Customer Support System

An Email to the Customer

K. Sri Vardhan

Index

- Introduction
- Design
- Implementation
- Test
- Enhancement Ideas
- Conclusion
- References

Introduction

The Customer Support System is a web application that utilizes the OpenAI GPT-3.5 Turbo model to provide customer support for an electronic product company. This system generates customer comments, email subjects, comment summaries, and email responses based on user interactions. It offers support in multiple languages, including English, Spanish, and Portuguese.

Design

The system is designed as a Flask web application with a straightforward user interface. Users can select the language in which they want to generate email responses and view the generated customer comment and email response. The application follows these key design components:

- Flask Web Application: The application is built using Flask, a lightweight web framework in Python, to handle user interactions.
- OpenAI Integration: It integrates with the OpenAI GPT-3.5 Turbo model to generate text-based responses for customer support tasks.
- HTML Template: The user interface is designed using an HTML template (index.html) with a language dropdown menu, question, and answer containers.

Design

Step	Description	Input to ChatGPT	ChatGPT's Response
1	Generate Customer's Comment	Detailed product descriptions	A 100-word comment about the products
2	Generate Email Subject	Customer's comment from Step 1	Email subject inferred from the customer's comment
3	Generate Comment Summary	Customer's comment from Step 1, Customer's selected language	**3.1:** Summary in English using Summarization **3.2:** Translation to the selected language
4	Analyze Customer Comment Sentiment	Customer's comment from Step 1	Sentiment analysis (Positive/Negative)
5	Generate Customer Email	Customer's comment from Step 1 Comment summary from Step 3.2 Sentiment result	An email response to the customer

Implementation

To run project on an Ubuntu System:

1. Create and activate the virtual environment:

```
chinni@LAPTOP-UVNKR98P:~/W3$ python3 -m venv venv
chinni@LAPTOP-UVNKR98P:~/W3$ . venv/bin/activate
(venv) chinni@LAPTOP-UVNKR98P:~/W3$
```

2. Install the required Python packages:

```
(venv) chinni@LAPTOP-UVNKR98P:~/W3$ pip install flask openai python-dotenv
Collecting flask
  Using cached flask-3.0.0-py3-none-any.whl (99 kB)
Collecting openai
  Using cached openai-0.28.1-py3-none-any.whl (76 kB)
Collecting python-dotenv
  Downloading python_dotenv-1.0.0-py3-none-any.whl (19 kB)
```

Implementation

3. Start the Flask application:

```
(venv) chinni@LAPTOP-UVNKR98P:~/W3$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a pr
d.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 477-038-285
```

Once the application is running, open a web browser and navigate to `http://localhost:5000`.

