P-ISSN: 1979-5661
E-ISSN: 2622-321X

Implementasi Aplikasi Penerjemah Multi Bahasa Berbasis Python dengan Integrasi Google *Translate* API dan GUI Tkinter

Made Detriasmita Saientisna[a1], Ni Wayan Sukarini[a2], I Gusti Agung Gede Arya Kadyanan[a1], Ni Komang Ayu Juliana[a4]

[a,b]Program Studi Sastra Inggris, Fakultas Ilmu Budaya
[c,d]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Udayana, Bali
Jln. Raya Kampus UNUD, Bukit Jimbaran, Kuta Selatan, Badung, 08261, Bali, Indonesia
[1]detriasmita@unud.ac.id
[2]gungde@unud.ac.id
[3]wayan_sukarini@unud.ac.id
[4]juliana.2208561046@student.unud.ac.id

Abstract

Dalam era globalisasi, komunikasi lintas bahasa menjadi semakin penting. Teknologi penerjemahan otomatis menawarkan solusi untuk mengatasi hambatan bahasa, salah satunya adalah dengan mengintegrasikan Google Translate API dalam aplikasi penerjemah berbasis Python. Penelitian ini bertujuan untuk merancang dan mengimplementasikan aplikasi penerjemah multi bahasa yang memanfaatkan Google Translate API dan antarmuka grafis berbasis Tkinter. Aplikasi ini memungkinkan pengguna untuk menerjemahkan teks antarbahasa dengan cepat dan akurat, mendukung lebih dari 100 bahasa. Proses penelitian mencakup studi pustaka tentang penerjemahan otomatis dan antarmuka pengguna, serta pengembangan aplikasi menggunakan metode pengembangan perangkat lunak. Hasil dari penelitian ini menunjukkan bahwa aplikasi yang dikembangkan mampu memberikan pengalaman pengguna yang baik melalui desain antarmuka yang intuitif dan responsif, serta kemampuan penerjemahan yang efektif. Aplikasi ini diharapkan dapat menjadi solusi praktis bagi pengguna yang membutuhkan penerjemahan lintas bahasa secara mudah dan cepat.

Kata Kunci: *Penerjemah, Python, Google Translate API, Tkinter*

1. Pendahuluan

Dalam era globalisasi yang terus berkembang, kebutuhan akan komunikasi lintas bahasa semakin menjadi prioritas. Bahasa tidak lagi menjadi penghalang dalam berinteraksi di era digital ini, terutama dengan pesatnya perkembangan teknologi informasi. Peran teknologi dalam menjembatani perbedaan bahasa sangat krusial, terutama dalam mendukung berbagai sektor seperti pendidikan, bisnis, dan pemerintahan. Aplikasi penerjemah multi bahasa telah menjadi alat penting untuk memfasilitasi komunikasi tersebut, dan salah satu teknologi yang memungkinkan hal ini adalah Google Translate API. Aplikasi penerjemah berbasis Python yang memanfaatkan Google Translate API merupakan sebuah solusi inovatif yang mampu mengatasi tantangan komunikasi antarbahasa. API ini menawarkan kemampuan penerjemahan otomatis dalam lebih dari 100 bahasa, memberikan fleksibilitas tinggi dalam penggunaan global. Kombinasi teknologi ini dengan antarmuka grafis yang intuitif, seperti yang disediakan oleh pustaka Tkinter pada Python, dapat menciptakan pengalaman pengguna yang optimal dan mudah diakses oleh berbagai kalangan.

Google Translate API tidak hanya memberikan akses cepat dan akurat terhadap penerjemahan teks, tetapi juga menyediakan alat yang efisien untuk diterapkan dalam berbagai aplikasi praktis. API kepanjangan dari *Application Programming Interface* adalah gabungan dari sekumpulan fungsi,protocol, perintah, yang dapat digunakan *programmer* di saat membuat program perangkat lunak yang menggunakan sebuah sistem operasi tertentu. Dengan menggunakan dan merancang aplikasi penterjemah buatan sendiri ini, kita tidak perlu lagi membuka web situs Google Translate setiap kali akan menterjemahkan sebuah teks, namun cukup dengan menjalankan aplikasi yang telah kita dibuat sendiri ini [1]. Integrasi API ini dengan Python memungkinkan pengembang untuk membangun aplikasi penerjemah yang tidak hanya fungsional tetapi juga dapat dikustomisasi sesuai kebutuhan pengguna.

Selain itu, Tkinter sebagai pustaka GUI standar untuk Python memungkinkan pembuatan antarmuka yang user-friendly, meningkatkan pengalaman pengguna dalam berinteraksi dengan aplikasi tersebut. Hal ini penting karena antarmuka pengguna yang baik dapat menentukan keberhasilan aplikasi dalam menarik dan mempertahankan pengguna.

