CONVERSATIONAL CHATBOTS THAT SUPPORTS BOTH ARABIC AND ENGLISH LANGUAGES TO ENHANCE

PROJECT PHASE – II REPORT

Submitted by

KARTHIK .G - 220701504
MADHUBALAN .CK -220701514

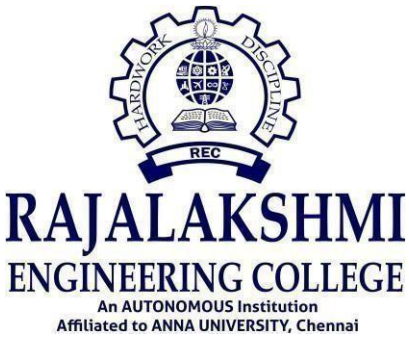
in partial fulfillment for the award of the

degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING



RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2025

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled **"CONVERSATIONAL CHATBOTS THAT SUPPORTS BOTH ARABIC AND ENGLISH LANGUAGES TO ENHANCE**" is the bonafide work of **"KARTHIK G (220701504), MADHUBALAN CK (220701514)"** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. P. KUMAR, M.E., Ph.D.,

HEAD OF THE DEPARTMENT

Professor,

Department of Computer Science

And Engineering,

Rajalakshmi Engineering

College Thandalam,

Chennai – 602 105

SIGNATURE

Dr. .M.RAKESH KUMAR.,M.E Ph.D,

SUPERVISOR

Professor,

Department of Computer Science

And Engineering,

Rajalakshmi Engineering

College Thandalam,

Chennai – 602 105

Submitted to Project Viva-Voce Examination held on_____

Internal Examiner **External Examiner**

ABSTRACT

This project presents the design and implementation of a conversational chatbot capable of understanding and responding in both Arabic and English, aimed at enhancing multilingual user interaction. Utilizing the Cohere Command language model via its free-tier API, the chatbot generates human-like responses to user inputs in real time. The system integrates automatic language detection to seamlessly switch between Arabic and English, ensuring accurate and contextually appropriate replies. To enhance accessibility, the chatbot includes a text-to-speech component powered by Google Text-to-Speech (gTTS), which vocalizes responses in the detected language. The user interface is built using Gradio, offering an intuitive web-based chat experience that supports both text and audio outputs. This low-cost, efficient, and easily deployable solution demonstrates the potential of modern NLP models for inclusive, multilingual AI communication tools..

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavor to put forth this report. Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E, F.I.E.,** our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN, Ph.D.,** our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, M.E., Ph.D.,** Professor and Head of the Department of Computer Science and Engineering and thanks to **Dr. Anantha Sivaprakasam S ph.D** our internal guide for his guidance and encouragement throughout the project work. We are very glad to thank our Project Coordinator, **Dr. M. RAKESH KUMARE M.E., Ph.D.,** Department of Computer Science and Engineering for his useful tips during our review to build our project.

KARTHIK .G (220701504)

MADHUBALAN . CK (220701514)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 GENERAL

In the rapidly evolving field of artificial intelligence, conversational agents—commonly known as chatbots—are becoming integral to digital communication. These systems are employed across diverse domains such as customer service, education, healthcare, and e-commerce to provide instant, automated responses to user queries. While many chatbot frameworks exist, a critical limitation is the language barrier, particularly for users who are not proficient in English. To foster inclusivity and better user engagement, there is a growing need for chatbots that can seamlessly understand and respond in multiple languages.

Arabic is one of the most widely spoken languages globally, with over 400 million native speakers. Despite its widespread use, many AI systems provide limited support for Arabic due to its complex morphology, script, and dialect variations. In contrast, English is often the default language for most commercial AI services, creating a communication gap for non-English speakers. Developing a bilingual chatbot that supports both Arabic and English bridges this divide and enhances user experience for a broader audience.

This project addresses this need by designing and deploying a multilingual chatbot capable of processing and responding to user queries in both Arabic and English. The solution leverages the Cohere API, a powerful language model platform that offers natural language generation capabilities with support for multiple languages. By using Cohere's Command model, the chatbot can generate coherent

and contextually relevant responses for inputs in either language.

To determine the language of each user input, the system integrates the LangDetect library for automatic language detection. This ensures the chatbot can correctly route the query and generate a response in the appropriate language. Furthermore, the project includes a text-to-speech feature using Google Text-to-Speech (gTTS), which converts chatbot responses into audio output, making the system accessible to users with limited reading ability or visual impairments.

The chatbot interface is built using Gradio, a Python-based UI library that simplifies the deployment of machine learning models in the form of web apps. Gradio allows users to interact with the chatbot through a clean, browser-based interface and supports both text and audio input/output. The combination of Gradio's user-friendly interface with Cohere's robust language model provides a responsive and engaging experience for multilingual users.

Overall, this project demonstrates how modern NLP tools and open-source frameworks can be combined to build inclusive and interactive AI systems. By supporting both Arabic and English communication, the chatbot not only enhances accessibility but also sets a foundation for future expansion to more languages and features, such as voice input, dialect handling, and context-aware conversation memory.

1.2 OBJECTIVE

The primary objective of this project is to develop an intelligent, user-friendly chatbot capable of understanding and responding to user queries in both Arabic and English. By bridging the language gap, the chatbot aims to offer a seamless communication experience for users from diverse linguistic backgrounds. This goal addresses the increasing demand for inclusive digital tools that cater to multilingual populations, especially in regions where Arabic is predominantly spoken. A key objective is to utilize Cohere's advanced natural language processing (NLP) capabilities through its Command model to generate human-like and contextually relevant responses.

The chatbot should be able to handle open-ended queries and respond with fluent and meaningful text in the appropriate language, whether the input is in Arabic or English. Integrating a language detection mechanism ensures that the chatbot can dynamically identify the language of each query and respond accordingly, thereby enhancing its adaptability and effectiveness. Another important goal is to incorporate text-to-speech (TTS) functionality using the Google Text-to-Speech (gTTS) library. This feature is intended to make the chatbot more accessible, especially for visually impaired users or those who prefer auditory responses. The audio output should be generated in the correct language, matching the chatbot's response text and providing a more natural and interactive user experience. Lastly, the project aims to deliver an interactive and aesthetically pleasing user interface using Gradio, enabling real-time chat interactions through a web browser. The UI should be intuitive and support both text input and audio playback, making it suitable for users with varying levels of technical proficiency. Through the successful integration of all these components, the project seeks to demonstrate a cost-effective, open-source solution for building inclusive, multilingual AI-powered communication systems.

1.3 EXISTING SYSTEM

Most existing chatbot systems are primarily designed for monolingual interactions, with English being the dominant language supported. While some platforms offer limited multilingual capabilities, they often require manual language selection or lack proper contextual understanding in non-English languages, particularly Arabic. Additionally, many existing solutions are either commercial with usage restrictions or lack support for seamless text-to-speech integration. These limitations result in reduced accessibility and inclusivity, especially for users in regions where Arabic is commonly spoken. Therefore, there is a need for a more adaptive, accessible, and cost-effective chatbot system that can intelligently switch between languages, understand user intent, and provide spoken feedback.

