AI/Automated%20Email%20to%20customer>

References

- https://hc.labnet.sfbu.edu/~henry/sfbu/course/deeplearning_ai/chatgpt_prompt_eng_for_developer/slide/exercise_chatgpt_prompt_eng_for_developer.html
- <https://learn.deeplearning.ai/chatgpt-prompt-eng/lesson/2/guidelines>