***EKSPERIMEN PENGETAHUAN DAN WAWASAN   
Membangun Sistem Chatbot Pelayanan Pelanggan Otomatis untuk Website E-commerce dengan ChatGPT***

*Laporan ini disusun untuk memenuhi Tugas Mata Kuliah Pengembangan Web*



**Disusun Oleh:**

Arnanda Prasatya 221524034

Bhisma Chandra Yudha Setiawan 221524037

Jeihan Ilham Kusumawardhana 221524042

Keanu Rayhan Harits 221524043

Mochamad Fathur Rabbani 221524045

Muhammad Hanif 221524050

Reza Maulana Aziz 221524057

**Mata Kuliah :**

Nama Mata Kuliah : Pengembangan Web

Kode Mata Kuliah : 21TI3005

**Dosen Pengampu:**

Djoko Cahyo Utomo Lieharyani, S.Kom., M.MT

**PROGRAM STUDI D4 TEKNIK INFORMATIKA**

**JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA**

**POLITEKNIK NEGERI BANDUNG**

**2024**

Contents

[Methodology 3](#_Toc176116612)

[Environment 4](#_Toc176116613)

[Source Code 5](#_Toc176116614)

# Methodology

Membangun Sistem Chatbot Pelayanan Pelanggan Otomatis untuk Website E-commerce dengan ChatGPT" berarti membuat sebuah chatbot berbasis AI untuk menyediakan dukungan pelanggan secara otomatis di situs web e-commerce dengan menggunakan teknologi dari ChatGPT, yang merupakan model bahasa yang dikembangkan oleh OpenAI.

**Penjelasan Terperinci**

1. **Sistem Chatbot**:
   * **Definisi**: Chatbot adalah program perangkat lunak yang dirancang untuk mensimulasikan percakapan dengan pengguna manusia, baik melalui teks atau suara.
   * **Fungsi**: Dalam konteks e-commerce, chatbot dapat membantu menjawab pertanyaan pelanggan, memberikan informasi produk, menangani keluhan, dan memberikan bantuan secara umum.
2. **Pelayanan Pelanggan Otomatis**:
   * **Definisi**: Ini berarti bahwa chatbot akan melakukan interaksi dengan pelanggan secara otomatis tanpa memerlukan intervensi manusia secara langsung.
   * **Tujuan**: Meningkatkan efisiensi layanan pelanggan dengan memberikan jawaban cepat dan akurat atas pertanyaan pelanggan, serta menangani permintaan dasar secara mandiri.
3. **Website E-commerce**:
   * **Definisi**: Situs web yang menjual produk atau layanan secara online.
   * **Integrasi**: Chatbot akan diintegrasikan ke dalam situs web e-commerce untuk memberikan pengalaman pengguna yang lebih baik dan meningkatkan kepuasan pelanggan.
4. **ChatGPT**:
   * **Definisi**: ChatGPT adalah model bahasa yang dikembangkan oleh OpenAI, yang mampu menghasilkan teks yang mirip dengan manusia berdasarkan input yang diberikan. Ini bisa digunakan untuk mengembangkan chatbot yang dapat berkomunikasi secara alami dan efektif dengan pengguna.

# Environment

Untuk membangun dan menjalankan sistem chatbot pelayanan pelanggan otomatis menggunakan ChatGPT di situs web e-commerce, kamu perlu menyiapkan beberapa komponen dan lingkungan pengembangan. Berikut adalah rincian lingkungan yang diperlukan:

**\*\*1. Lingkungan Pengembangan**

* **Komputer/Server**:
  + Komputer atau server tempat kamu akan mengembangkan dan menjalankan aplikasi. Bisa berupa laptop, desktop, atau server cloud.
* **Sistem Operasi**:
  + Windows, macOS, atau Linux. Pastikan sistem operasi yang digunakan mendukung perangkat lunak yang diperlukan.

**\*\*2. Perangkat Lunak dan Alat**

* **Python**:
  + Bahasa pemrograman yang digunakan untuk membangun backend aplikasi. Pastikan Python versi terbaru terinstal.
* **Virtual Environment (Env)**:
  + Gunakan virtual environment seperti venv atau virtualenv untuk mengelola dependensi proyek secara terpisah dari sistem Python utama.
  + **Command**: python -m venv myenv
* **Flask**:
  + Framework web Python untuk membuat aplikasi web yang akan menghubungkan chatbot dengan antarmuka pengguna di situs web e-commerce.
  + **Install**: pip install Flask
* **OpenAI Python Library**:
  + Library yang digunakan untuk berinteraksi dengan API OpenAI.
  + **Install**: pip install openai
* **Editor Kode**:
  + Alat untuk menulis dan mengedit kode seperti Visual Studio Code, PyCharm, atau editor teks lainnya.
* **Web Browser**:
  + Untuk menguji dan memverifikasi aplikasi web. Browser seperti Chrome, Firefox, atau Edge dapat digunakan.