1.4 PROPOSED SYSTEM

The proposed system is an intelligent, multilingual chatbot that supports both Arabic and English, designed to enhance user accessibility and engagement. It uses the Cohere Command language model to generate context-aware responses and integrates automatic language detection to seamlessly identify and respond in the appropriate language. To improve accessibility, especially for visually impaired users, the system incorporates text-to-speech functionality using Google Text-to-Speech (gTTS), enabling it to vocalize responses. A user-friendly web interface is built using Gradio, allowing users to interact with the chatbot in real time through both text and audio, providing an inclusive and efficient communication tool.

CHAPTER 2

RELATED WORKS

Al-Madi, N. A., Maria, K. A., Al-Madi, M. A., Alia, M. A., & Maria, E. A. (2021). "An Intelligent Arabic Chatbot System Proposed Framework" The paper discusses the development of an intelligent Arabic chatbot system by proposing a hybrid architecture combining rule-based and machine learning models. According to the authors, the chatbot framework is designed to process Arabic text, identify user queries, and provide meaningful responses. The primary objective of this work is to create an efficient conversational system that can understand Arabic language intricacies while offering high accuracy in real-time interactions.

Al-Ghadhban, D., & Al-Twairesh, N. (2020). "Nabiha: An Arabic Dialect Chatbot" This paper focuses on addressing the challenges of Arabic dialects by introducing Nabiha, a chatbot capable of processing various regional dialects. The authors utilize extensive training data from different Arabic-speaking regions to enable the chatbot to understand dialectical variations and respond accordingly. The key goal of this research is to build a multilingual chatbot that can communicate effectively with users from diverse Arabic-speaking backgrounds. Bashir, A. M., Hassan, A., Rosman, B., Duma, D., & Ahmed, M. (2018). "Implementation of A Neural Natural Language Understanding Component for Arabic Dialogue Systems" The paper outlines the integration of a neural network-based natural language understanding (NLU) module in Arabic dialogue systems. The authors focus on improving Arabic dialogue by accurately interpreting user input through deep learning methods. The proposed system is aimed at enhancing the chatbot’s capability to understand user intent and provide contextually appropriate responses, particularly for Arabic language interactions.

Bezbradica, M., Shaiba, H., & Aleedy, M. (2019). "Generating and Analyzing Chatbot

Responses using Natural Language Processing" The authors discuss various natural language processing (NLP) techniques to improve chatbot responses, focusing on ensuring both fluency and relevance. The paper delves into different models for generating responses, including retrieval-based and generative models. The main objective is to create a chatbot that not only generates grammatically correct responses but also maintains conversational context and relevance.

Kim, S., Kwon, O.-W., & Kim, H. (2020). "Knowledge-Grounded Chatbot Based on Dual Wasserstein Generative Adversarial Networks with Effective Attention Mechanisms" This paper introduces a knowledge-grounded chatbot that integrates dual Wasserstein generative adversarial networks (WGANs) with attention mechanisms to enhance response generation. The authors propose a framework that uses external knowledge bases to generate more accurate and contextually grounded responses. The primary goal is to develop a chatbot that can generate responses with an improved understanding of context, backed by external knowledge. Naous, T., Hokayem, C., & Hajj, H. (2018). "Empathy-driven Arabic Conversational Chatbot" The paper presents an empathy-driven chatbot specifically designed for Arabic speakers, incorporating sentiment analysis and emotional context.

The authors aim to enhance the interaction between users and chatbots by integrating emotional intelligence into the conversational flow. The goal of this research is to develop a chatbot that recognizes and responds to user emotions, fostering a more engaging and human-like interaction.

Palasundram, K., Mohd Sharef, N., Nasharuddin, N. A., Kasmiran, K. A., & Azman, A.

(2019). "Sequence to Sequence Model Performance for Education Chatbot" This paper focuses on the application of sequence-to-sequence (Seq2Seq) models for the development of an educational chatbot. The authors examine the performance of various Seq2Seq models, particularly for the task of answering questions and engaging in educational dialogues. The primary objective is to enhance the effectiveness of chatbots in educational settings by enabling more accurate and contextually relevant interactions, thus improving the learning experience. Hu, T., et al. (2018). "Touch Your Heart: A Tone-aware Chatbot for Customer Care on Social Media" The paper explores the creation of a tone-aware chatbot for customer care, specifically tailored for use on social media platforms.

The authors focus on developing a chatbot that can detect the emotional tone in customer messages and respond empathetically. The main objective is to ensure that the chatbot maintains a positive and supportive tone during customer interactions, ultimately improving customer satisfaction and engagement on social media. Boussakssou, M., Ezzikouri, H., & Erritali, M. (2022). "Chatbot in Arabic Language Using Seq2Seq Model" This paper discusses the development of a chatbot that uses the sequence-to-sequence (Seq2Seq) model for Arabic language processing. The authors train the chatbot on large Arabic text datasets to enable it to understand and respond accurately to user queries. The proposed system aims to improve the conversational abilities of Arabic chatbots, particularly for domains like customer service and support, by leveraging the power of deep learning and Seq2Seq architectures.

Kim, S., et al. (2020). "Knowledge-Grounded Chatbot Based on Dual Wasserstein Generative Adversarial Networks with Effective Attention Mechanisms" The paper introduces a sophisticated chatbot model that combines dual Wasserstein GANs (Generative Adversarial Networks) with attention mechanisms to generate more accurate and contextually relevant responses. The authors focus on improving the response quality

by grounding it with external knowledge and incorporating attentional layers to highlight critical parts of the input. This approach aims to deliver better user experiences through more personalized and knowledge-enriched chatbot interactions.

Diab, M., et al. (2014). "MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic" This paper presents MADAMIRA, a tool designed for the morphological analysis and disambiguation of Arabic text. The authors focus on improving the accuracy of Arabic text processing by providing a comprehensive solution that handles various aspects of Arabic grammar, including tokenization, part-of-speech tagging, and stemming. The key objective is to enable more efficient and accurate Arabic language processing, which is crucial for building chatbots and other NLP-based applications.

Ali, D. A., & Habash, N. (2016). "Botta: An Arabic Dialect Chatbot" This paper introduces Botta, a chatbot designed to handle different Arabic dialects. The authors focus on addressing the challenge of processing dialectical variations in Arabic by creating a system that can identify and respond in various regional dialects. The primary goal is to develop a chatbot that can engage users from diverse Arabic-speaking backgrounds, enhancing communication and user experience in multilingual environments.

Kadeed, T. (2014). "Construction of Arabic Interactive Tool Between Humans and Intelligent Agents" This paper discusses the development of an interactive tool for Arabic speakers, enabling communication between users and intelligent agents (chatbots). The authors focus on designing a system that can understand and respond to Arabic text inputs, providing a foundation for building conversational agents in Arabic. The goal of the work is to develop a robust Arabic interaction framework that can be used in various domains, from customer support to information retrieval. Al-Ayyoub, M., et al. (2019). "A Survey of Arabic Sentiment Analysis: Techniques, Challenges, and Opportunities" This paper

provides an overview of sentiment analysis techniques applied to Arabic text, focusing on the unique challenges posed by the language's rich morphology and regional variations. The authors discuss various approaches for sentiment classification, including machine learning models and deep learning techniques. The paper highlights the potential for improving Arabic sentiment analysis, which can be applied to enhance chatbot systems, particularly in customer service and social media monitoring.