Penelitian terdahulu telah menunjukkan bahwa penerjemahan otomatis berbasis API dapat secara signifikan meningkatkan efisiensi komunikasi. Al-Harbi (2020) menemukan bahwa penggunaan aplikasi penerjemah berbasis API dapat mengurangi waktu yang dibutuhkan untuk menerjemahkan dokumen, sehingga meningkatkan produktivitas secara keseluruhan [2]. Selain itu, studi oleh Zhang, *et al*. (2019) menyoroti pentingnya desain antarmuka pengguna yang baik dalam meningkatkan kepuasan pengguna terhadap aplikasi penerjemah [3]. Dengan demikian, menggabungkan penerjemahan otomatis dengan antarmuka grafis yang ramah pengguna menjadi strategi yang efektif untuk menciptakan aplikasi penerjemah yang andal dan efisien. Pengembangan aplikasi penerjemah ini juga mencerminkan tren yang lebih luas dalam pemanfaatan kecerdasan buatan dan machine learning dalam pemrosesan bahasa alami. Di masa depan, aplikasi penerjemah berbasis API seperti ini berpotensi berkembang lebih lanjut dengan dukungan teknologi yang semakin canggih, seperti penerjemahan kontekstual dan kemampuan memahami nuansa budaya dalam bahasa. Inovasi ini akan menjadi landasan penting dalam mendukung komunikasi global yang lebih inklusif dan efektif, menjadikan bahasa sebagai jembatan, bukan penghalang.

2. Metode Penelitian

Penelitian ini menggunakan pendekatan pengembangan perangkat lunak untuk merancang dan mengimplementasikan aplikasi penerjemah multibahasa. Proses penelitian terbagi menjadi dua tahap, yakni: studi pustaka dan perancangan serta pengembangan aplikasi.

2.1 Studi Pustaka

Pada tahap ini, dilakukan peninjauan literatur terkait dengan teknologi yang digunakan, yaitu Google Translate API dan antarmuka pengguna GUI Tkinter. Studi ini mencakup pemahaman tentang cara kerja API, bagaimana integrasinya dalam aplikasi Python, dan desain antarmuka pengguna yang efektif. Selain itu, dilakukan analisis terhadap aplikasi serupa yang sudah ada untuk mengidentifikasi kelebihan dan kekurangan yang dapat menjadi bahan pertimbangan dalam pengembangan aplikasi ini.

a. Penerjemah Bahasa

Penerjemahan bahasa, atau *language translation*, adalah proses mengubah teks atau ucapan dari satu bahasa (bahasa sumber) ke bahasa lain (bahasa target) tanpa mengubah makna asli dari konten tersebut [4]. Penerjemahan bahasa telah menjadi kebutuhan penting di era globalisasi, di mana komunikasi lintas bahasa semakin sering terjadi. Proses penerjemahan tidak hanya melibatkan pemahaman struktur bahasa tetapi juga konteks budaya, idiom, dan nuansa yang melekat pada teks asli.

Teknologi penerjemahan bahasa telah mengalami perkembangan signifikan sejak diperkenalkannya metode berbasis aturan (*rule-based methods*) pada pertengahan abad ke-20. Metode ini didasarkan pada aturan tata bahasa dan kosakata yang telah diprogram secara manual. Namun, metode ini sering kali menghasilkan terjemahan yang kaku dan tidak alami karena keterbatasan dalam menangani kompleksitas bahasa manusia.

Perkembangan selanjutnya membawa penerjemahan berbasis korpus (*corpus-based translation*), yang melibatkan penggunaan data terjemahan besar (korpus) untuk melatih model penerjemahan. Salah satu pendekatan populer dalam kategori ini adalah *statistical machine translation* (SMT), yang menggunakan probabilitas untuk menentukan terjemahan terbaik berdasarkan pasangan teks yang sudah diterjemahkan sebelumnya [5].

Saat ini, penerjemahan bahasa didominasi oleh *neural machine translation* (NMT), yang merupakan bagian dari kemajuan dalam pembelajaran mendalam (*deep learning*) [6]. NMT menggunakan jaringan saraf tiruan untuk mempelajari hubungan kompleks antara bahasa sumber dan bahasa target, menghasilkan terjemahan yang lebih alami dan kontekstual. Keunggulan NMT dibandingkan metode sebelumnya adalah kemampuannya dalam memahami konteks kalimat secara keseluruhan, bukan hanya berdasarkan kata per kata.

b. Google Translate API

Google Translate API merupakan layanan penerjemahan berbasis *cloud* yang dikembangkan oleh Google, yang mendukung lebih dari 100 bahasa [7]. API ini digunakan untuk menerjemahkan teks, mendeteksi bahasa, dan menghasilkan terjemahan secara real-time.

Teknologi yang mendasari API ini adalah *neural machine translation* (NMT), yang mampu menghasilkan terjemahan yang lebih akurat dan alami dibandingkan dengan metode penerjemahan berbasis aturan tradisional. Google Translate adalah salah satu translate bahasa online paling terkenal dan paling banyak digunakan orang di seluruh dunia saat ini.

