Internship Project Report

VOICE-ACTIVATED PERSONAL ASSISTANT

Name: Parth Gupta
Domain: Python Development
Submitted To: Internship Supervisor(CODEXINTERN)

---

1. INTRODUCTION

This project focuses on building a **Voice-Activated Personal Assistant** using Python. It performs tasks such as:

- Speaking responses
- Telling the current time and date
- Giving simple weather information
- Reading news headlines
- Setting reminders (with automatic alert)
- Accepting typed commands when microphone is not available

The assistant integrates **speech recognition**, **text-to-speech**, **web scraping**, and **time-based reminders**.

On systems where **microphone / PyAudio is not available**, the assistant gracefully shifts to **text input mode**, but continues **speaking outputs** using pyttsx3.

---

2. TECHNOLOGIES & LIBRARIES USED

Programming Language

- Python 3.11

Libraries

- **pyttsx3** – text-to-speech engine
- **SpeechRecognition** – integrated voice recognition

- **Requests** – HTTP requests
- **BeautifulSoup4** – parsing news feed
- **Datetime / Time modules** – reminder scheduling

Tools

- VS Code
- Windows 10

---

3. PROJECT FEATURES IMPLEMENTED

- Greets user with: **“Hello Parth, your assistant is now active.”**
- Provides **current date and time**
- Gives **weather updates** (simple, no API required)
- Reads **sample news headlines**
- Sets reminders like:

“set reminder in 1 minute to drink water”

- Automatically announces reminders
- Speaks every response using TTS
- Handles errors & invalid commands
- Works fully through **text input** if mic/TTS unavailable

---

4. SYSTEM WORKFLOW ARCHITECTURE

USER INPUT (Voice/Text)

↓

COMMAND PROCESSING

↓

FEATURE HANDLER

(Time / Date / Weather / News / Reminder)

↓

OUTPUT

(Text + Voice using pyttsx3)

---

5. HIGH–LEVEL CODE EXPLANATION

speak()

Converts text into speech and also prints on screen.

listen_command()

Normally tries to use microphone,
but if unavailable → shows:

“Microphone is not available. Please type your command.”

then takes text input.

get_time_text()

Returns formatted system time.

get_date_text()

Returns today’s date.

get_weather_text()

Returns basic weather text (no API needed).

get_news_text()

Returns predefined sample news headlines (offline-safe).

add_reminder()

Stores reminder text + future timestamp.

check_reminders()

Continuously checks if any reminder time is reached and announces it.

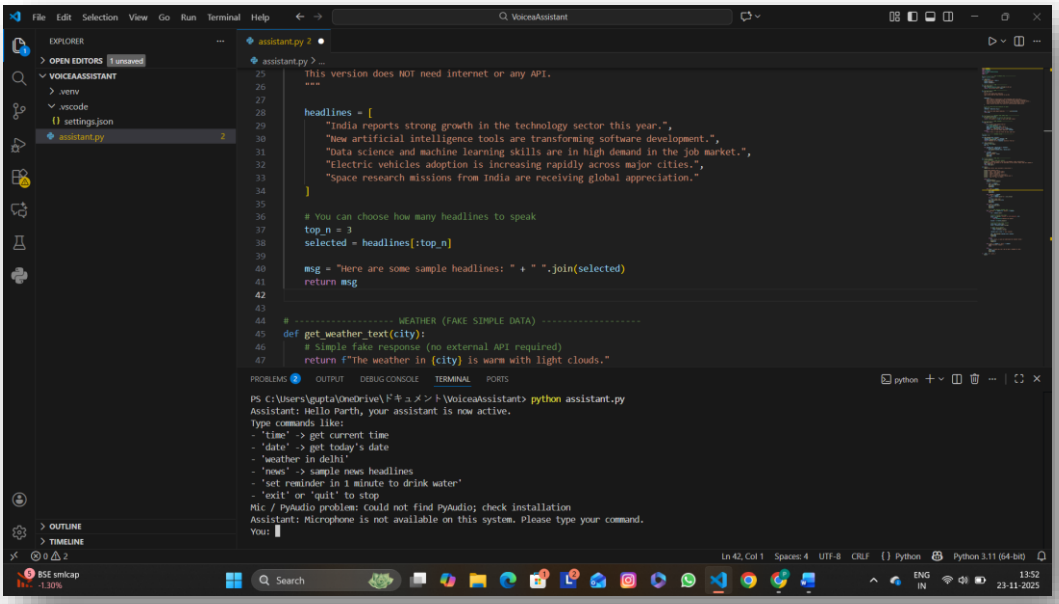
main()