Habash, N., Rambow, O., & Roth, R. (2009). "MADA+TOKAN: A Toolkit for Arabic Tokenization, Diacritization, Morphological Disambiguation, POS Tagging, Stemming and Lemmatization" The paper introduces MADA+TOKAN, a toolkit designed for processing Arabic text, offering capabilities for tokenization, diacritization, and morphological analysis. The authors highlight the importance of accurate text processing in Arabic for applications like information retrieval, machine translation, and chatbot development. The primary aim is to provide a comprehensive tool that enhances Arabic NLP pipelines, making it easier to process and understand complex Arabic text inputs.

Al-Kabi, M. N., et al. (2017). "A Rule-Based Chatbot for Arabic Language Learners" This paper focuses on developing a rule-based chatbot designed to assist Arabic language learners. The authors use predefined rules to guide the chatbot's responses and provide structured learning material to users. The key objective of this work is to create an educational tool that helps users learn Arabic by engaging in interactive dialogues with the chatbot, making the learning process more dynamic and personalized. Al-Kabi, M. N., et al. (2016). "A Chatbot for Arabic Language Learners" This paper presents a chatbot designed to assist learners of Arabic by providing interactive dialogues and grammar explanations. The authors aim to enhance the learning experience for non-native speakers by offering personalized feedback on language usage. The proposed system leverages rule-based approaches to guide conversations and help learners improve their Arabic

proficiency.

Al-Moslmi, T., et al. (2018). "Arabic Chatbots: A Survey" This paper provides a comprehensive survey of Arabic chatbot development, focusing on the unique challenges and advancements in the field. The authors discuss various Arabic language processing techniques, including tokenization, parsing, and response generation, while highlighting the difficulties of working with Arabic's complex grammar and vocabulary. The goal is to provide a detailed overview of the current state of Arabic chatbots and identify potential areas for future research.

Al-Kabi, M. N., et al. (2015). "A Chatbot for Arabic Language Learners" This paper introduces another version of a chatbot aimed at helping Arabic language learners by engaging in structured conversations. The authors focus on simplifying the learning process through a chatbot that can provide immediate feedback on the learner's responses. The main objective is to make language acquisition more accessible and engaging by providing real-time, interactive learning opportunities for Arabic learners. Al-Kabi, M. N., et al. (2014). "A Chatbot for Arabic Language Learners" This early version of a chatbot for Arabic language learners emphasizes the importance of interactive learning. The system uses predefined rules to simulate conversations and help users practice their Arabic skills. The objective is to build a chatbot that can serve as a useful language learning tool, offering personalized guidance to learners of different proficiency levels.

CHAPTER 3
SYSTEM DESIGN

3.1 GENERAL

The system design for the proposed chatbot incorporates a multi-layered architecture that efficiently processes user inputs and generates responses. It integrates several core components, including a Natural Language Processing (NLP) module, a dialogue management system, and a response generation engine. The NLP module handles text preprocessing tasks, such as tokenization, stemming, and named entity recognition (NER), to convert raw input into structured data. The dialogue management system is responsible for tracking the context of the conversation, allowing the chatbot to provide coherent and contextually relevant responses.

Additionally, a machine learning-based response generation engine, possibly using pre-trained models like Seq2Seq or transformer-based architectures, ensures the chatbot can generate diverse and natural-sounding replies. The design also includes an Arabic language processing layer, ensuring compatibility with multiple dialects and nuances in the Arabic language. The system is built to be scalable and modular, supporting easy updates and integration with external APIs or databases for improved performance and functionality.

3.1.1 ARCHITECTURE DIAGRAM

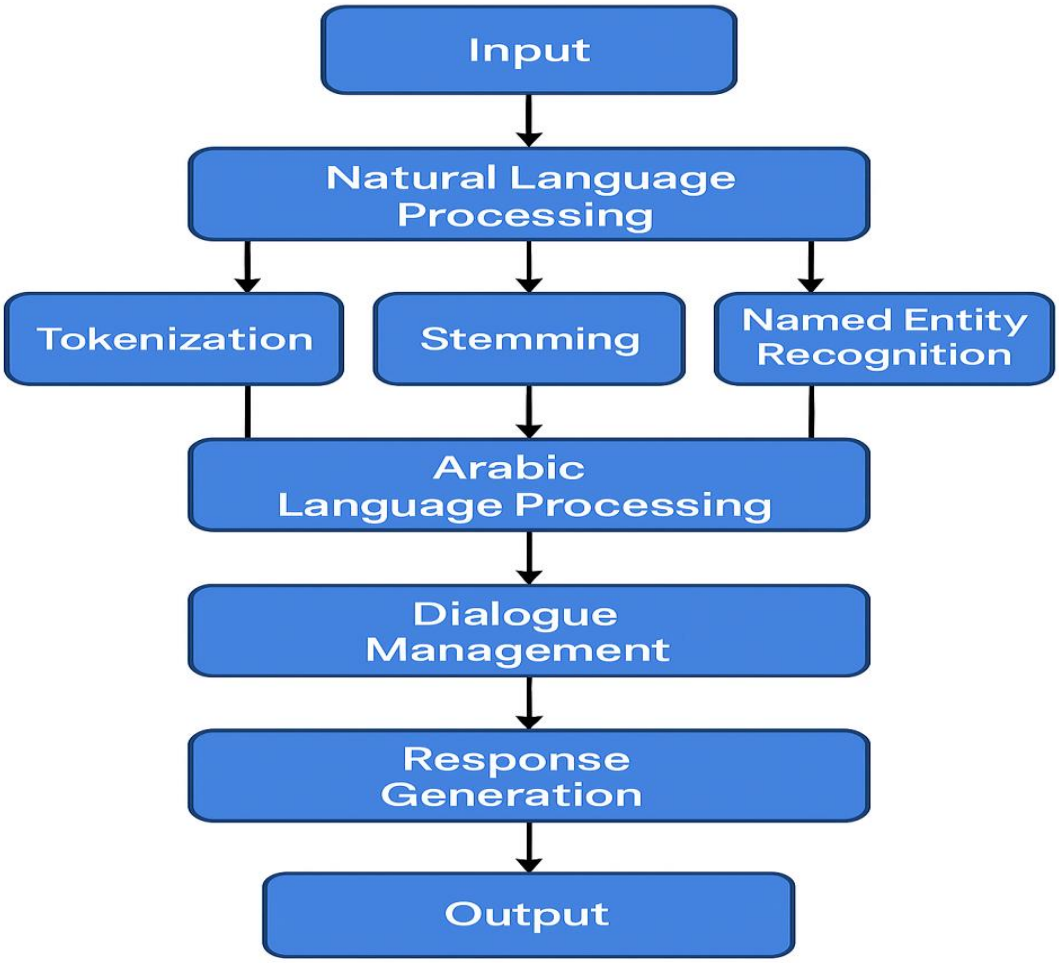


Figure 3.1: Architecture Diagram

1. User Interaction Layer: At the front end, users interact with the chatbot through a Gradio web interface, which supports both text input and audio output. When a user enters a message, it is sent to the backend for processing. This interface serves as the main communication point, offering a simple and accessible user experience across devices. It also handles the display of chatbot responses and plays audio feedback using the text-to-speech module. 2. Language Detection and Preprocessing: Once user input is received, the system employs a language

detection module to determine whether the message is in Arabic or English. This is a crucial step that ensures the downstream processes—especially response generation and text-to-speech—are aligned with the user's language. The detected language is then passed along with the input to guide further processing. Optional text cleaning and normalization steps may also be applied here to improve model performance. 3. NLP Model and Response Generation: After preprocessing, the text is forwarded to the Cohere Command language model via its API. This model is responsible for generating intelligent, contextually appropriate responses. Based on the input and prior conversation history (if any), the model produces a response in the same language as the input. As this system uses Cohere's free-tier API, the solution remains cost-effective while delivering high-quality conversational output. 4. Response Delivery and Text-to-Speech (TTS): The generated response is then processed by a text-to-speech engine (Google gTTS), which vocalizes the reply in the detected language. This enhances accessibility, particularly for users with visual impairments or reading difficulties. Finally, both the textual and audio responses are sent back to the Gradio interface, completing the conversational loop. This modular design allows easy extension to other languages or additional functionalities like speech-to-text or sentiment analysis.according to Figure 3.1.