Dalam berbagai penelitian, Google Translate API telah terbukti efektif dalam mendukung berbagai aplikasi, seperti aplikasi pembelajaran bahasa, penerjemah teks otomatis, dan sistem informasi multibahasa. Keunggulan utama dari penggunaan API ini meliputi kecepatan dalam menghasilkan terjemahan, dukungan terhadap berbagai bahasa, serta kemampuannya untuk terus memperbaiki kualitas terjemahan melalui pembaruan algoritma yang dilakukan secara berkala oleh Google. Selain itu, fleksibilitas dan kemudahan integrasi dengan berbagai bahasa pemrograman, termasuk Python, menjadikan Google Translate API pilihan yang populer dalam pengembangan aplikasi multibahasa.

c. GUI Tkinter

Graphical User Interface (GUI) adalah suatu antarmuka yang memungkinkan pemakai berinteraksi dengan *system* operasi melalui seperangkat elemen atau yang biasa disebut widget, seperti list box, radio button, ikon, dan kotak dialog [8]. Tkinter adalah pustaka standar yang disediakan oleh Python untuk pengembangan antarmuka pengguna grafis (GUI). Tkinter berfungsi sebagai antarmuka ke *toolkit* GUI Tk, yang merupakan *toolkit* lintas *platform* yang banyak digunakan dan telah terintegrasi dengan Python [9]. Tkinter menyediakan berbagai komponen GUI, seperti tombol, label, kotak teks, dan menu, yang memungkinkan pengembang untuk menciptakan aplikasi desktop yang interaktif.

2.2 Perancangan dan Pengembangan Aplikasi

Perancangan aplikasi penerjemah multi bahasa dengan integrasi Google Translate API menggunakan GUI Tkinter merupakan tahapan inti dalam penelitian ini. Pada tahap ini, aplikasi dirancang dan dikembangkan untuk mencapai tujuan yang telah ditetapkan, yaitu menciptakan solusi penerjemahan yang mudah digunakan oleh berbagai kalangan pengguna. Proses perancangan dan pengembangan ini melibatkan beberapa langkah utama, yang dirinci sebagai berikut:

a. Perancangan Antarmuka Pengguna

Perancangan antarmuka pengguna (*User Interface*/UI) adalah aspek fundamental dalam pengembangan aplikasi. Pada tahap ini, dilakukan analisis mendalam terhadap kebutuhan pengguna untuk menentukan tata letak dan elemen-elemen UI yang tepat. Desain antarmuka harus menjamin kemudahan penggunaan (*usability*), sehingga pengguna dari berbagai latar belakang dapat dengan mudah memahami dan menggunakan aplikasi. Beberapa faktor yang dipertimbangkan dalam perancangan UI ini antara lain:

1. **Penempatan Komponen**:Penempatan elemen seperti kotak input teks, tombol terjemahan, dan area hasil terjemahan dirancang agar mudah diakses dan intuitif. Tata letak harus memastikan bahwa pengguna dapat fokus pada fungsi utama aplikasi tanpa terganggu oleh elemen yang tidak perlu.
2. **Desain Visual**: Desain visual mencakup pemilihan warna, font, dan ikon yang tidak hanya mendukung keterbacaan tetapi juga memberikan kenyamanan visual bagi pengguna. Skema warna yang tidak melelahkan mata dan tata letak yang rapi serta terstruktur diprioritaskan untuk memberikan pengalaman pengguna yang optimal.
3. **Responsivitas**: Alur navigasi harus sederhana, logis, dan memungkinkan pengguna untuk berpindah antar fungsi aplikasi dengan mudah tanpa kebingungan. Navigasi yang efisien ini dirancang untuk meminimalkan jumlah klik dan tindakan yang diperlukan pengguna untuk mencapai tujuannya.
4. **Navigasi yang Efisien**: Sebelum implementasi akhir, dilakukan uji coba pada prototipe antarmuka untuk mendapatkan umpan balik dari pengguna potensial. Hal ini penting untuk memastikan bahwa desain yang dibuat benar-benar memenuhi kebutuhan pengguna dan tidak ada elemen yang membingungkan atau sulit digunakan.

b. Integrasi Google Translate API

Integrasi Google Translate API adalah langkah krusial dalam pengembangan aplikasi penerjemah ini. API digunakan untuk memfasilitasi proses penerjemahan teks secara otomatis, memungkinkan pengguna untuk mendapatkan hasil terjemahan dengan cepat dan akurat. Tahap ini melibatkan beberapa sub-langkah, di antaranya:

1. **Pemahaman API**: Proses dimulai dengan mempelajari dokumentasi Google Translate API untuk memahami bagaimana API bekerja. Ini termasuk mempelajari endpoint yang tersedia, parameter yang diperlukan, serta cara mengirim dan menerima data dengan format yang benar.
2. **Autentikasi API**: Mengatur autentikasi adalah langkah penting untuk mendapatkan akses ke API. Ini biasanya melibatkan pembuatan kunci API (API *key*) melalui Google Cloud Console dan konfigurasi pengaturan yang diperlukan untuk proyek aplikasi. Proses ini memastikan bahwa hanya aplikasi yang sah yang dapat mengakses layanan API.
3. **Pengiriman Permintaan (Request)**: Setelah autentikasi berhasil, langkah berikutnya adalah mengimplementasikan fungsi dalam aplikasi yang dapat mengirimkan permintaan ke Google Translate API. Fungsi ini akan mengambil teks yang dimasukkan oleh pengguna, menentukan bahasa sumber dan bahasa target, lalu mengirimkan data tersebut ke API untuk diterjemahkan.
4. **Pengolahan Hasil (Response Handling)**: Setelah permintaan diterima oleh API dan hasil terjemahan dikirimkan kembali, hasil tersebut diolah dan ditampilkan dalam antarmuka pengguna. Proses pengolahan mencakup penguraian data JSON yang diterima, penanganan karakter non-standar, serta penyelesaian kasus-kasus khusus seperti teks panjang atau terjemahan yang mungkin memerlukan penyesuaian.

3. Hasil Dan Pembahasan

3.1. Penginstalan Library

Langkah pertama yang harus dilakukan adalah menginstal library yang diperlukan. Salah satu library utama yang digunakan adalah *googletrans*, sebuah library Python yang memudahkan interaksi dengan layanan Google Translate. *Library googletrans* memungkinkan pengguna untuk menerjemahkan teks secara otomatis dengan mendeteksi bahasa sumber dan menerjemahkannya ke bahasa target yang diinginkan. Hal ini sangat berguna dalam pengembangan aplikasi penerjemah multi bahasa, di mana pengguna mungkin tidak selalu mengetahui bahasa sumber dari teks yang ingin diterjemahkan. Selain itu, *googletrans* mendukung lebih dari 100 bahasa, menjadikannya solusi yang sangat fleksibel untuk berbagai kebutuhan penerjemahan.

```
[7]: pip install googletrans==4.0.0-rc1

Requirement already satisfied: googletrans==4.0.0-rc1 in c:\users\anant\appdata\local\programs\python\python311\lib\site-pac
Requirement already satisfied: httpx==0.13.3 in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (fr
(0.13.3)
Requirement already satisfied: certifi in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (from htt
0-rc1) (2023.7.22)
Requirement already satisfied: hstspreload in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (from
4.0.0-rc1) (2024.8.1)
Requirement already satisfied: sniffio in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (from htt
0-rc1) (1.3.1)
Requirement already satisfied: chardet==3.* in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (fro
=4.0.0-rc1) (3.0.4)
Requirement already satisfied: idna==2.* in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (from h
0.0-rc1) (2.10)
Requirement already satisfied: rfc3986<2,>=1.3 in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (
ns==4.0.0-rc1) (1.5.0)
Requirement already satisfied: httpcore==0.9.* in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (
ns==4.0.0-rc1) (0.9.1)
Requirement already satisfied: h11<0.10,>=0.8 in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (f
0.13.3->googletrans==4.0.0-rc1) (0.9.0)
Requirement already satisfied: h2==3.* in c:\users\anant\appdata\local\programs\python\python311\lib\site-packages (from htt
```

Gambar 1. Instalasi *Library*

Instalasi library ini dapat dilakukan melalui *pip*, manajer paket standar untuk Python, dengan perintah sederhana seperti pip install googletrans. Setelah terinstal, library ini dapat langsung digunakan dalam proyek Python dengan mengimpor modulnya. Selain fungsionalitas dasar untuk menerjemahkan teks, *googletrans* juga mendukung fitur tambahan seperti deteksi bahasa, yang memungkinkan aplikasi

untuk secara otomatis mengenali bahasa teks yang dimasukkan oleh pengguna tanpa memerlukan input manual.

```
from tkinter import *
from tkinter import ttk
from googletrans import Translator, LANGUAGES
```

Gambar 2. *Import Module*

Library lainnya juga diperlukan agar program dapat terealisasikan dengan baik, adapun library lainnya yang diperlukan yaitu, *tkinter* dan ttk. *Tkinter* juga dapat digunakan untuk membuat aplikasi yang interaktif, seperti aplikasi yang dapat menerima input dari pengguna. Selain itu, ttk (Tkinter Toolkit) adalah submodul dari Tkinter yang menyediakan komponen-komponen tampilan yang lebih modern dan fleksibel. Ttk menyediakan berbagai komponen, seperti tombol, label, dan lain-lain, yang dapat digunakan untuk membuat tampilan aplikasi yang lebih menarik.