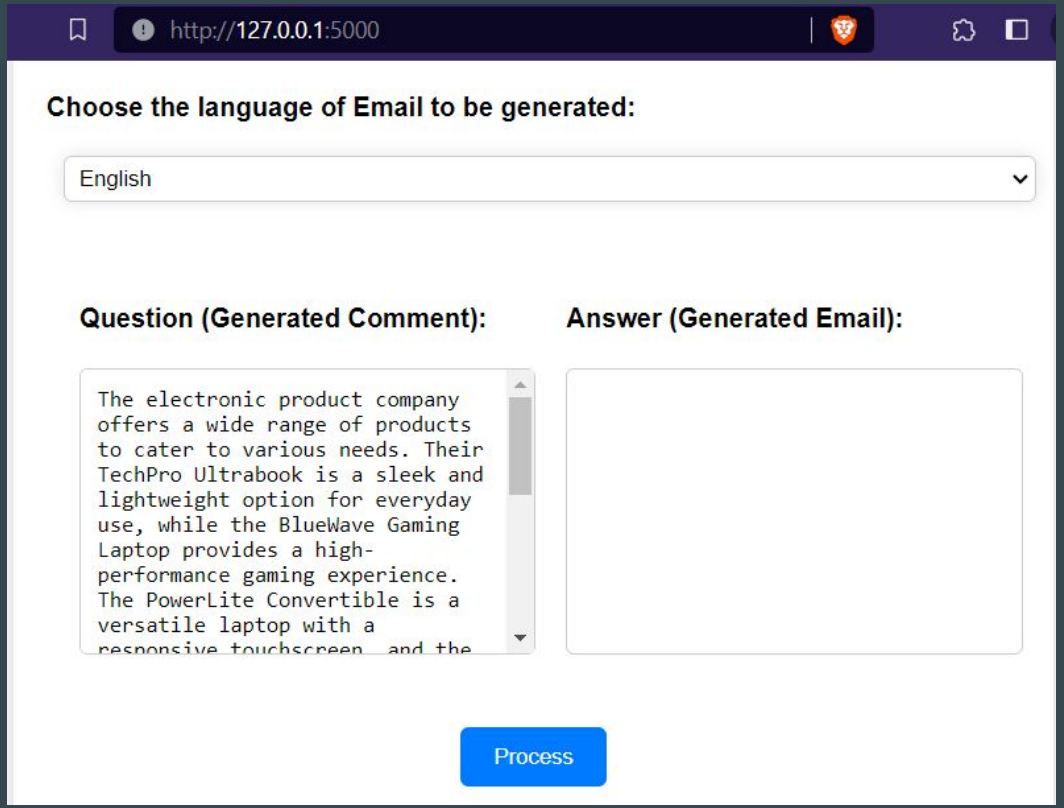
```
127.0.0.1 - - [02/Oct/2023 18:42:38] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [02/Oct/2023 18:42:38] "GET /static/styles.css HTTP/1.1" 304 -
A customer comment has been generated.
Selected language: en
An email subject is generated from the customer's comment.
A Summary is generated from the customer comment.
The summary has been translated to requested language.
Sentiment of the comment is detected as: positive
A customer email has been generated.
```

Test #1 (English Input)

Firstly, I used the English input

(products' description) in my flask application.

So, the comment generated is:

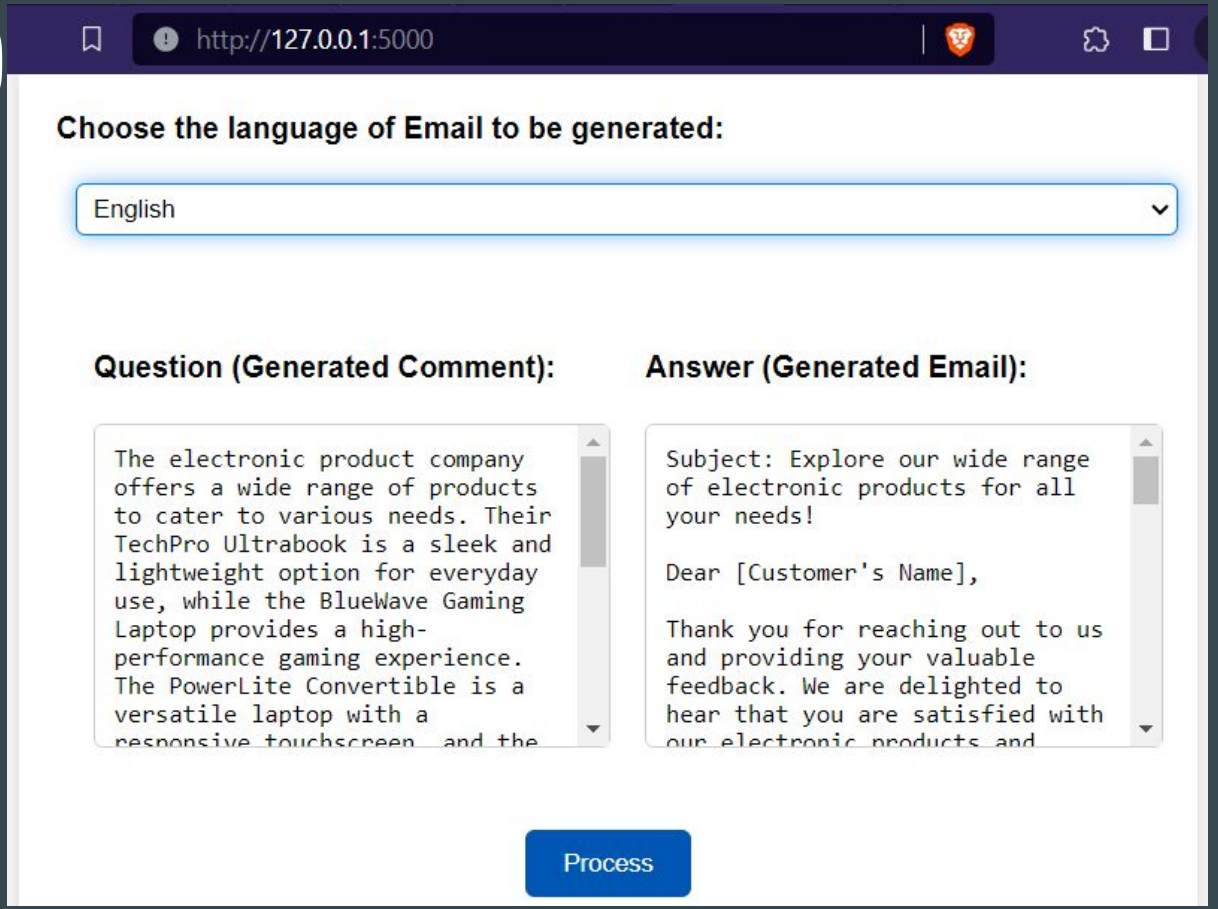


Test #1.1 (English Input)

I selected English from the dropdown

And clicked on process button.

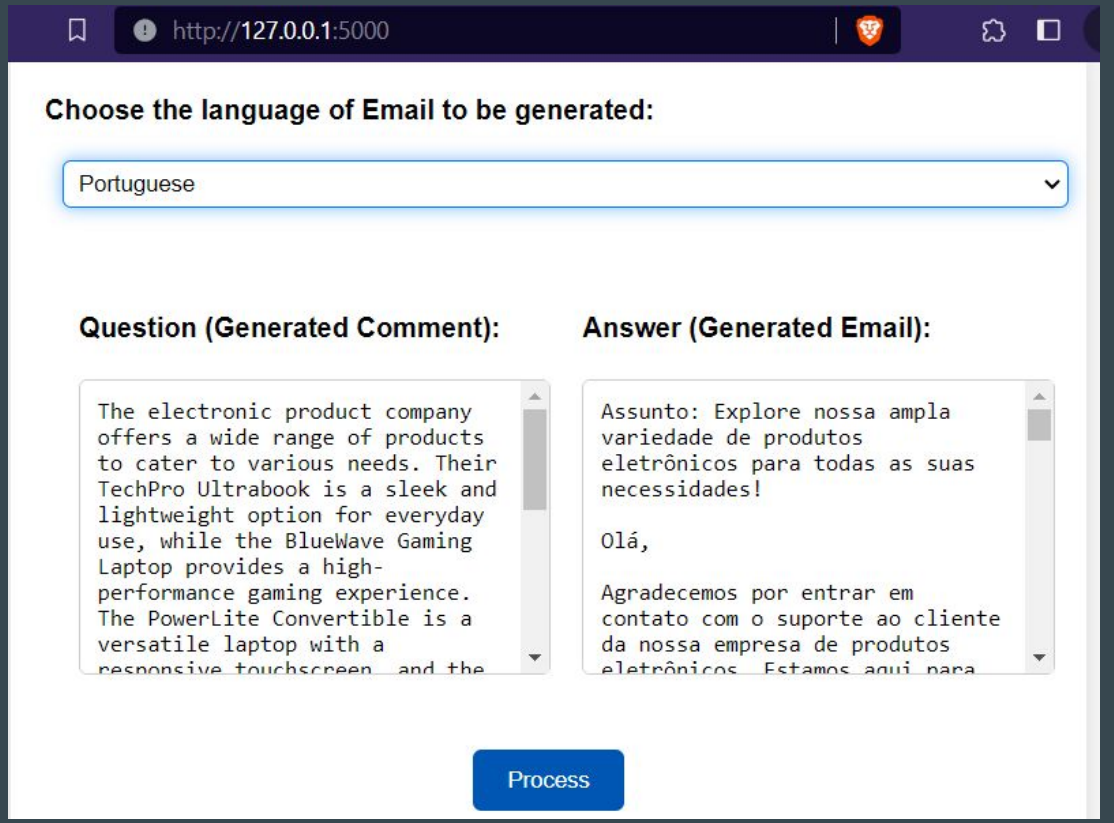
This generated the Email in English using the comment.