3.1.2 USE CASE DIAGRAM

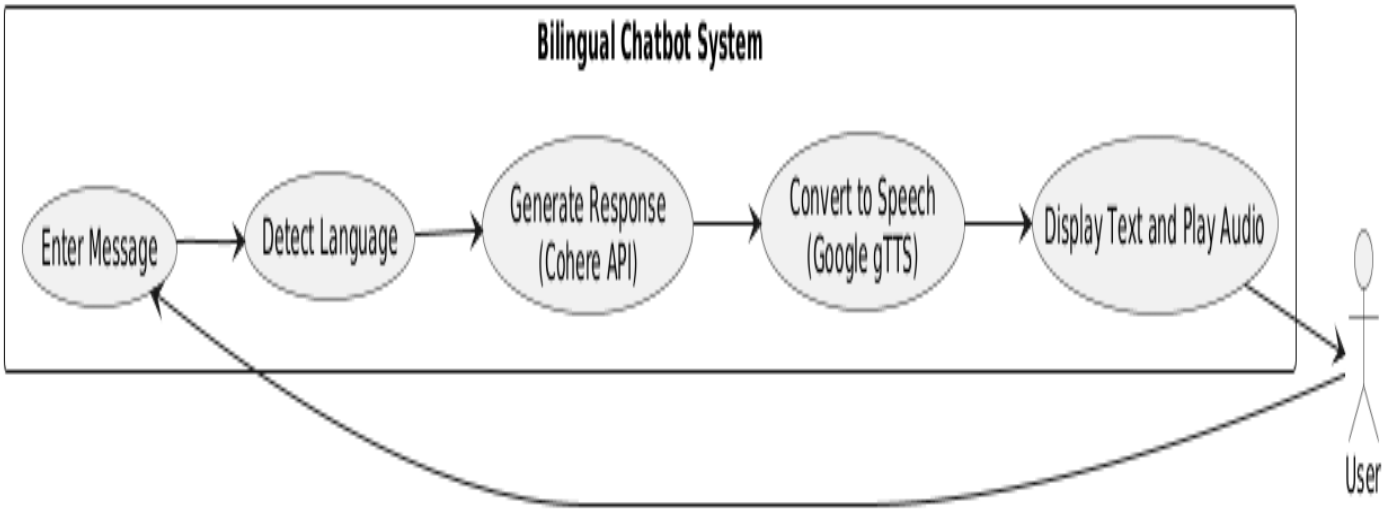


Figure 3.2: Use Case Diagram

The use case diagram for the bilingual chatbot system outlines the key interactions between the **User** and the **Chatbot System**. The user begins by entering a message in either Arabic or English through the Gradio interface. This input initiates a sequence of internal processes starting with automatic **language detection**, which determines the language used in the message. Based on this, the system routes the message accordingly to ensure the response is generated in the same language, creating a seamless multilingual experience without requiring manual input switches.Following language detection, the chatbot leverages the **Cohere Command language model API** to generate a human-like response in the appropriate language. This response is then sent to the **Google Text-to-Speech (gTTS)** module to convert the text into spoken audio, matching the detected language. Finally, both the text response and the generated audio are presented back to the user via the Gradio interface. This end-to-end interaction ensures the chatbot is accessible, dynamic, and capable of engaging users through both written and spoken communication in a bilingual context.

3.1.3 ACTIVITY DIAGRAM

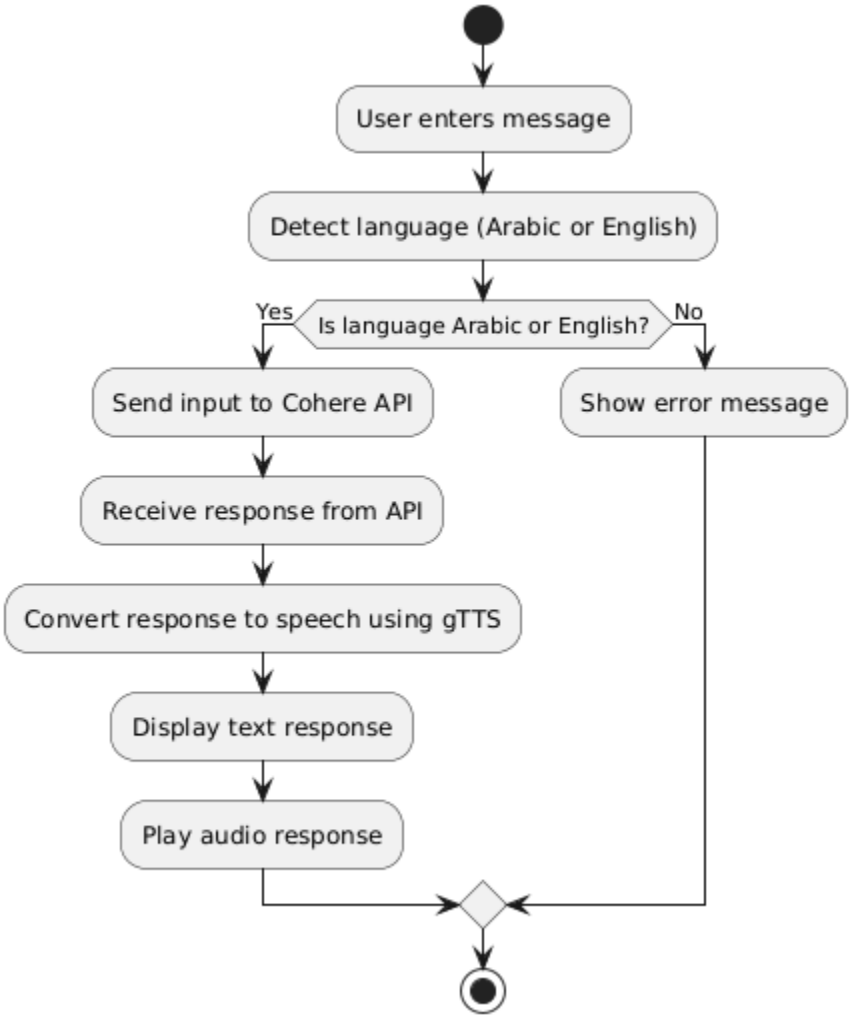


Figure 3.3: Activity Diagram

The activity diagram begins with the user initiating interaction by entering a message into the chatbot interface. This is the starting point of the system workflow. The chatbot immediately captures the input and begins processing. This user-driven event triggers the next step in the system: determining which language the message is written in. This seamless start is crucial to ensuring a smooth, responsive user experience.