Ttk juga dapat digunakan untuk membuat tema-tema yang dapat digunakan untuk mengubah tampilan aplikasi. Dengan menggunakan Tkinter dan ttk, kita dapat membuat aplikasi yang memiliki tampilan yang menarik dan interaktif. Kita dapat menggunakan Tkinter untuk membuat jendela, tombol, dan label, serta menggunakan ttk untuk membuat tema-tema yang dapat digunakan untuk mengubah tampilan aplikasi.

3.2. Pengaturan Tampilan Aplikasi

```
root = Tk()
root.geometry('1100x320')
root.resizable(0,0)
root['bg']= 'skyblue'
```

Gambar 3. Pengaturan Jendela Aplikasi

Program diatas akan membuat sebuah tampilan jendela utama aplikasi dengan menggunakan library tkinter. Dalam membuat jendela utama aplikasi, kita dapat menggunakan fungsi geometry() yang disediakan oleh tkinter untuk mengatur ukuran jendela aplikasi.

Fungsi ini memungkinkan kita untuk mengatur ukuran jendela aplikasi dengan lebar dan tinggi yang diinginkan. Dalam contoh kode di atas, kita dapat menambahkan fungsi geometry() untuk mengatur ukuran jendela aplikasi menjadi 1100 x 320. Ukuran ini dipilih untuk memberikan ruang yang cukup untuk menampilkan berbagai komponen aplikasi, seperti label, tombol, dan lain-lain.

Selain itu, kita juga dapat menggunakan fungsi resizable() untuk mencegah jendela diubah ukurannya. Fungsi ini memungkinkan kita untuk mengatur apakah jendela aplikasi dapat diubah ukurannya atau tidak. Dalam contoh kode di atas, kita menggunakan fungsi resizable() untuk mencegah jendela diubah ukurannya, sehingga tampilan aplikasi menjadi lebih stabil dan konsisten.

Warna dari latar belakang jendela aplikasi akan diatur berwarna skyblue. Dengan menggunakan fungsi configure(), kita dapat mengatur warna latar belakang jendela aplikasi sesuai dengan kebutuhan. Dalam contoh kode di atas, kita menggunakan fungs*i configure*() untuk mengatur warna latar belakang jendela aplikasi menjadi skyblue, sehingga tampilan aplikasi menjadi lebih menarik dan atraktif.

3.3. Pengaturan Kotak *Input*

```
Label(root, text="Enter Text", font='Arial 12 bold', bg='skyblue').place(x=30, y=90)
Input_text = Entry(root, width=60, font='Arial 12')
Input_text.place(x=30, y=130)
```

Gambar 4. Pengaturan Kotak *Input*

Gambar 4 diatas menunjukkan *code* program yang berfungsi untuk membangun 2 elemen penting yaitu label dan kotak input teks. Label digunakan untuk memberikan petunjuk atau informasi kepada

pengguna tentang apa yang harus dilakukan, sedangkan kotak input teks digunakan untuk memungkinkan pengguna memasukkan data atau teks yang ingin diproses oleh aplikasi.

Pertama, kita akan membuat label dengan tulisan "Enter Text" yang berfungsi sebagai petunjuk bagi pengguna untuk memasukkan teks yang ingin diterjemahkan. Label ini ditempatkan pada posisi tertentu di jendela aplikasi, sehingga pengguna dapat dengan mudah melihat dan memahami apa yang harus dilakukan. Dalam contoh kode di atas, kita menggunakan fungsi Label() dari tkinter untuk membuat label dengan tulisan "Enter Text". Kita juga dapat mengatur font dan ukuran label dengan menggunakan fungsi config().

Kedua, kita akan membuat kotak input teks yang memungkinkan pengguna untuk mengetikkan teks secara langsung. Kotak input ini memiliki ukuran yang cukup lebar agar pengguna dapat memasukkan teks dengan nyaman. Dalam contoh kode di atas, kita menggunakan fungsi Entry() dari tkinter untuk membuat kotak input teks. Kita juga dapat mengatur ukuran dan font kotak input teks dengan menggunakan fungsi config(). Kedua elemen ini dirancang dengan tampilan yang konsisten dengan keseluruhan desain aplikasi, menggunakan font “arial 12 bold” dan warna latar belakang skyblue.

3.4. Pengaturan Kotak *Output*

```
Label(root, text = "Output", font = 'arial 12 bold', bg='skyblue').place(x=780, y=90)
Output_text = Text(root, font='arial 10', height = 11, wrap = WORD, padx = 5, width= 50)
Output_text.place(x=700, y=130)
```

Gambar 5, Pengaturan Kotak *Output*

Gambar 5 diatas menunjukkan *code* program yang berfungsi untuk menampilkan antarmuka pengguna (GUI) sederhana yang terdiri dari sebuah label dan area teks. Label dengan teks "Output" berfungsi sebagai penunjuk atau judul untuk area teks di bawahnya. Area teks ini memungkinkan pengguna untuk memasukkan atau menampilkan teks multi-baris, yang bisa digunakan untuk berbagai keperluan, seperti menampilkan hasil output dari sebuah proses, menerima masukan dari pengguna, atau sekadar menampung teks panjang.

