**(GROUP 13) BUILDING A PYTHON CHATBOT FOR LANGUAGE TRANSLATION**

**Table of Contents**

1. Introduction

2. Prerequisites

3. Setup

4. Architecture

5. Implementation

6. Usage

7. Conclusion

**Introduction**

This document describes the implementation of a language translation chatbot using Python. The chatbot can translate text between multiple languages, leveraging pre-trained models from the `transformers` library.

**Prerequisites**

Before we begun on our project, we installed:

Python

Jupyter notebook

Pycharm

Other installations would include:

pip install transformers

pip install google translate

pip install googletrans

**Setup**

1. Install Required Libraries:

pip google translate

pip install googletrans

2. Download:

Python

Jupyter notebook

pycharm

**Architecture**

Our aim was for the chatbot architecture to consists of the following components:

1. Input Handling: where the bot was able to captures user input.

2. Language Detection: Identifies the source language.

3. Translation: Translates text to the target language.

4. Response Generation: Constructs the chatbot's response.

5. Output: Displays the translated text to the user.

**Implementation**

The first step was to import the needed libraries;

Pip install googletrans

Pip install google translate

from googletrans import Translator

def translate\_text(text, dest\_language):

translator = Translator()

translation = translator.translate(text, dest=dest\_language)

return translation.text

def main():

print("Welcome to the Python Translation App!")

text = input("What do you want to translate: ")

dest\_language = input("Enter the destination language (e.g., 'es' for Spanish, 'fr' for French): ")

try:

translated\_text = translate\_text(text, dest\_language)

print(f"Translated text: {translated\_text}")

except Exception as e:

print(f"An error occurred: {e}")

if \_name\_ == "\_main\_":

main()

**Usage**

1. Run the Chatbot:

We run and debugged the chatbot using jupyter and pycharm

2.Interact with the Chatbot:

After ensuring that the chatbot was running and had no errors. We checked to ensure that it could interact with the user.

-Example:

You: translate Hello, how are you?

Chatbot: Bonjour, comment ça va ?

3. End the Conversation:

- Type `bye` to terminate the chat.

**Conclusion**

This project documentation provides a structured approach to implementing a python chatbot for language translation. The chatbot utilizes the google translate API for translation and google trans for text processing. With this foundation, the user can extend the chatbot's capabilities to support more languages and additional features.