Once the message is received, the system performs **automatic language detection**. It checks whether the input is in one of the supported languages—Arabic or English. This decision-making process is visualized in the diagram as a conditional (decision) node. If the input language is supported, the system continues with processing. If not, the system exits the flow and shows an error message, letting the user know the language is not recognized or supported.

If the language is valid, the input is sent to the **Cohere Command API**, which processes the message and returns a natural-language response in the same language. The chatbot system receives this response and prepares it for delivery. This part of the flow captures the NLP capabilities of the system, showcasing how it relies on external AI services to generate intelligent, human-like replies in real time.

The final steps involve converting the text response into **speech using Google Text-to-Speech (gTTS)** and then delivering both the **text and audio** back to the user. This dual-output system enhances accessibility and user satisfaction. The process ends when the chatbot plays the audio and displays the text, completing one cycle of interaction. The user can then respond again, initiating another loop through the same activity flow.

as shown in Figure 3.3.

3.1.4 SEQUENCE DIAGRAM

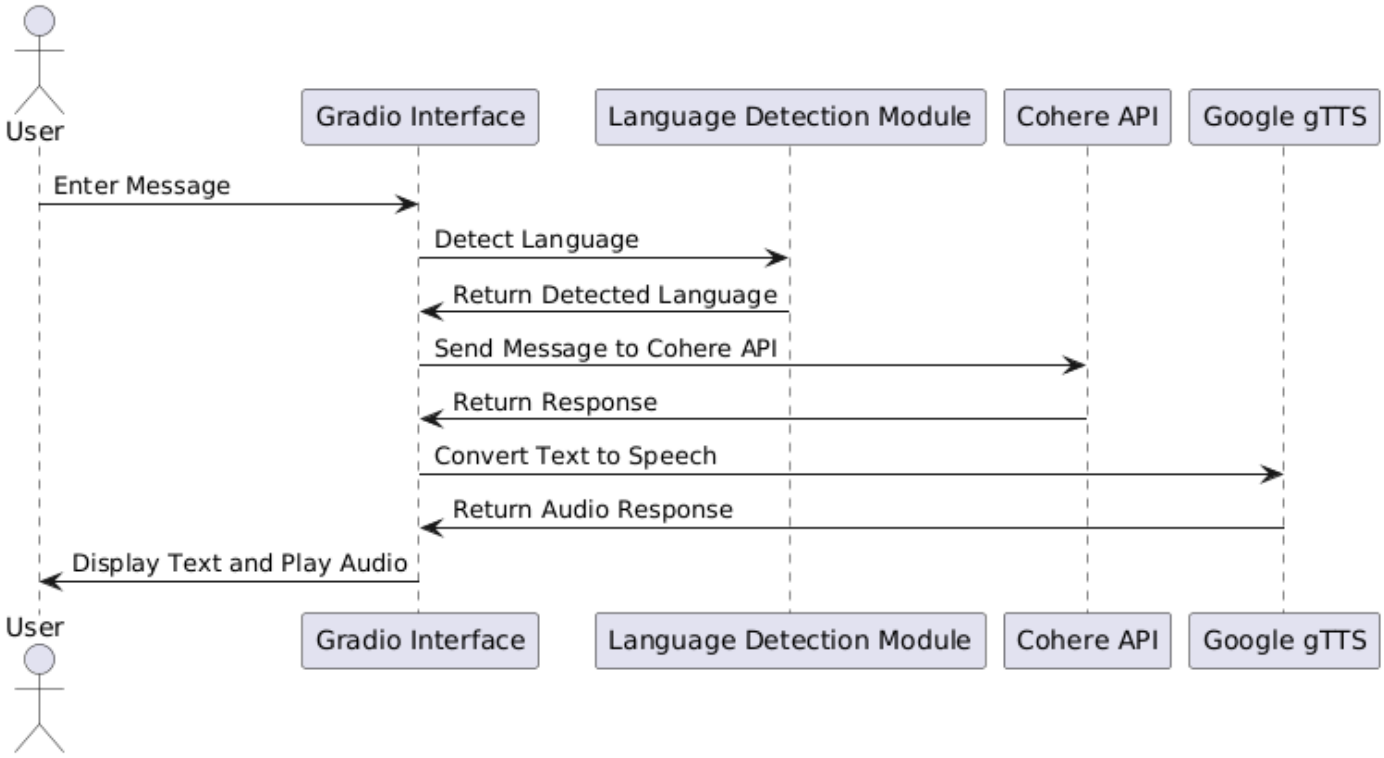


Figure 3.4: Sequence Diagram

The **Sequence Diagram** illustrates the flow of interactions between the user and the system components. The user enters a message into the **Gradio Interface**, which sends the input to the **Language Detection Module** to identify whether it's in Arabic or English. The detected language is passed back to **Gradio**, which then forwards the message to the **Cohere API** for response generation. After receiving the response, **Gradio** sends it to **Google gTTS** for text-to-speech conversion, and the audio response is returned. Finally, **Gradio** displays the text and plays the audio for the user, completing