Dengan menggunakan Tkinter, program ini membuat antarmuka yang bisa menempatkan elemen-elemen GUI pada posisi yang diinginkan dalam jendela aplikasi, membuatnya lebih fleksibel dalam desain dan penggunaan. Output dari program ini adalah jendela aplikasi yang menampilkan label "Output" di atas area teks, memberikan indikasi kepada pengguna bahwa teks yang dimasukkan atau ditampilkan di area tersebut merupakan hasil dari suatu proses atau data yang relevan.

3.5. Inisialisasi Fungsi Penerjemah

```
language = list(LANGUAGES.values())

dest_lang = ttk.Combobox(root, value= language, width = 22)
dest_lang.place(x=130, y=160)
dest_lang.set('Choose Language')

def Translate() :
    translator = Translator()
    translated = translator.translate(text=Input_text.get(), dest = dest_lang.get())
    Output_text.delete(1.0, END)
    Output_text.insert(END, translated.text)

trans_btn = Button(root, text='Translate', font='arial 12 bold', pady = 5, command= Translate, bg= 'orange', activebackground='green')
trans_btn.place(x=445, y=180)

root.mainloop()
```

Gambar 6. Inisialisasi Fungsi Penerjemah

Gambar 6 menunjukkan potongan program yang berfungsi untuk membuat antarmuka pengguna grafis (GUI) yang memungkinkan pengguna untuk menerjemahkan teks dari satu bahasa ke bahasa lain menggunakan library Tkinter dan Googletrans. Berikut adalah penjelasan mengenai setiap bagian kode tersebut:

a. Pertama, language = list(LANGUAGES.values()) mengambil daftar bahasa yang tersedia dari konstanta LANGUAGES dan mengkonversinya menjadi sebuah list yang berisi nama-nama

bahasa. List ini digunakan untuk mengisi opsi dalam sebuah combobox dengan teks *default* "*Choose Language*", yang memungkinkan pengguna memilih bahasa tujuan untuk terjemahan.

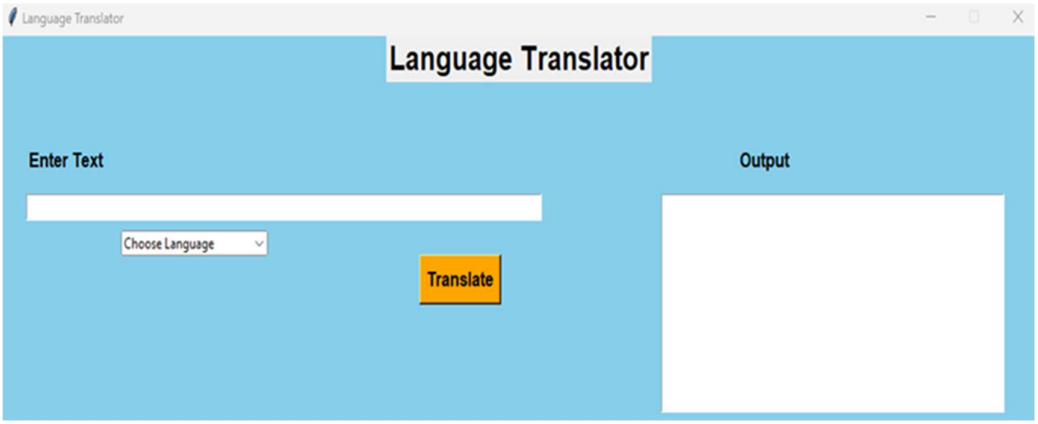
b. Fungsi Translate() kemudian didefinisikan untuk menangani proses penerjemahan. Di dalam fungsi ini, diinisialisasi objek penerjemah dari library Googletrans. Fungsi tersebut kemudian digunakan untuk mengambil teks yang dimasukkan oleh pengguna dari kotak input, lalu menerjemahkannya ke bahasa yang dipilih dalam combobox dest_lang, dan menyimpan hasil terjemahannya dalam variabel translated. Setelah itu, akan dihapus teks yang ada di widget Output_text untuk memastikan hasil terjemahan baru dapat ditampilkan tanpa tumpang tindih dengan teks sebelumnya. Hasil terjemahan kemudian dimasukkan ke dalam Output_text.
c. Bagian selanjutnya, membuat sebuah tombol dengan label "Translate" yang menggunakan font Arial berukuran 12 dan gaya tebal. Tombol ini memiliki padding vertikal sebesar 5 piksel, warna latar belakang oranye, dan warna latar belakang aktif hijau ketika ditekan. Tombol ini juga dikaitkan dengan fungsi Translate(), sehingga ketika tombol diklik, fungsi tersebut akan dieksekusi.
d. Terakhir, akan dijalankan loop utama Tkinter yang menjaga agar jendela aplikasi tetap terbuka dan merespons interaksi pengguna, seperti mengklik tombol dan memilih bahasa. Dengan begitu, seluruh kode ini membentuk sebuah aplikasi penerjemah sederhana yang memungkinkan pengguna memasukkan teks, memilih bahasa tujuan, dan mendapatkan terjemahan secara langsung melalui antarmuka grafis yang intuitif.