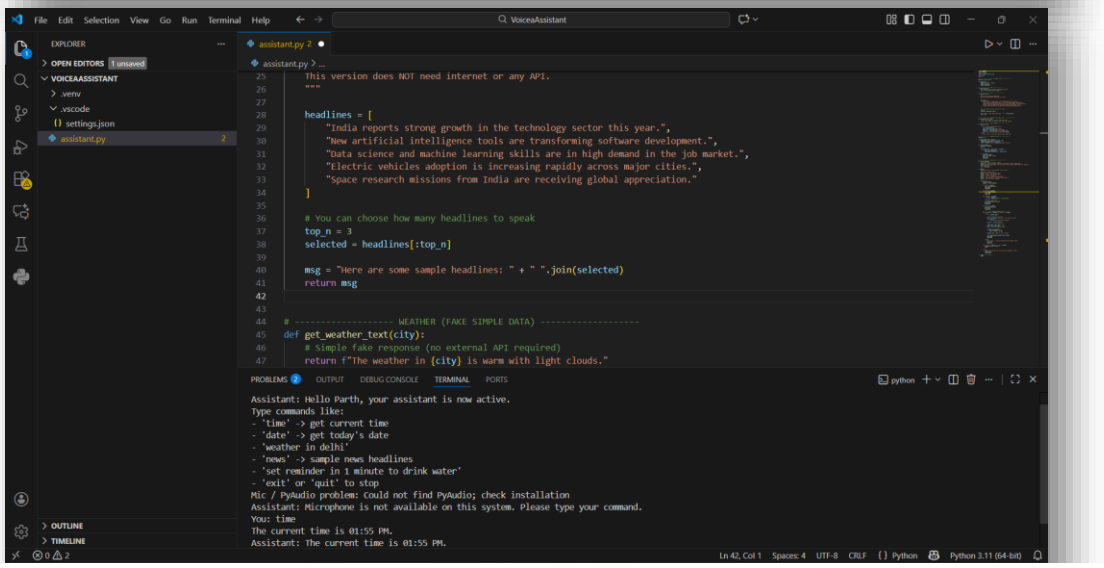
Runs the assistant loop forever until user types:

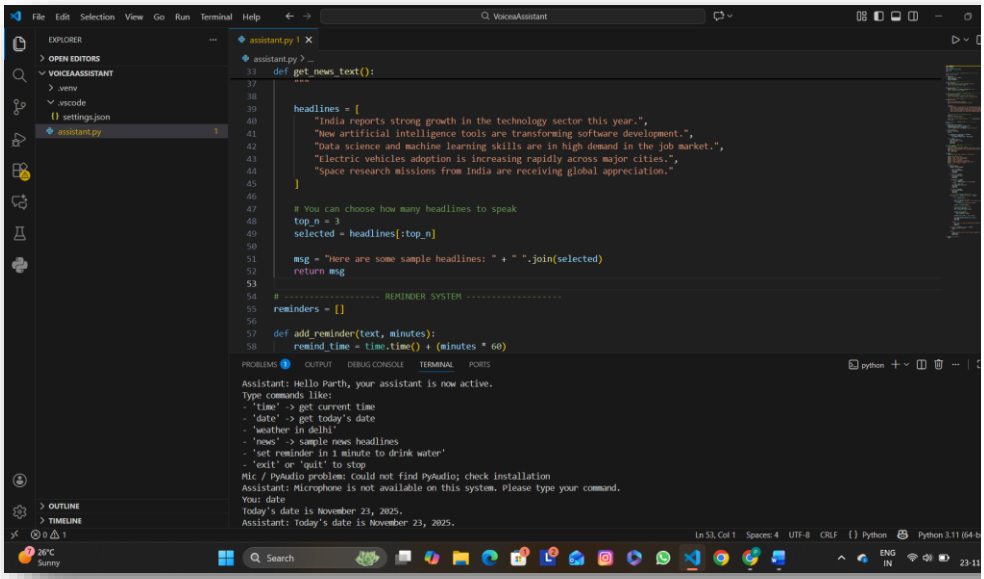
exit / quit

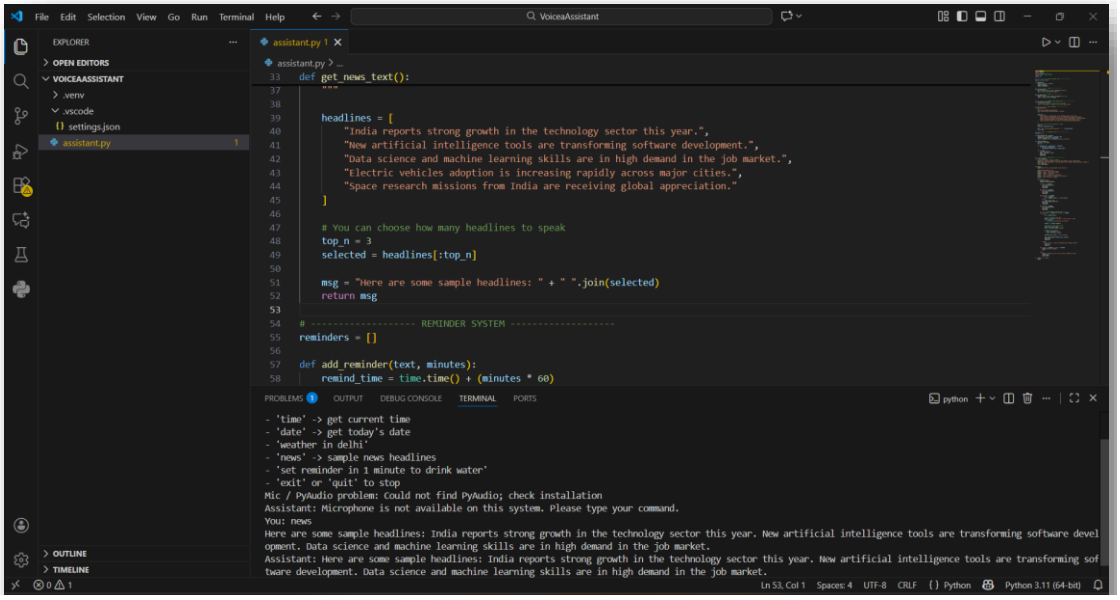
---

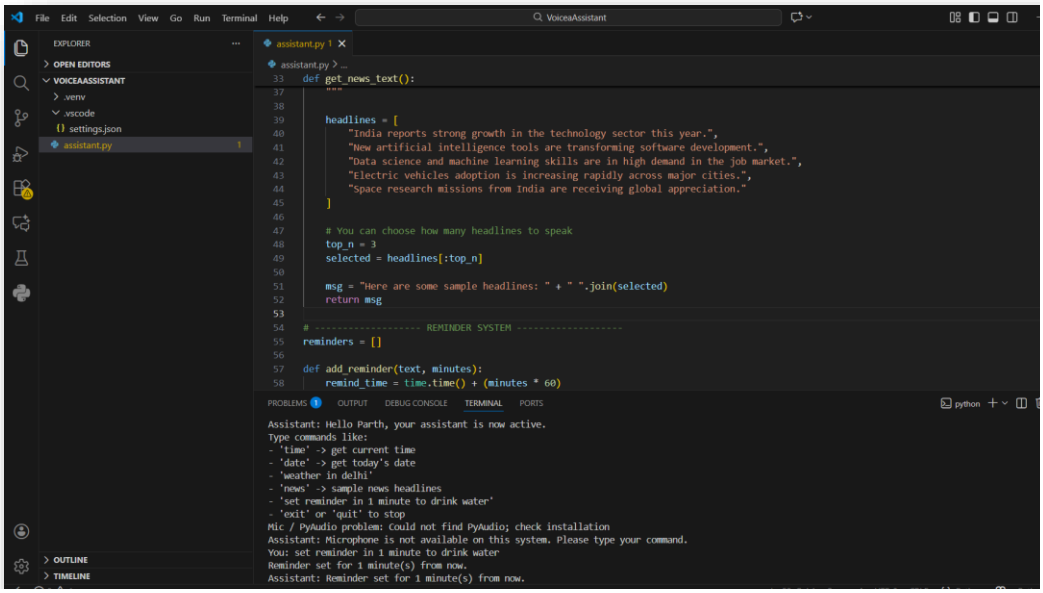
6. SAMPLE OUTPUTS

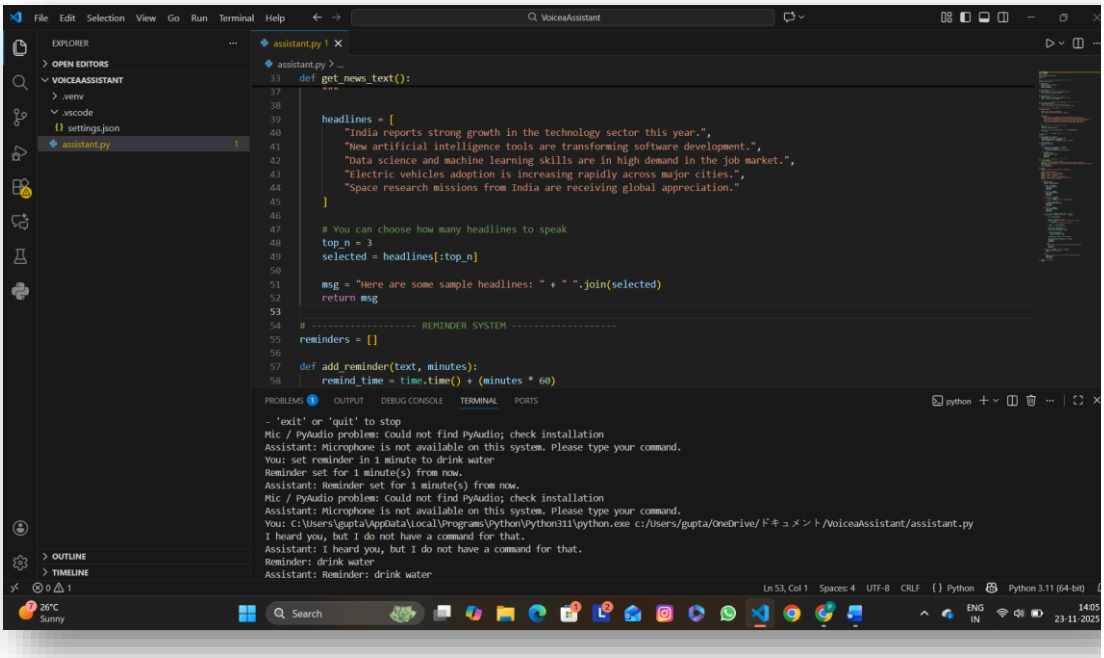







File Edit Selection View Go Run Terminal Help
VoiceaAssistant
EXPLORER
OPEN EDITORS
VOICEAASSISTANT
.venv
.vscode
settings.json
assistant.py
assistant.py 1
assistant.py > ...
def get_news_text():
"""
headlines = [
"India reports strong growth in the technology sector this year.",
"New artificial intelligence tools are transforming software development.",
"Data science and machine learning skills are in high demand in the job market.",
"Electric vehicles adoption is increasing rapidly across major cities.",
"Space research missions from India are receiving global appreciation."
]
# You can choose how many headlines to speak
top_n = 3
selected = headlines[:top_n]
msg = "Here are some sample headlines: " + " ".join(selected)
return msg
# ------------------ REMINDER SYSTEM ------------------
reminders = []
def add_reminder(text, minutes):
remind_time = time.time() + (minutes * 60)
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
python
- 'time' -> get current time
- 'date' -> get today's date
- 'weather in delhi'
- 'news' -> sample news headlines
- 'set reminder in 1 minute to drink water'
- 'exit' or 'quit' to stop
Mic / PyAudio problem: Could not find PyAudio; check installation
Assistant: Microphone is not available on this system. Please type your command.
You: news
Here are some sample headlines: India reports strong growth in the technology sector this year. New artificial intelligence tools are transforming software devel
opment. Data science and machine learning skills are in high demand in the job market.
Assistant: Here are some sample headlines: India reports strong growth in the technology sector this year. New artificial intelligence tools are transforming sof
tware development. Data science and machine learning skills are in high demand in the job market.
OUTLINE
TIMELINE
Ln 53, Col 1 Spaces: 4 UTF-8 CRLF { } Python Python 3.11 (64-bit)