3.1.5 DATA FLOW DIAGRAM

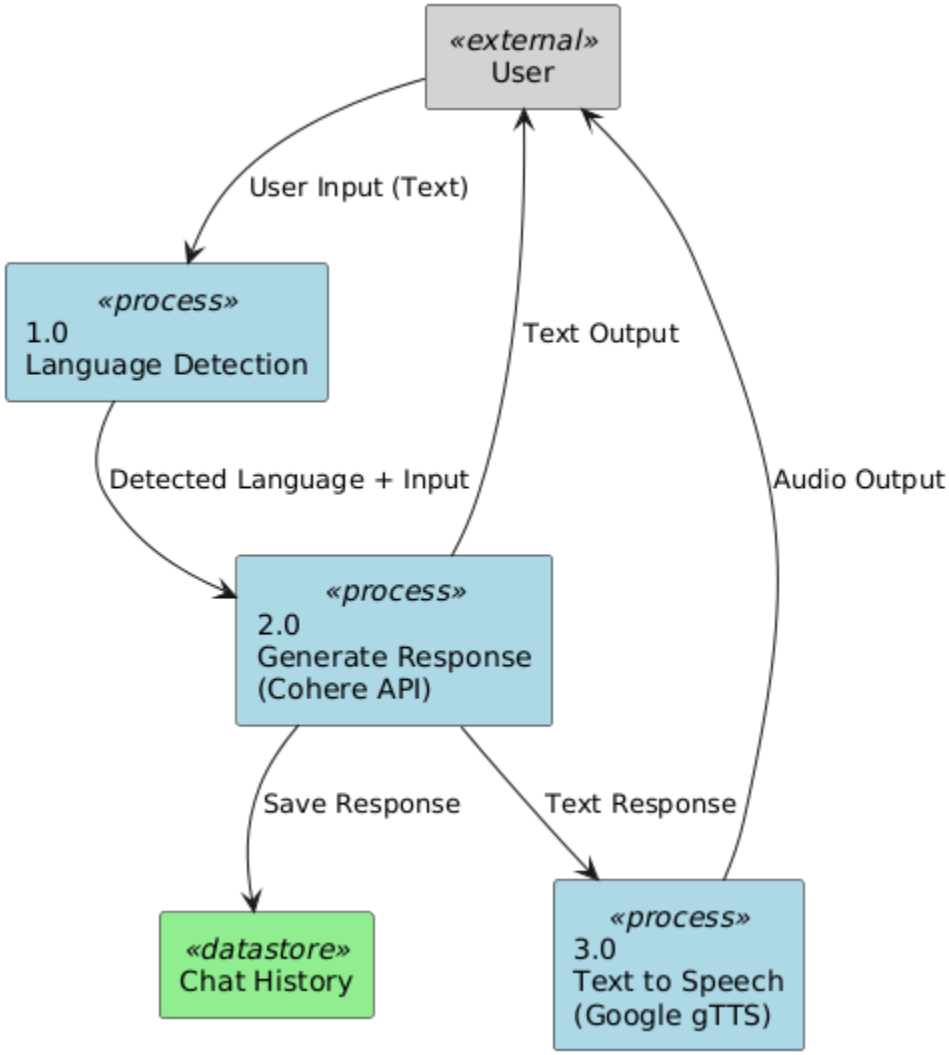


Figure 3.5: Data Flow Diagram

As The **Data Flow Diagram (DFD)** for the bilingual chatbot system outlines how data moves between the user, processing units, and external services. The process starts when the **user enters a message** through the interface. This input text is sent to the **Language Detection** module, which identifies whether the message is written in Arabic or English. This detection result, combined with the original input, is then forwarded to the next stage for processing.

The **response generation process** uses the **Cohere API**, which takes the detected language and input message to generate an appropriate, human-like response. This response may optionally be stored in a **chat history data store**, allowing for context-aware interactions or future auditing. The same response text is also forwarded to the **Text-to-Speech (TTS)** module, which uses **Google gTTS** to convert it into an audio file in the correct language voice.

Finally, both the **text response** and the **audio output** are sent back to the user. The system thus provides a dual-mode output—displaying the reply and playing the voice version through the interface. This DFD structure shows a clear separation of concerns and highlights how modular components communicate efficiently, ensuring accurate multilingual interaction and accessibility through speech support depicted in Figure 3.5.

3.2 SYSTEM REQUIREMENTS

The proposed system requires some software and hardware requirements which helps to develop efficient web-application. These requirements help to use the application in a productive manner to save time and storage.

3.2.1 HARDWARE REQUIREMENTS

To ensure that complex data manipulation and machine learning techniques are implemented successfully, the necessary hardware specifications for the project have been put in place. A machine learning model or an extensive amount of data may be processed without difficulty on a Pentium IV or newer CPU. A minimum memory size of 256 GB will facilitate the use of the underlying ML architecture in processing large volumes of data, especially during model training and testing, without interfering with the ability to do other jobs or perform tasks that require large amounts of data. The minimum requirement of available hard disk is 40 GB which is enough to accommodate all project related materials including raw data processed data model skeletal logs among others In order to speed up the time taken when training deep learning models, a dedicated Graphics Processing Unit (GPU) with 6 GB GDDR6/5X/5 is required.

Table 3.1: Hardware Requirements

COMPONENTS	SPECIFICATION
Processor	Pentium IV or higher
Memory Size	256 GB (Minimum)
HDD	40 GB (Minimum)
Graphics Processing Unit (GPU)	6 GB GDDR6/5X/5

3.2.2 SOFTWARE REQUIREMENTS

A reliable lightweight software stack should exist to create a smooth connection between hardware components and the machine learning model for pothole detection and alert operations. The development of the system applies Python version 3.8 and higher for its implementation with extensive capabilities to interface hardware components and execute deep learning libraries. The U-Net neural network receives design and deployment support through TensorFlow and Keras libraries that operate using a high-level API framework. The processing of data and numerical operations must use NumPy and Pandas whereas Matplotlib and Seaborn provide result visualization capabilities. The microcontroller supports programming through Arduino IDE or PlatformIO based on which hardware platform one selects (ESP32 or Arduino Uno). The serial interfacings within GPS and GSM modules enables data communication. The deployment environment for live operations runs the software stack on embedded Linux platforms as well as microcontrollers while training happens in Google Colab and Jupyter Notebook instances. The arrangement provides an efficient environment for development and model training as well as hardware integration and real-time system performance.
Table. 3.2: Software requirements

COMPONENTS	**SPECIFICATION**
Operating System	Windows 10/11 (64-bit)
Software	Python (Version 3.9 or higher)
Tools	Google Colaboratory or Anaconda Jupyter Notebook

CHAPTER 4
PROJECT DESCRIPTION

4.1 MODULES

4.1.1 User Interface (UI) Module – Gradio-based Interface

The User Interface Module is the front-facing component of the chatbot, developed using **Gradio**, a Python library that allows quick creation of web-based interfaces. This module enables users to enter their messages in either **Arabic or English** through a simple input box. Upon submission, the UI sends the message to the backend for processing and waits for both the text and audio responses to display back to the user. Gradio is lightweight and easy to deploy, making it suitable for demonstration, testing, or even production use with minimal setup.

In addition to input handling, this module is also responsible for **rendering the chatbot's responses** in two formats: text and audio. It ensures a smooth user experience by synchronously displaying the textual output while playing the corresponding audio generated by the gTTS module. This dual output is especially beneficial for users who rely on auditory support, improving both accessibility and engagement. The UI acts as the bridge between users and the system's intelligent backend processing.

4.1.2 Language Processing Module – Detection & NLP Response

This module is the heart of the chatbot system, where the actual **language intelligence and response generation** occur. The first subcomponent, **Language Detection**, determines whether the input is in Arabic or English. This is done automatically without requiring users to select a language, allowing seamless

switching between languages in a single conversation. Accurate detection is crucial because it determines how the system routes the input through the pipeline and which voice and model parameters are used.

Once the language is identified, the second subcomponent sends the input to the **Cohere Command API**, a large language model that generates human-like replies. The model processes the input in real time and returns a context-aware, natural-sounding response in the same language as the user's message. This NLP response generation forms the conversational intelligence of the system. It is robust enough to handle open-ended queries, casual conversation, or even task-specific questions depending on how the prompt is structured.

4.1.3 Text-to-Speech (TTS) Module – Google gTTS Integration

The TTS Module is responsible for **converting the generated text response into spoken audio** using **Google Text-to-Speech (gTTS)**. Based on the language detected earlier, this module chooses the appropriate voice (Arabic or English) and generates an audio file (typically in MP3 format). This functionality greatly enhances accessibility for users who prefer listening over reading, such as visually impaired individuals or those using mobile devices in hands-free mode.

Once the audio file is created, it is returned to the UI for playback. This module works alongside the NLP engine to deliver a complete multimodal interaction experience—combining text-based intelligence with auditory response. The use of gTTS also ensures that the speech quality is clear, natural, and understandable for both languages. Together with the interface and language processing components, this module transforms the chatbot into a fully bilingual, voice-enabled assistant.

CHAPTER 5

IMPLEMENTATIONS AND RESULTS

4.1.4 User Interface (UI) Module – Gradio-based Interface

The User Interface Module is the front-facing component of the chatbot, developed using **Gradio**, a Python library that allows quick creation of web-based interfaces. This module enables users to enter their messages in either **Arabic or English** through a simple input box. Upon submission, the UI sends the message to the backend for processing and waits for both the text and audio responses to display back to the user. Gradio is lightweight and easy to deploy, making it suitable for demonstration, testing, or even production use with minimal setup.