Test #1.2 (English Input)

Now, I selected portuguese from the dropdown and clicked on process button.

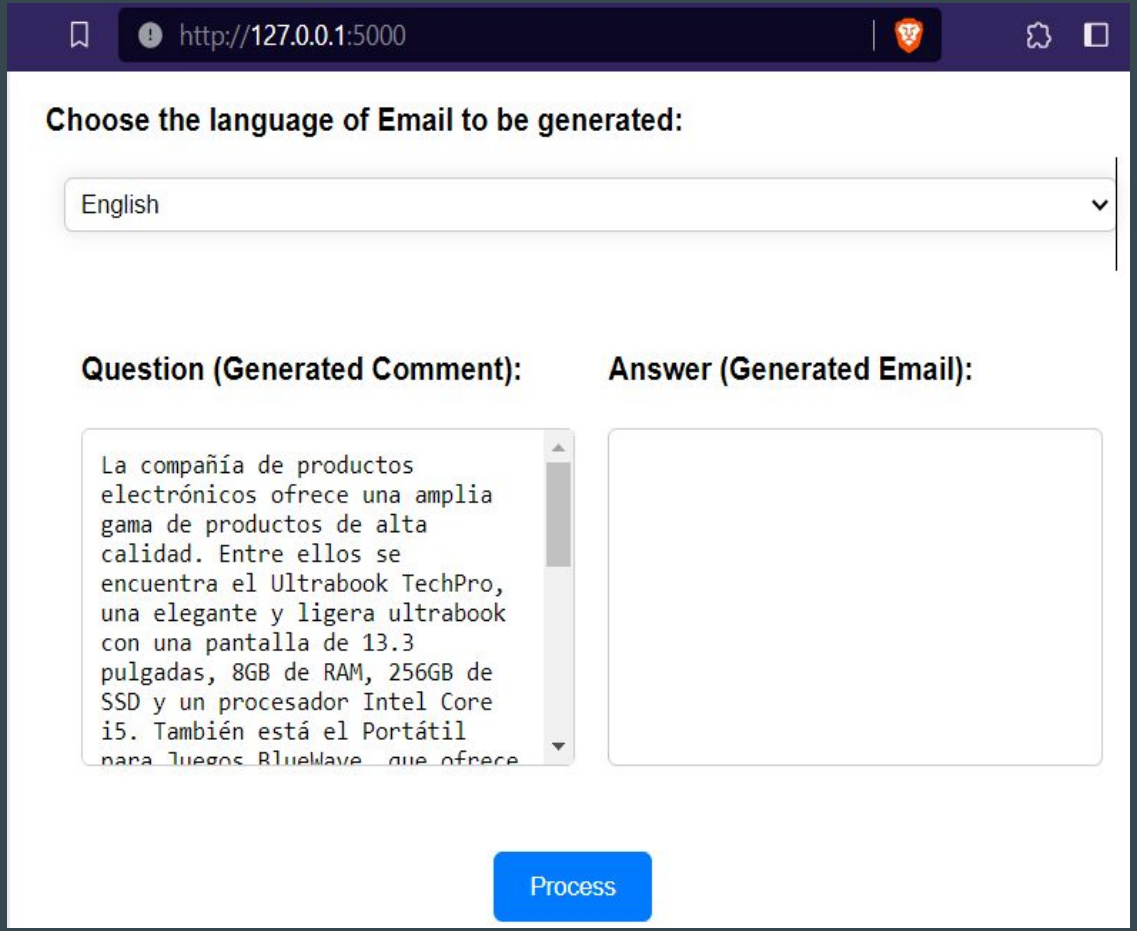
This translated the english comment to portuguese and generated an email with subject in portugese