3.6. Tampilan Aplikasi Penerjemah

Adapun tampilan dari aplikasi penerjemah yang telah dikembangkan dengan Tkinter dan Google Translate API ditunjukkan pada gambar 7. Pada aplikasi ini, terdapat dua elemen input utama. Pertama adalah kotak teks yang diberi label "Enter Text," yang berfungsi sebagai tempat pengguna memasukkan teks yang ingin diterjemahkan. Area ini luas dan cukup besar, sehingga memungkinkan pengguna untuk menulis atau menempelkan teks yang panjang dengan nyaman.

Elemen kedua adalah combobox dengan label "Choose Language," yang menyediakan daftar bahasa tujuan yang dapat dipilih oleh pengguna. Daftar ini mencakup berbagai bahasa yang tersedia dalam Google Translate API, dan pilihan bahasa dilakukan dengan mengklik menu dropdown, yang menampilkan semua opsi dalam format yang mudah dipahami.

Setelah pengguna memasukkan teks dan memilih bahasa tujuan, mereka dapat mengklik tombol "Translate," yang ditampilkan dengan gaya tebal dan warna yang mencolok agar mudah dikenali. Tombol ini terletak strategis di bawah kedua input, sehingga memudahkan alur kerja pengguna. Ketika tombol ditekan, hasil terjemahan akan muncul secara otomatis di area output yang telah disiapkan di bawahnya. Area output ini diberi label "Output," dan tampilannya serupa dengan area input teks, memberikan kesan yang konsisten dan memudahkan pengguna untuk membandingkan teks asli dengan terjemahan.



Gambar 7. Tampilan Aplikasi Penerjemah

3.7 *Black Box Testing* Aplikasi Penerjemah

Black box testing adalah metode pengujian perangkat lunak yang fokus pada evaluasi fungsionalitas sistem tanpa memerlukan pengetahuan tentang kode internal atau rincian implementasi. Berikut adalah

black box testing dari Aplikasi Penerjemah yang telah dibuat menggunakan Tkinter dan Google Translate API, seperti yang terlihat pada tabel 1.

Tabel 1. Black Box Testing Aplikasi Penerjemah

No	Skenario Pengujian	Langkah Pengujian	Hasil Yang Diharapkan	Kesimpulan
1	Terjemahan teks dari bahasa Inggris ke bahasa Indonesia	Masukkan teks dalam bahasa Inggris ke dalam kotak teks "Enter Text," kemudian pilih bahasa "Indonesian" dari combobox "Choose Language." Setelah itu, tekan tombol "Translate."	Aplikasi menampilkan hasil terjemahan teks dalam bahasa Indonesia di area "Output."	Sesuai harapan
2	Pengujian dengan input teks kosong	Biarkan kotak "Enter Text" kosong tanpa memasukkan teks apa pun, kemudian pilih bahasa "Indonesian" dari combobox. Tekan tombol "Translate."	Aplikasi menampilkan pesan kesalahan atau tidak memberikan hasil terjemahan di area "Output."	Sesuai harapan
3	Pengujian tanpa memilih bahasa tujuan	Masukkan teks dalam bahasa apa pun di kotak "Enter Text," kemudian biarkan combobox bahasa tujuan kosong (tidak memilih bahasa), lalu tekan tombol "Translate."	Aplikasi menampilkan pesan kesalahan bahwa bahasa tujuan belum dipilih.	Sesuai harapan
4	Terjemahan dengan bahasa input berbeda dari bahasa tujuan	Masukkan teks dalam bahasa Jepang ke kotak teks "Enter Text," lalu pilih bahasa "Spanish" di combobox bahasa tujuan. Tekan tombol "Translate."	Aplikasi menampilkan hasil terjemahan dalam bahasa Spanyol meskipun teks input dalam bahasa Jepang.	Sesuai harapan
5	Terjemahan dengan bahasa input sama dengan bahasa tujuan	Masukkan teks dalam bahasa Inggris ke kotak teks "Enter Text," lalu pilih bahasa "English" dari combobox bahasa tujuan. Tekan tombol "Translate."	Aplikasi menampilkan teks yang sama di area "Output" tanpa perubahan.	Sesuai harapan
6	Terjemahan dengan memilih bahasa yang tidak didukung	Masukkan teks apa pun di kotak teks "Enter Text," lalu pilih bahasa yang tidak didukung oleh Google Translate API dari combobox "Choose Language." Tekan tombol "Translate."	Aplikasi menampilkan pesan kesalahan bahwa bahasa tidak didukung atau tidak ada hasil terjemahan.	Sesuai harapan

