Jarvis AI

🧠 Mega Project 1: Jarvis AI – Personal Voice Assistant

Level: Intermediate

Goal: Create a smart voice assistant capable of performing system tasks, web browsing, playing music, fetching news, and answering general queries using Gemini AI.

❓ What is the Problem Statement?

> Build a Python-based personal voice assistant, "Jarvis", that:

- Understands voice input
- Executes various commands (system tasks, browsing, music, apps)
- Fetches news from News API
- Integrates with **Google Gemini AI** for natural responses

📋 Requirements & Installation

🛠️ Modules to Install

```
#Shell Script
pip install speechrecognition
pip install pyttsx3
pip install requests
pip install google-generativeai
pip install pocketJarvis
```

📄 Files & Code Walkthrough

🔷 File: `musicLibrary.py`

```
# Dictionary of music keywords and their YouTube links
music = {
    "bones" : "https://www.youtube.com/watch?v=YyV2k8Almuk",
    "deva"  : "https://www.youtube.com/watch?v=mNuhKUOD_A0",
    "kesariya" : "https://www.youtube.com/watch?v=BddP6PYo2gs"
}
```

✅ Simple music library to map keywords to songs.

🔷 File: `main.py`

📌 Key Functionalities:

- Voice input using `speech_recognition`
- Speak back with `pyttsx3`
- Uses Gemini AI (via Google Generative AI) for intelligent replies
- Can open websites and perform system tasks
- Plays music from `musicLibrary`

- Fetches latest news using **NewsAPI**

🔐 Note on API Keys:

🔒 API Keys have been replaced.

API Keys have been removed for privacy.

```
genai.configure(api_key="YOUR_GEMINI_API_KEY")
api_key = "YOUR_NEWS_API_KEY"
```

✅ Full Code with Comments: main.py


```
import speech_recognition as sr
import webbrowser
import pyttsx3
import musicLibrary
import requests
from datetime import datetime
import google.generativeai as genai
import os

# Config Gemini AI – Don't expose real API key
genai.configure(api_key="YOUR_GEMINI_API_KEY")
gemini_model = genai.GenerativeModel("gemini-1.5-flash")

# Initialize recognizer and text-to-speech
recognizer = sr.Recognizer()
engine = pyttsx3.init()

# NewsAPI key – replace before publishing
api_key = "YOUR_NEWS_API_KEY"

def speak(text):
    engine.say(text)
    engine.runAndWait()
```

```
# Main command handler
def processCommand(c):
    if "open google" in c.lower():
        webbrowser.open("https://google.com")
    elif "open youtube" in c.lower():
        webbrowser.open("https://youtube.com")
    elif "open facebook" in c.lower():
        webbrowser.open("https://facebook.com")
    elif "open whatsapp" in c.lower():
        webbrowser.open("https://web.whatsapp.com")
    elif "open github" in c.lower():
        webbrowser.open("https://github.com")
    elif "open instagram" in c.lower():
        webbrowser.open("https://instagram.com")
    elif "open amazon" in c.lower():
        webbrowser.open("https://amazon.com")
    elif "open flipkart" in c.lower():
        webbrowser.open("https://flipkart.com")
    elif "open twitter" in c.lower():
        webbrowser.open("https://twitter.com")
    elif "open myntra" in c.lower():
        webbrowser.open("https://myntra.com")
    elif "open hotstar" in c.lower():
        webbrowser.open("https://hotstar.com")
    elif "open notepad" in c.lower():
        os.system("notepad")
    elif "open code" in c.lower():
        os.system("code")  # if VS Code is added to PATH
    elif "open calculator" in c.lower():
        os.system("calc")
    elif "open paint" in c.lower():
        os.system("mspaint")
    elif "open command prompt" in c.lower() or "open cmd" in c.lower():
        os.system("start cmd")
    elif "open control panel" in c.lower():
```

```
        os.system("control")
    elif "open task manager" in c.lower():
        os.system("taskmgr")
    elif "open file explorer" in c.lower():
        os.system("explorer")
    elif "open settings" in c.lower():
        os.system("start ms-settings:")
    elif "open chrome" in c.lower():
        speak("Chrome is not installed")
    elif "open microsoft edge" in c.lower() or "open edge" in c.lower():
        os.system("start msedge")
    elif "open word" in c.lower():
        os.system("start winword")
    elif "open excel" in c.lower():
        os.system("start excel")
    elif "open chat gpt" or "open chatgpt" in c.lower():
        # Fun restriction for ChatGPT
        speak("I hate ChatGPT, so I won't open it. Use me instead!")

    # NEWS FEATURE
    elif "news" in c.lower():
        url = f"https://newsapi.org/v2/everything?q=india&sortBy=publishedAt&l
anguage=en&apiKey={api_key}"
        r = requests.get(url)
        if r.status_code == 200:
            data = r.json()
            articles = data.get('articles', [])
            if articles:
                for article in articles[:5]:
                    print(article['title'])
                    speak(article['title'])
            else:
                speak("No news articles available right now.")
        else:
            speak("Sorry, I'm unable to fetch news right now.")
```

```
    # MUSIC FEATURE
    elif "play" in c.lower():
        command_lower = c.lower()
        for song_name in musicLibrary.music:
            if song_name in command_lower:
                speak(f"Playing {song_name}")
                webbrowser.open(musicLibrary.music[song_name])
                break
        else:
            speak("Sorry, I couldn't find that song in your music library.")

    # FALLBACK TO AI
    else:
        response = gemini_model.generate_content(c)
        print("Jarvis:", response.text)
        speak(response.text)

# Main loop
if __name__ == "__main__":
    speak("Initializing Jarvis")
    while True:
        # Listen for input via mic
        r = sr.Recognizer()
        with sr.Microphone() as source:
            print("Listening...")
            audio = r.listen(source)

        print("Recognising...")
        try:
            command = r.recognize_google(audio)
            print(f"User: {command}")
            if "shutdown" in command.lower():
                speak("Goodbye!")
                break
            processCommand(command)
        except sr.UnknownValueError:
```

```
            print("Jarvis could not understand audio")
        except Exception as e:
            print("Jarvis error; {0}".format(e))

print("Program closed successfully.")
```

🧠 Concepts Used

Feature	Python Concept
Voice recognition	`speech_recognition`
Text-to-speech	`pyttsx3`
Web navigation	`webbrowser`
Gemini AI integration	`google.generativeai`
News fetching	`requests` & News API
Music library	Python dict, YouTube links
OS automation	`os.system`

🚨 Improvements & Future Add-ons

- 🎙️ Wake-word support (only trigger on "Hey Jarvis")
- 🧠 Gemini long-term memory (contextual convos)
- 📂 File reading, summarizing PDFs, mouse and keyboard access
- 🔐 Secure config with `.env` files
- 🧪 Add GUI for fallback typing