Test #2 (Spanish Input)

Then,
I changed the input(products' description) to input, which is in spanish, in my flask application

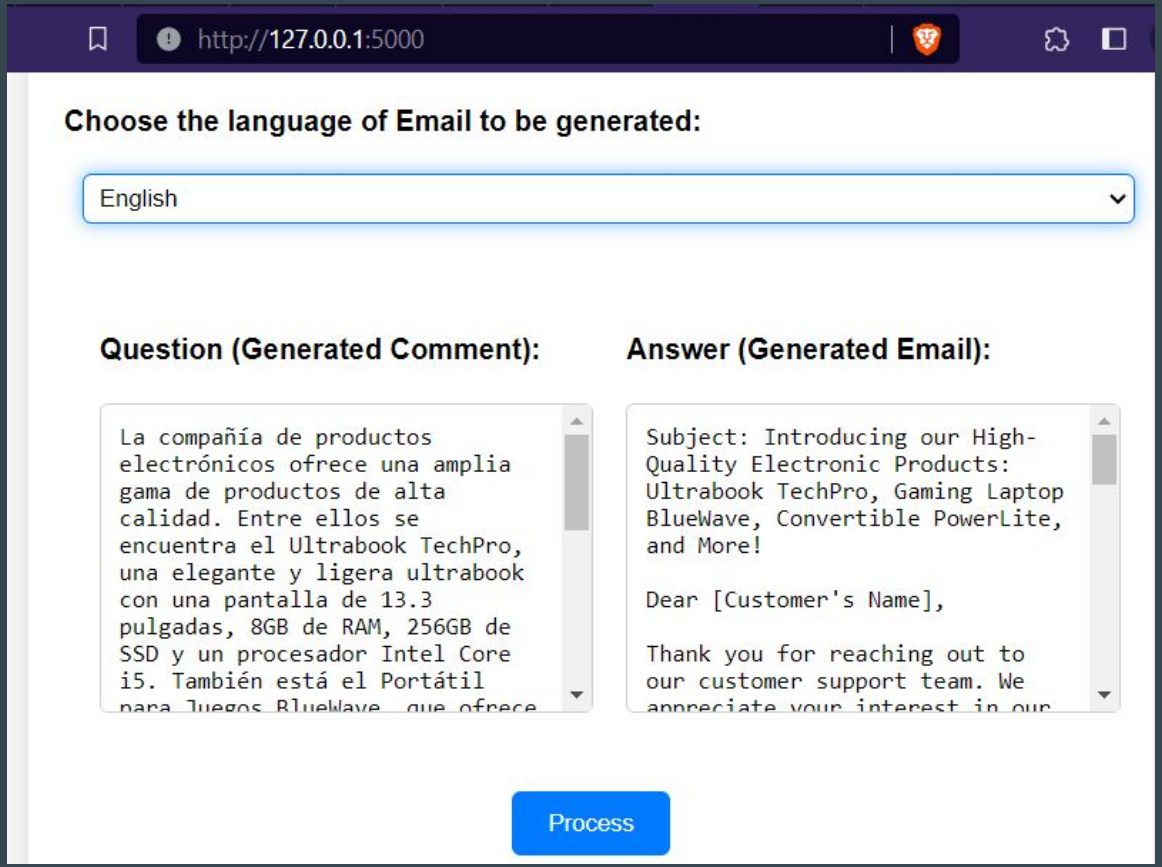
The comment is generated in spanish.



Test #2.1 (Spanish Input)

I chose the English language from the dropdown menu.