7	Menekan tombol translate tanpa memasukkan teks atau memilih bahasa tujuan	Jangan masukkan teks apa pun di kotak "Enter Text" dan jangan pilih bahasa tujuan, kemudian tekan tombol "Translate."	Aplikasi tidak melakukan apa-apa atau menampilkan pesan kesalahan bahwa teks input atau bahasa tujuan tidak boleh kosong.	Sesuai harapan
8	Terjemahan dengan input teks yang sangat panjang	Masukkan teks yang sangat panjang (lebih dari 500 karakter) ke dalam kotak "Enter Text," lalu pilih bahasa tujuan di combobox. Tekan tombol "Translate."	Aplikasi menampilkan hasil terjemahan secara lengkap di area "Output" tanpa terpotong.	Sesuai harapan
9	Uji tampilan antarmuka dengan memperbesar dan memperkecil jendela aplikasi	Ubah ukuran jendela aplikasi dengan memperbesar atau memperkecil tampilan.	Elemen-elemen antarmuka tetap berada di tempat yang benar dan tidak ada elemen yang terpotong atau rusak.	Sesuai harapan

4. Kesimpulan

Penelitian ini berhasil merancang dan mengimplementasikan aplikasi penerjemah multi bahasa berbasis Python yang mengintegrasikan Google *Translate* API dan menggunakan GUI Tkinter. Aplikasi ini memudahkan pengguna untuk menerjemahkan teks antarbahasa dengan cepat dan akurat, berkat dukungan teknologi Neural Machine Translation (NMT) dari Google Translate API. Tkinter memungkinkan pembuatan antarmuka yang intuitif dan responsif, dengan desain yang mempertimbangkan kemudahan penggunaan dan navigasi yang efisien. Studi pustaka yang dilakukan membantu dalam memahami teknologi dan mengidentifikasi kelebihan serta kekurangan dari aplikasi serupa, sehingga mendukung pengembangan aplikasi yang lebih baik. Hasilnya, penelitian ini sukses menghasilkan aplikasi penerjemah yang fungsional dan ramah pengguna.

Referensi

[1] Zein, A. 2018. Peran Text Processing Dalam Aplikasi Penerjemah Multi Bahasa Menggunakan Ajax API Google. *Sainstech*, 28(1), 19-23.

[2] Al-Harbi, A. 2020. The Impact of Machine Translation on Language Learning: A Study of EFL Students. *International Journal of Language and Linguistics*, 7(3), 45-52.

[3] Zhang, Y., Wang, L., & Liu, H. 2019. User Experience in Machine Translation: A Study of User Interface Design. *Journal of Computer and Communications*, 7(5), 12-20.

[4] Sudarmaji, S., Santoso, I., Mulyati, R. E. S. (2023). Analisis Kesalahan Hasil Terjemahan Mesin Penerjemah Teks Bahasa Jerman ke dalam Bahasa Indonesia. *Diglosia: Jurnal Kajian Bahasa, Sastra, Dan Pengajarannya*, 6(2), 483–500. https://doi.org/10.30872/diglosia.v6i2.668.

[5] Untara, W., Setiawan, T. (2020). PROBLEMA MESIN PENERJEMAH BERBASIS AI DALAM PROSES PENERJEMAHAN BUKU INGGRIS-INDONESIA DAN SOLUSINYA. *Adabiyyāt: Jurnal Bahasa Dan Sastra,* 4(1), 92–115. https://doi.org/10.14421/ajbs.2020.04105

[6] Chandra, M. R. (2024). Perbandingan Analisis Kesalahan Penginputan Speech to Text pada Aplikasi Terjemahan Google Translate dan Baidu Translate. *Salingka*, 21(1), 62–74. https://doi.org/10.26499/salingka.v21i1.1080.

[7] Ghirrid, A. A., Sari, R. T. K., & Aldisa, R. T. (2024). Algoritma natural language processing untuk aplikasi penerjemah (indonesia – jawa) menggunakan metode speech processing. *Jurnal JTIK (Jurnal Teknologi Informasi Dan Komunikasi)*, *8*(3), 746–759. https://doi.org/10.35870/jtik.v8i3.2244.

[8] Nova, S., Khotimah, N., & Wahyuningrum, M. Y. A. (2024). PEMANFAATAN CHATBOT MENGGUNAKAN NATURAL LANGUAGE PROCESSING UNTUK PEMBELAJARAN DASAR-DASAR GUI TKINTER PADA BAHASA PEMROGRAMAN PYTHON. *Jurnal Ilmiah Teknik*, *3*(1), 58–65. https://doi.org/10.56127/juit.v3i1.1162

[9] Akbar, M. G., Witriyono, H., Apridiyansyah, Y., & Abdullah, D. (2023). Implementation of the inter tk package, sub-process and os in the network management application development with python programming language. *Jurnal Komputer, Informasi Dan Teknologi*, *3*(1), 187-196-187–196. https://doi.org/10.53697/jkomitek.v3i1.1210.