Project Report: Offline Voice Assistant using Vosk and Python

1. Title

Offline Voice Assistant Using Vosk, PyAudio, and Tkinter GUI in Python

2. Abstract

This project presents the design and implementation of an **offline voice assistant** capable of recognizing speech and responding to predefined queries using natural voice synthesis. The assistant is built using the **Vosk speech recognition library**, **PyAudio** for capturing audio input, and **pyttsx3** for offline text-to-speech. A simple **Tkinter GUI** is also implemented to display status updates, recognized queries, and corresponding responses in real-time. The system functions without internet connectivity, making it suitable for secure or limited-access environments.

3. Objectives

- Implement a fully offline voice-controlled assistant.
- Recognize and interpret voice commands using Vosk.
- Provide audio responses using pyttsx3.
- Display status and interaction logs using a GUI.
- Maintain a log of user queries and system responses.

4. Tools and Technologies Used

Tool/Library	Purpose
Python 3.x	Programming Language
Tkinter	GUI development
Vosk	Offline speech recognition engine
PyAudio	Audio stream handling from microphone
pyttsx3	Offline text-to-speech engine
difflib	Approximate string matching for fuzzy queries
threading	To prevent UI blocking during recognition

5. System Architecture

Modules:

1. **VoiceAssistant Class**

- Handles microphone input, speech recognition, query matching, response generation, and logging.

2. **AssistantGUI Class**
   - Provides a simple, clean interface for displaying system status, user input, and assistant response.
3. **Keyword Detection & Fuzzy Matching**
   - Recognizes key phrases (e.g., "cobot", "AI") and handles unknown queries using difflib.
4. **Offline Models**
   - Utilizes vosk-model-small-en-in-0.4, an Indian English acoustic model, for accurate recognition without internet.

6. Working Principle

1. The assistant waits in standby mode until it hears the keyword **"assistant"**.
2. Upon activation, it listens for a command.
3. The input speech is converted to text using the **Vosk recognizer**.
4. The query is matched with predefined commands.
5. The corresponding response is:
   - Shown in the GUI.
   - Spoken out using pyttsx3.
   - Logged in a text file with timestamps.
6. The system returns to standby mode after a response.

7. Features

- Works entirely offline.
- Clean and responsive GUI.
- Keyword and fuzzy-based query matching.
- Logging of interactions to conversation_log.txt.
- Voice feedback via text-to-speech.
- Error handling for failed recognitions or model loading.

HIGH-LEVEL DIFFERENCE:

Feature	Earlier Version	Improved Version (This One)
Response Type	MP3 Audio Files (pre-recorded)	Fully dynamic TTS (speaks using pyttsx3)
GUI Interface	None	Yes (with status, input, and response shown)
Logging	No logs saved	Saves logs with timestamps in a .txt file
Speech Accuracy	Only keyword matching	Keyword + close match with difflib (fuzzy logic)
Listening Feedback	None	GUI shows real-time status and timestamps
Activation Word	Yes, hardcoded word "assistant"	Still present with standby and active mode GUI
Internet Check	Has fallback between online/offline	This version is fully offline for now
Streaming Handling	Opened once at start	Smart stream start/stop to save resources
Custom Error Messages	Minimal	Detailed error logging if model not found etc.

8. Sample Predefined Commands

User Query Keywords	Assistant Response
"what is a cobot"	A cobot is a collaborative robot designed to work safely with humans.
"what is ai"	AI stands for Artificial Intelligence...
"tell me a joke"	Why did the robot go on vacation? It needed to recharge!
"bye", "exit", "quit"	Goodbye! Have a great day.

9. Advantages

- No dependency on internet services (unlike Google Assistant or Alexa).
- Ensures user privacy as audio stays local.
- Lightweight and customizable.
- Educational value for understanding ASR and GUI integration.

10. Limitations & Future Scope

Limitation	Possible Improvement
Only predefined responses	Add GPT-based or dynamic response generation
No continuous conversation context	Add NLP state management
Fixed commands in code	Allow external JSON configuration
No command to stop assistant from GUI	Add GUI-based "Stop/Exit" button

11. Screenshots

- GUI Interface showing "Listening" and "Response"
- Console logs or conversation_log.txt sample

12. Conclusion

This project successfully demonstrates an offline voice assistant built with open-source Python libraries. It emphasizes privacy, offline accessibility, and customization. The assistant can be enhanced further for industrial, educational, or home automation use cases.