**\*\*3. API dan Layanan**

* **OpenAI API Key**:
  + Kamu memerlukan kunci API dari OpenAI untuk menggunakan ChatGPT. Kamu dapat mendaftar dan mendapatkan kunci API di [OpenAI Platform](https://platform.openai.com/account/api-keys).
* **Hosting (Opsional)**:
  + Jika kamu ingin mempublikasikan aplikasi ke internet, kamu akan memerlukan layanan hosting seperti Heroku, AWS, Google Cloud Platform, atau penyedia hosting lain.

**\*\*4. Database (Opsional)**

* **Database**:
  + Jika aplikasi memerlukan penyimpanan data (misalnya, log chat atau data pengguna), kamu akan memerlukan database seperti SQLite, PostgreSQL, MySQL, atau MongoDB.

**\*\*5. Alat Pengujian**

* **Alat Pengujian API**:
  + Alat seperti Postman untuk menguji endpoint API dan memastikan mereka berfungsi dengan benar.
* **Alat Debugging**:
  + Alat bawaan dari IDE atau alat tambahan untuk debugging dan pemantauan.

# Source Code

App.py

A screenshot of a computer

Description automatically generated

Index.html

A screenshot of a computer screen

Description automatically generated

# Experiment

1. **Instalasi Python**
   * Pastikan Python terinstal di sistemmu. Kamu bisa mengunduhnya dari [situs resmi Python](https://www.python.org/downloads/).
2. **Membuat Virtual Environment**
   * Buka terminal atau Command Prompt dan buat virtual environment:

bash

Salin kode

python -m venv myenv

* + Aktifkan virtual environment:

bash

Salin kode

# Windows

myenv\Scripts\activate

# Linux/macOS

source myenv/bin/activate

1. **Instalasi Dependensi**
   * Instal Flask dan OpenAI API client:

bash

Salin kode

pip install flask openai

**2. Pengembangan Backend**

1. **Membuat File app.py**
   * Buat file app.py dengan kode berikut:

python

Salin kode

from flask import Flask, request, jsonify

import openai

app = Flask(\_\_name\_\_)

openai.api\_key = 'your-openai-api-key' # Ganti dengan API key OpenAI

def get\_chatbot\_response(user\_input):

response = openai.ChatCompletion.create(

model="gpt-4",

messages=[

{"role": "system", "content": "You are a helpful assistant for an e-commerce website."},

{"role": "user", "content": user\_input}

]

)

return response['choices'][0]['message']['content']

@app.route('/chat', methods=['POST'])

def chat():

data = request.json

user\_input = data.get("message")

if not user\_input:

return jsonify({"error": "No input message provided"}), 400

chatbot\_response = get\_chatbot\_response(user\_input)

return jsonify({"response": chatbot\_response})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

1. **Menjalankan Aplikasi Flask**
   * Jalankan aplikasi Flask dengan:

bash

Salin kode

python app.py

* + Kamu akan melihat pesan bahwa server berjalan di http://127.0.0.1:5000.

**3. Pengembangan Frontend**

1. **Membuat File index.html**
   * Buat file index.html di direktori yang sama dengan app.py dengan kode berikut:

html

Salin kode

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Chatbot E-commerce</title>

</head>

<body>

<div>

<h3>Customer Support Chatbot</h3>

<div id="chat-window">

<div id="chat-output"></div>

<input type="text" id="user-input" placeholder="Ask me anything...">

<button onclick="sendMessage()">Send</button>

</div>

</div>

<script>

function sendMessage() {

const userInput = document.getElementById('user-input').value;

const chatOutput = document.getElementById('chat-output');

fetch('/chat', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify({ message: userInput })

})

.then(response => response.json())

.then(data => {

chatOutput.innerHTML += `<p><strong>You:</strong> ${userInput}</p>`;

chatOutput.innerHTML += `<p><strong>Chatbot:</strong> ${data.response}</p>`;

})

.catch(error => console.error('Error:', error));

}

</script>

</body>

</html>

1. **Menjalankan dan Mengakses Frontend**
   * Buka file index.html di browser atau atur Flask untuk melayani file tersebut. Tambahkan route di app.py:

python

Salin kode

@app.route('/')

def index():

return app.send\_static\_file('index.html')

* + Tempatkan index.html di folder static jika menggunakan send\_static\_file.