File Edit Selection View Go Run Terminal Help
VoiceaAssistant
EXPLORER
OPEN EDITORS
VOICEAASSISTANT
.venv
.vscode
settings.json
assistant.py
assistant.py 1
assistant.py > ...
def get_news_text():
"""
headlines = [
"India reports strong growth in the technology sector this year.",
"New artificial intelligence tools are transforming software development.",
"Data science and machine learning skills are in high demand in the job market.",
"Electric vehicles adoption is increasing rapidly across major cities.",
"Space research missions from India are receiving global appreciation."
]
# You can choose how many headlines to speak
top_n = 3
selected = headlines[:top_n]
msg = "Here are some sample headlines: " + " ".join(selected)
return msg
# ------------------ REMINDER SYSTEM ------------------
reminders = []
def add_reminder(text, minutes):
remind_time = time.time() + (minutes * 60)
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
python
Assistant: Hello Parth, your assistant is now active.
Type commands like:
- 'time' -> get current time
- 'date' -> get today's date
- 'weather in delhi'
- 'news' -> sample news headlines
- 'set reminder in 1 minute to drink water'
- 'exit' or 'quit' to stop
Mic / PyAudio problem: Could not find PyAudio; check installation
Assistant: Microphone is not available on this system. Please type your command.
You: set reminder in 1 minute to drink water
Reminder set for 1 minute(s) from now.
Assistant: Reminder set for 1 minute(s) from now.
OUTLINE
TIMELINE


File Edit Selection View Go Run Terminal Help
VoiceaAssistant
EXPLORER
OPEN EDITORS
VOICEAASSISTANT
.venv
.vscode
settings.json
assistant.py
assistant.py 1
assistant.py > ...
def get_news_text():
"""
headlines = [
"India reports strong growth in the technology sector this year.",
"New artificial intelligence tools are transforming software development.",
"Data science and machine learning skills are in high demand in the job market.",
"Electric vehicles adoption is increasing rapidly across major cities.",
"Space research missions from India are receiving global appreciation."
]
# You can choose how many headlines to speak
top_n = 3
selected = headlines[:top_n]
msg = "Here are some sample headlines: " + " ".join(selected)
return msg
# ------------------ REMINDER SYSTEM ------------------
reminders = []
def add_reminder(text, minutes):
remind_time = time.time() + (minutes * 60)
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS
python
- 'exit' or 'quit' to stop
Mic / PyAudio problem: Could not find PyAudio; check installation
Assistant: Microphone is not available on this system. Please type your command.
You: set reminder in 1 minute to drink water
Reminder set for 1 minute(s) from now.
Assistant: Reminder set for 1 minute(s) from now.
Mic / PyAudio problem: Could not find PyAudio; check installation
Assistant: Microphone is not available on this system. Please type your command.
You: C:\Users\gupta\AppData\Local\Programs\Python\Python311\python.exe c:/Users/gupta/OneDrive/ドキュメント/VoiceaAssistant/assistant.py
I heard you, but I do not have a command for that.
Assistant: I heard you, but I do not have a command for that.
Reminder: drink water
Assistant: Reminder: drink water
OUTLINE
TIMELINE
Ln 53, Col 1 Spaces: 4 UTF-8 CRLF { } Python Python 3.11 (64-bit)
26°C
Sunny
Search
ENG
IN
14:05
23-11-2025

