Smart AI Chatbot

STEVEN MARK KHRISTI

O23MCA110279

Abstract

AI chatbot built using Python, NLTK, and SpeechRecognition.

Accepts both text and voice inputs.

Responds with pre-defined templates based on intent.

Provides GUI using Tkinter and logs conversation history.

Problem Definition

Most systems rely on structured input and rigid responses.

Need for intelligent systems to simulate human-like conversations.

Existing chatbots lack robust NLP and flexibility.

Aim: Develop chatbot with text/voice input and context-aware responses.

Objective

Demonstrate use of AI and NLP for basic conversational assistants.

Improve human-computer interaction via voice/text.

Use NLTK for processing and Tkinter for GUI.

Log conversations for review and improvement.

Hardware Requirements

Intel Core i3 or above, 4GB RAM, 500MB storage.

Microphone for voice input.

Standard display monitor.

Software Requirements

Python 3.8+, NLTK, speechrecognition, Tkinter.

Windows/Linux/macOS, VS Code/PyCharm/IDLE.

Google Speech API, internet connection.

SDLC of the Project

Requirement Analysis: Define chatbot scope and features.

System Design: GUI and processing logic architecture.

Implementation: Python modules for GUI, NLP, voice.

Testing: Unit, integration, system testing.

Deployment: Local desktop-based setup.

Maintenance: Future upgrades with ML and database.

System Design & Architecture

Modular architecture with GUI, NLP, and voice modules.

Tkinter-based GUI with text and voice input.

NLP using NLTK for intent matching.

Logs conversation to a text file.

GUI Design & Flow

Scrolled text area for chat display.

Entry box, Send and Voice buttons.

Processes text/voice input → identifies intent → gives response.

Chat logged to file.

Code & Logic Highlights

Intent Dictionary: Predefined intents with keywords and responses.

get_intent(): Matches keywords from input.

get_response(): Picks a random response based on intent.

Tkinter GUI for chat interface.

Testing Strategy

Unit Testing: Core functions (intent, response, logging).

Integration Testing: Voice + GUI + NLP.

System Testing: Windows 10/11 with required libs.

Functional, Regression, Usability Testing.

Literature Survey

Existing: Manual or biometric systems, contact-based.

Proposed: Contactless AI-based chatbot system.

Uses NLP, GUI, and voice input for better interaction.

Feasibility Study

Technical: Uses stable, open-source tools.

Economic: Cost-effective, scalable, low maintenance.

Tools: Python, OpenCV, MySQL (optional).

System Analysis

Functional: Accepts text/voice, identifies intent, logs chat.

Non-functional: Usable, fast (<1 sec), modular, reliable.

Portable across platforms with Python setup.

Data Flow Diagram

User Input → NLP → Intent Match → Response → GUI Display → Chat Log

Use Case Diagram

User types or speaks → Bot processes → Responds on GUI.

Actors: User, Bot

Processes: Input, NLP, Response generation, Display

Applications

Educational FAQs assistant.

Customer Support automation.

Personal voice/text assistant.

Information desk / accessible communication.

Academic demonstration of NLP + GUI + voice tech.

Conclusion & Future Scope

Successful integration of NLP and voice in a GUI chatbot.

Improves digital interactions using AI.

Future: ML-based dynamic responses, database, mobile/web deployment.

References

NLTK Book - nltk.org

Google Cloud Speech-to-Text

Python 3 Docs - python.org

TkDocs.com, Real Python, Stack Overflow

AI Chatbots by Rao, Packt Publishing