**4. Pengujian**

1. **Uji Interaksi dengan Endpoint /chat**
   * **Menggunakan Postman**:
     + Buka Postman dan buat permintaan POST ke http://127.0.0.1:5000/chat.
     + Atur header Content-Type ke application/json.
     + Masukkan body request JSON:

json

Salin kode

{

"message": "Apa produk terbaru?"

}

* + - Klik "Send" dan periksa respons yang diterima.
  + **Menggunakan cURL**:
    - Jalankan perintah berikut di terminal:

bash

Salin kode

curl -X POST http://127.0.0.1:5000/chat -H "Content-Type: application/json" -d "{\"message\": \"Apa produk terbaru?\"}"

1. **Uji Antarmuka Pengguna di Browser**
   * Buka http://127.0.0.1:5000 di browser jika route / sudah diatur.
   * Kirimkan pesan melalui antarmuka dan verifikasi respons dari chatbot.
2. **Uji Kasus Ekstrem dan Error Handling**
   * Coba kirimkan input yang kosong atau tidak sesuai dan periksa penanganan kesalahan.
3. **Pengujian Kinerja**
   * Gunakan alat pengujian beban untuk mensimulasikan banyak pengguna dan memeriksa performa.

**5. Penerapan ke Produksi (Opsional)**

1. **Pengaturan Web Server**
   * Gunakan server produksi seperti Gunicorn untuk menjalankan aplikasi Flask:

bash

Salin kode

pip install gunicorn

gunicorn -w 4 app:app

1. **Deploy ke Platform Cloud**
   * Deploy aplikasi ke layanan seperti Heroku, AWS, atau Google Cloud Platform mengikuti panduan mereka.

# Result

**Penjelasan Kegagalan Eksperimen karena Keterbatasan Rate Limit ChatGPT**

**1. Apa itu Rate Limit?** Rate limit adalah batasan jumlah permintaan (requests) yang dapat dilakukan ke API dalam periode waktu tertentu. Batasan ini diterapkan untuk mencegah penyalahgunaan layanan dan untuk memastikan bahwa sumber daya server digunakan secara adil di antara semua pengguna.

**2. Mengapa Eksperimen Gagal?** Eksperimen kamu mengalami kegagalan karena melebihi batas jumlah permintaan API yang diizinkan. Hal ini dapat terjadi karena beberapa alasan:

* **Frekuensi Permintaan yang Tinggi**: Jika aplikasi atau eksperimen kamu mengirimkan permintaan API ke ChatGPT lebih sering dari batas yang ditentukan, maka kamu akan menghadapi error rate limit.
* **Penggunaan Berlebihan**: Jika eksperimen kamu memerlukan banyak interaksi dengan model dalam waktu singkat atau membutuhkan volume data yang besar, ini bisa dengan cepat melampaui batas kuota yang diberikan.
* **Kesalahan Pengaturan**: Implementasi atau konfigurasi yang tidak sesuai dengan kebijakan penggunaan API juga bisa menyebabkan masalah ini.

**3. Dampak dari Keterbatasan Rate Limit** Keterbatasan rate limit mengakibatkan:

* **Kegagalan Permintaan**: Permintaan ke API tidak diproses, sehingga tidak ada respons dari ChatGPT yang diterima oleh aplikasi kamu.
* **Gangguan pada Fungsi**: Fungsi chatbot atau bagian dari aplikasi yang bergantung pada respons dari ChatGPT menjadi tidak berfungsi, mempengaruhi kinerja dan fungsionalitas keseluruhan dari sistem.

**4. Langkah-langkah yang Diambil**

* **Pemantauan Kuota**: Memantau penggunaan API secara berkala untuk memastikan bahwa jumlah permintaan tidak melampaui batas yang ditentukan.
* **Optimasi Permintaan**: Mengurangi jumlah permintaan yang dilakukan, menggunakan cache untuk respons yang sering digunakan, dan mengimplementasikan penjadwalan untuk menghindari lonjakan permintaan.
* **Peningkatan Rencana**: Menilai opsi untuk meningkatkan rencana API ke level yang lebih tinggi jika penggunaan yang diperlukan melebihi batas kuota yang tersedia.

**5. Solusi yang Dapat Diterapkan**

* **Periksa dan Sesuaikan Penggunaan**: Tinjau pengaturan penggunaan API dan sesuaikan permintaan agar sesuai dengan batas kuota yang tersedia.
* **Tingkatkan Langganan**: Pertimbangkan untuk meningkatkan rencana langganan API agar mendapatkan kuota yang lebih besar dan dapat mendukung kebutuhan eksperimen.
* **Kontak Dukungan**: Hubungi tim dukungan OpenAI untuk mendapatkan bantuan dalam mengelola batasan atau mengatur penggunaan yang lebih sesuai dengan kebutuhan eksperimen.