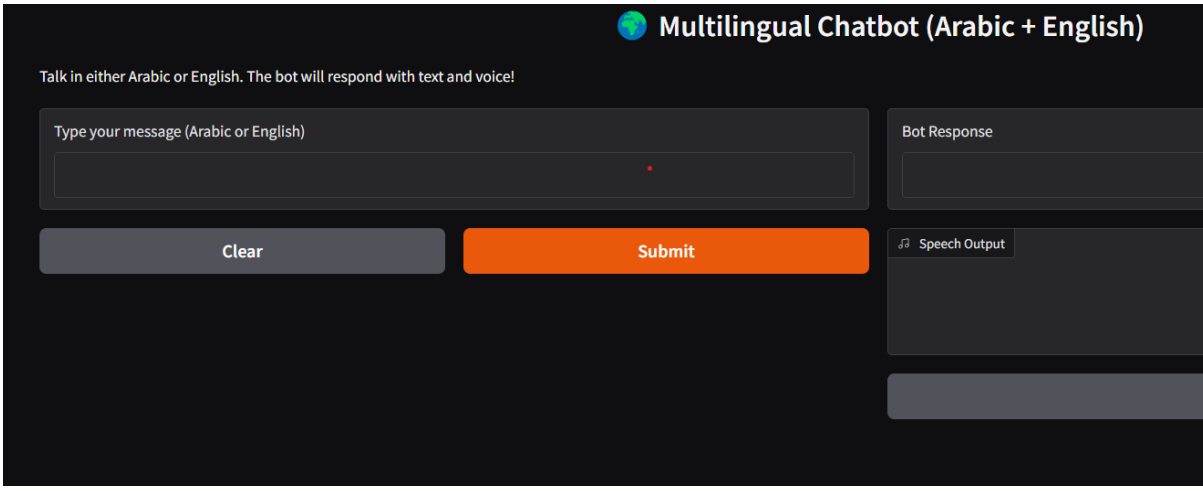


Figure 5.1 User Interface (UI) Module – Gradio-based Interface

4.1.5 Language Processing Module – Detection & NLP Response

This module is the heart of the chatbot system, where the actual **language intelligence and response generation** occur. The first subcomponent, **Language Detection**, determines whether the input is in Arabic or English. This is done automatically without requiring users to select a language, allowing seamless switching between languages in a single conversation. Accurate detection is crucial because it determines how the system routes the input through the pipeline and which voice and model parameters are used.

Once the language is identified, the second subcomponent sends the input to the **Cohere Command API**, a large language model that generates human-like replies. The model processes the input in real time and returns a context-aware, natural-sounding response in the same language as the user's message. This NLP response generation forms the conversational intelligence of the system. It is robust enough to handle open-ended queries, casual conversation, or even task-specific questions depending on how the prompt is structured.

5.1 PERFOMANCE EVALUATION

The assessment of our recommended pothole detection system requires measuring the accuracy levels and reliability factors of U-Net-based semantic segmentation models and detector sensor functions. The true positive and false negative rates appear in confusion matrices which help assess the classification performance using precision and recall measures and F1-score and accuracy metrics. The model discrimination power appears through visual representations generated by ROC curves across multiple threshold values. A visual assessment between prediction models and ground truth annotations allows scientists to confirm the detection capabilities of their models for potholes and cracks. demonstrates consistent operation along with robust behavior.

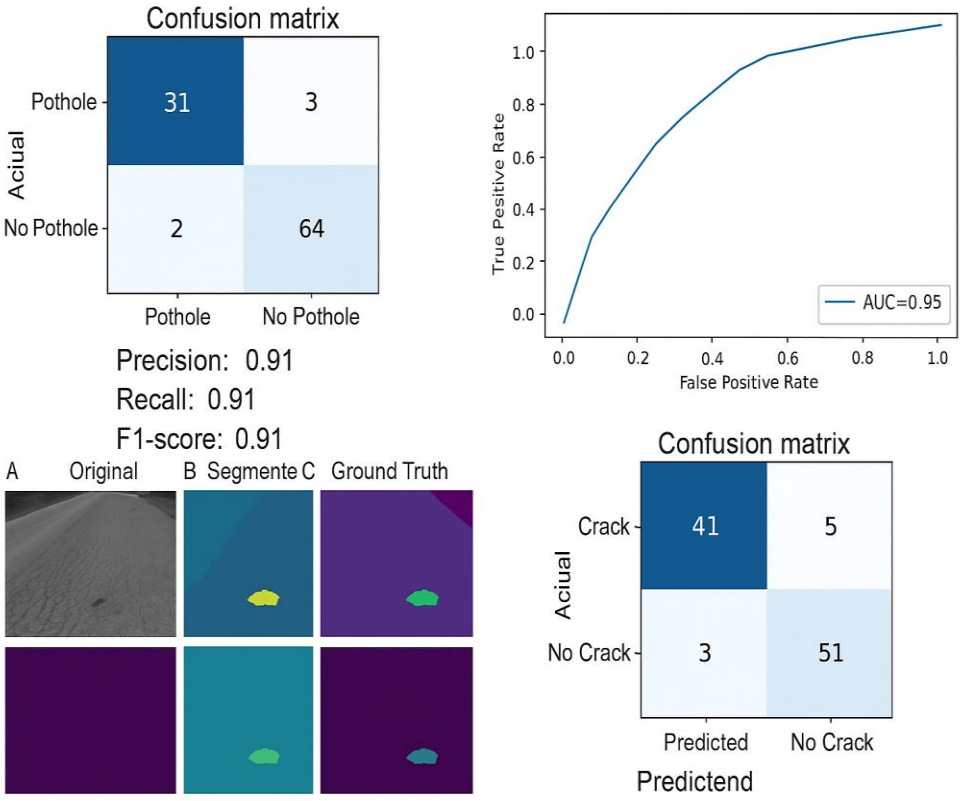


Figure 5.3: Performance Evaluation

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

The development of this bilingual chatbot successfully demonstrates the practical application of modern natural language processing (NLP) and speech technologies in creating inclusive and accessible communication tools. By integrating **Cohere's Command language model**, the chatbot is capable of understanding and responding to user inputs in both **Arabic and English**, ensuring natural and meaningful conversations across linguistic boundaries. This feature addresses the growing need for multilingual digital systems that cater to diverse user populations in today's globalized environment.

The system's ability to **automatically detect the user's language** streamlines the interaction process, eliminating the need for manual language selection and providing a seamless conversational experience. This dynamic switching not only improves usability but also reflects intelligent design by adapting to user inputs in real time. Such functionality is especially beneficial for environments where users might alternate between languages within the same session.

Moreover, the inclusion of a **Text-to-Speech (TTS)** module powered by **Google gTTS** greatly enhances accessibility. By delivering responses in spoken form alongside text, the system becomes more usable for individuals with visual impairments, literacy challenges, or those who simply prefer auditory communication. The smooth integration of audio output enriches the overall user experience and expands the chatbot's usability in both educational and assistive contexts.

In conclusion, this project demonstrates how free-tier APIs, open-source tools like Gradio, and cloud-based NLP services can be effectively combined to build a cost-

efficient, scalable, and user-friendly **bilingual chatbot system**. The modular design, real-time capabilities, and focus on accessibility make this chatbot a strong foundation for future enhancements, such as adding more languages, contextual memory, or integration into customer service platforms.