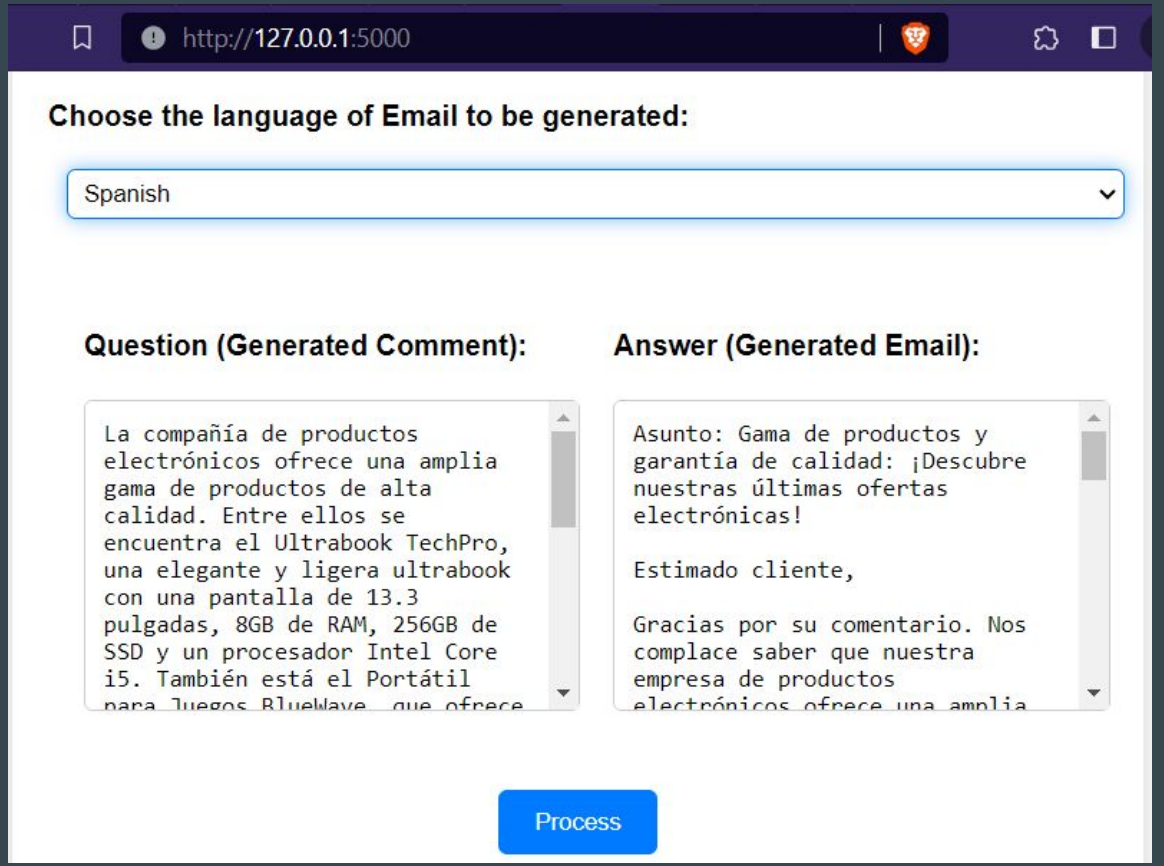
When clicked on process button, an email with subject is generated in English from the spanish comment.



Test #2.2 (Spanish Input)

I selected spanish from the dropdown menu.