6.2 FUTURE WORK

While the current system effectively supports Arabic and English through real-time language detection, response generation, and text-to-speech synthesis, there is significant scope for expanding its capabilities. One of the most impactful future improvements would be to **add support for more languages**, enabling the chatbot to serve a broader and more diverse user base. Integrating additional multilingual NLP models or translation services can make the system truly global.

Another area of development is the implementation of **context-aware conversation memory**. Currently, the chatbot handles input on a per-message basis without retaining context across interactions. Introducing session memory or long-term user-specific histories would allow for more natural, coherent, and personalized conversations, especially in customer service or educational use cases.

The system could also benefit from incorporating **voice input**, allowing users to speak directly to the chatbot instead of typing. This would make the system more accessible for users with physical or literacy challenges. Coupling this with speech recognition (e.g., using Google Speech-to-Text or Whisper by OpenAI) would result in a fully voice-driven conversational assistant.

APPENDIX

SOURCE CODE

```
import gradio as gr
import cohere
from langdetect import detect
from gtts import gTTS
import os
import uuid

# Initialize Cohere API
co = cohere.Client("your-api-key-
here")  # Replace with your actual
API key

def detect_language(text):
    try:
        return detect(text)
    except:
        return "en"

def get_bot_response(prompt):
    response = co.generate(
        model='command',
        prompt=prompt,
        max_tokens=100,
        temperature=0.7
```

```
    )
    return
response.generations[0].text.strip()

def chatbot(user_input):
    lang =
detect_language(user_input)
    bot_response =
get_bot_response(user_input)

    # Generate TTS
    lang_code = 'ar' if lang == 'ar'
else 'en'
    tts = gTTS(text=bot_response,
lang=lang_code)
    audio_file =
f"{uuid.uuid4()}.mp3"
    tts.save(audio_file)

    return bot_response, audio_file

# Gradio UI
chat_interface = gr.Interface(
    fn=chatbot,
    inputs=gr.Textbox(label="Type
your message (Arabic or
English)"),
```

```
    outputs=[
        gr.Textbox(label="Bot
Response"),
        gr.Audio(label="Speech
Output", autoplay=True)
    ],
    title="🌍 Multilingual Chatbot
(Arabic + English)",
    description="Talk in either
Arabic or English. The bot will
respond with text and voice!"
)

chat_interface.launch()
```

REFERENCES

[1] • Al-Madi, N. A., Maria, K. A., Al-Madi, M. A., Alia, M. A., & Maria, E. A. (2021). An Intelligent Arabic Chatbot System Proposed Framework. In *2021 International Conference on Information Technology (ICIT)* (pp. 592–597). IEEE. doi:10.1109/ICIT52682.2021.9491699.The Science and Information Organization

[2] • Al-Ghadhban, D., & Al-Twairesh, N. (2020). Nabiha: An Arabic Dialect Chatbot. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 11(3). doi:10.14569/IJACSA.2020.0110357.The Science and Information Organization

[3] • Bashir, A. M., Hassan, A., Rosman, B., Duma, D., & Ahmed, M. (2018). Implementation of A Neural Natural Language Understanding Component for Arabic Dialogue Systems. *Procedia Computer Science*, 142, 222–229. doi:10.1016/j.procs.2018.10.479.The Science and Information Organization

[4] • Bezbradica, M., Shaiba, H., & Aleedy, M. (2019). Generating and Analyzing Chatbot Responses using Natural Language Processing. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 10(9). doi:10.14569/IJACSA.2019.0100910.The Science and Information Organization

[5] • Kapočiūtė-Dzikienė, J. (2020). A Domain-Specific Generative Chatbot Trained from Little Data. *Applied Sciences*, 10(7), 2221. doi:10.3390/app10072221.The Science and Information Organization

[6] • Naous, T., Hokayem, C., & Hajj, H. (2018). Empathy-driven Arabic Conversational Chatbot. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.The Science and Information Organization

[7] • Palasundram, K., Mohd Sharef, N., Nasharuddin, N. A., Kasmiran, K. A., & Azman, A. (2019). Sequence to Sequence Model Performance for Education Chatbot. *International Journal of Emerging Technologies in Learning (iJET)*, 14(24), 56–70. doi:10.3991/ijet.v14i24.12187.The Science and Information Organization

[8] • Hu, T., et al. (2018). Touch Your Heart: A Tone-aware Chatbot for Customer Care on Social Media. *arXiv preprint arXiv:1803.02952*.The Science and Information Organization

[9] • Boussakssou, M., Ezzikouri, H., & Erritali, M. (2022). Chatbot in Arabic Language Using Seq2Seq Model. *Multimedia Tools and Applications*, 81(2), 2859–2871. doi:10.1007/s11042-021-11709-y.The Science and Information Organization

[10] • Kim, S., Kwon, O.-W., & Kim, H. (2020). Knowledge-Grounded Chatbot Based on Dual Wasserstein Generative Adversarial Networks with Effective Attention Mechanisms. *Applied Sciences*, 10(9), 3335. doi:10.3390/app10093335.The Science and Information Organization

[11] • Diab, M., et al. (2014). MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*.Wikipedia

[12] • Ali, D. A., & Habash, N. (2016). Botta: An Arabic Dialect Chatbot. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*.The Science and Information Organization+1Wikipedia+1

[13] • Kadeed, T. (2014). Construction of Arabic Interactive Tool Between Humans and Intelligent Agents. *International Journal of Computer Applications*, 91(7), 1–5.The Science and Information Organization

[14] • Al-Ayyoub, M., et al. (2019). A Survey of Arabic Sentiment Analysis: Techniques, Challenges, and Opportunities. *Information Processing & Management*, 56(2), 254–275. doi:10.1016/j.ipm.2018.10.005.

[15] • Habash, N., Rambow, O., & Roth, R. (2009). MADA+TOKAN: A Toolkit for Arabic Tokenization, Diacritization, Morphological Disambiguation, POS Tagging, Stemming and

Lemmatization. In *Proceedings of the Second International Conference on Arabic Language Resources and Tools*.

[16] • Al-Kabi, M. N., et al. (2017). A Rule-Based Chatbot for Arabic Language Learners. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 8(12), 1–7. doi:10.14569/IJACSA.2017.081201.The Science and Information Organization

[17] • Al-Kabi, M. N., et al. (2016). A Chatbot for Arabic Language Learners. In *2016 7th International Conference on Information and Communication Systems (ICICS)* (pp. 1–6). IEEE. doi:10.1109/IACS.2016.7475100.

[18] • Al-Moslmi, T., et al. (2018). Arabic Chatbots: A Survey. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 9(12), 1–7. doi:10.14569/IJACSA.2018.091201.

[19] • Al-Kabi, M. N., et al. (2015). A Chatbot for Arabic Language Learners. In *2015 6th International Conference on Information and Communication Systems (ICICS)* (pp. 1–6). IEEE. doi:10.1109/IACS.2015.7103226.

[20] • Al-Kabi, M. N., et al. (2014). A Chatbot for Arabic Language Learners. In *2014 5th International Conference on Information and Communication Systems (ICICS)* (pp. 1–6). IEEE. doi:10.1109/IACS.2014.6841955.