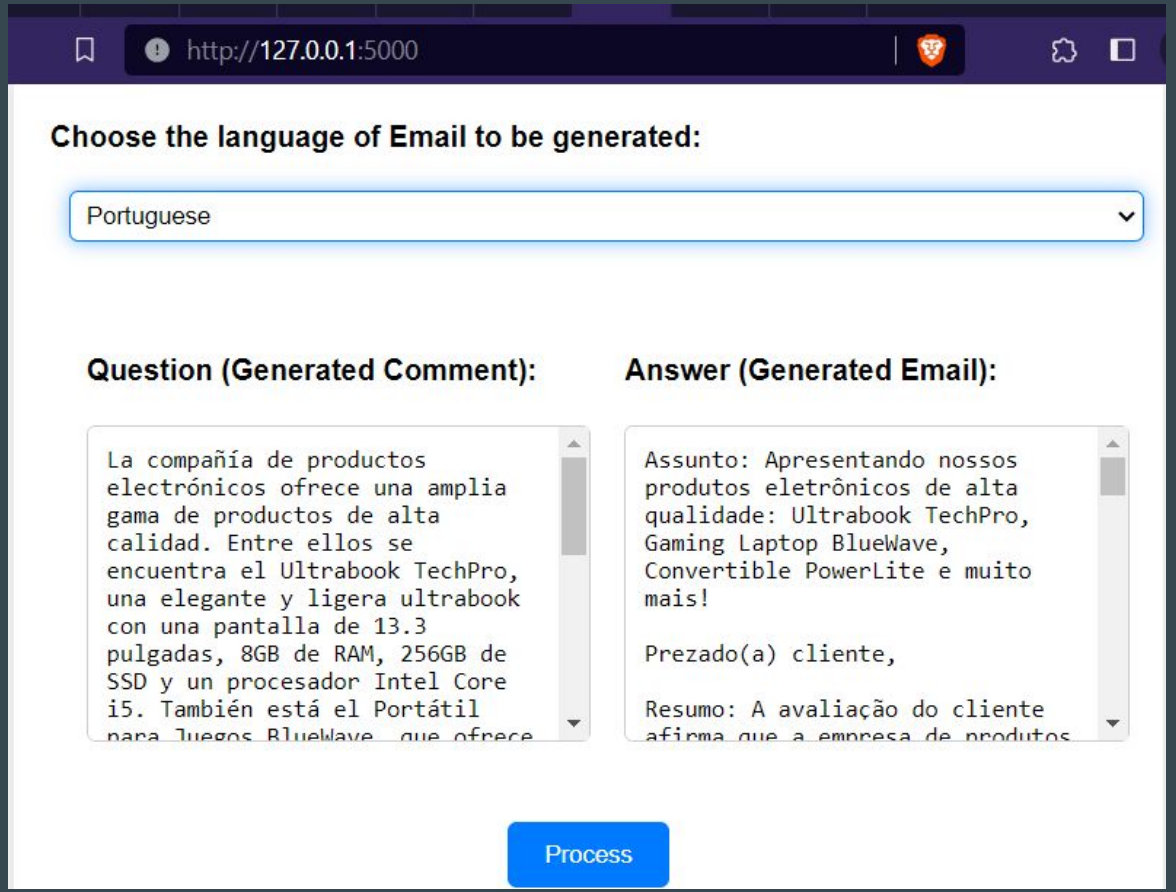
When clicked on process button, an email with subject is generated in spanish from spanish comment.



Test #2.3 (Spanish Input)

Finally, I tested selected portuguese.

It generated Portuguese email with subject from a spanish comment, when process button is clicked.



Enhancement Ideas

- **Multilingual Support:** Extend language support to include more languages, catering to a broader audience. This can be especially useful for companies with a global customer base.
- **Customizable Email Templates:** Allow users to customize the email templates generated by the system. This customization can include adding company branding, signatures, and more personalized responses.
- **User Feedback Mechanism:** Implement a feedback mechanism where users can rate the generated email responses. This feedback can be used to improve the quality of responses over time.
- **Integration with CRM Systems:** Integrate the system with Customer Relationship Management (CRM) systems to log and track customer interactions automatically. This enhances the efficiency of customer support operations.

Conclusion

In conclusion, the Customer Support System represents a powerful fusion of cutting-edge technology and user-centric design, aimed at revolutionizing the way companies interact with their customers. Through the creative application of OpenAI's GPT-3.5 Turbo model and the intuitive Flask web interface, we've created a versatile platform that empowers businesses to deliver exceptional customer support experiences.

As we've embarked on this journey, we've witnessed the potential of AI to seamlessly bridge language barriers, generate contextually relevant responses, and assist customers across diverse linguistic backgrounds. This project has been an exploration of the limitless possibilities that emerge when human ingenuity meets artificial intelligence.

References

- Professor Chang's material
- ChatGPT Prompt Engineering for Developers