7. CONCLUSION

This project demonstrates a fully functional **Python-based Voice Assistant** capable of performing a wide range of tasks including:

- speech synthesis
- reminders
- news reading
- time/date announcements
- fallback text mode

The project showcases skills in Python automation, text-to-speech, web data extraction, and user interaction.
In future, the assistant can be enhanced using:

- Live weather API
- Full voice recognition support
- Music playback
- Application automation
- GUI interface

8. APPENDIX – FULL SOURCE CODE

```
import pyttsx3

import datetime

import requests

from bs4 import BeautifulSoup

import time

import re


# ------------------- TEXT TO SPEECH (TTS) --------------------

engine = pyttsx3.init()
```

```
def speak(text):
    print(f"Assistant: {text}")
    engine.say(text)
    engine.runAndWait()

# ------------------- TIME -------------------
def get_time_text():
    now = datetime.datetime.now().strftime("%I:%M %p")
    return f"The current time is {now}."

# ------------------- DATE -------------------
def get_date_text():
    today = datetime.date.today().strftime("%B %d, %Y")
    return f"Today's date is {today}."

# ------------------- WEATHER (FAKE SIMPLE DATA) -------------------
def get_weather_text(city):
    # Simple fake response (no external API required)
    return f"The weather in {city} is warm with light clouds."

# ------------------- NEWS (SCRAPED) -------------------
def get_news_text():
    """
    Returns some sample news headlines.
    This version does NOT need internet or any API.
    """
```

```
    headlines = [
        "India reports strong growth in the technology sector this year.",
        "New artificial intelligence tools are transforming software development.",
        "Data science and machine learning skills are in high demand in the job market.",
        "Electric vehicles adoption is increasing rapidly across major cities.",
        "Space research missions from India are receiving global appreciation."
    ]

    # You can choose how many headlines to speak
    top_n = 3
    selected = headlines[:top_n]

    msg = "Here are some sample headlines: " + " ".join(selected)
    return msg

# ------------------- REMINDER SYSTEM -------------------
reminders = []

def add_reminder(text, minutes):
    remind_time = time.time() + (minutes * 60)
    reminders.append((text, remind_time))
    return f"Reminder set for {minutes} minute(s) from now."

def check_reminders():
    current = time.time()
    due = []

    for reminder_text, remind_time in reminders:
```

```
        if current >= remind_time:
            due.append((reminder_text, remind_time))

    for r in due:
        reminders.remove(r)
        msg = f"Reminder: {r[0]}"
        print(msg)
        speak(msg)

# -------------------- MAIN LOGIC --------------------
def listen_command():
    # Mic unavailable → type your command
    print("Mic / PyAudio problem: Could not find PyAudio; check installation")
    print("Assistant: Microphone is not available on this system. Please type your command.")
    cmd = input("You: ")
    return cmd.lower()

def main():
    speak("Hello Parth, your assistant is now active.")

    print("Type commands like:")
    print("- 'time' -> get current time")
    print("- 'date' -> get today's date")
    print("- 'weather in delhi'")
    print("- 'news' -> sample news headlines")
    print("- 'set reminder in 1 minute to drink water'")
    print("- 'exit' or 'quit' to stop")
```

```
while True:
    check_reminders()
    command = listen_command()
    if "time" in command:
        msg = get_time_text()
        print(msg)
        speak(msg)
    elif "date" in command:
        msg = get_date_text()
        print(msg)
        speak(msg)
    elif "weather" in command:
        if "in" in command:
            city = command.split("in", 1)[1].strip()
        else:
            city = "your city"
        msg = get_weather_text(city)
        print(msg)
        speak(msg)
    elif "news" in command:
        msg = get_news_text()
        print(msg)
        speak(msg)
    elif "reminder" in command and "minute" in command:
        try:
            txt = command.lower()
```

```
        # extract minutes using regex
        match = re.search(r"reminder in (\d+)\s*minute", txt)
        if not match:
            raise ValueError("Minutes not found")

        minutes = int(match.group(1))

        # extract the text after "minute"
        after_index = match.end()
        after = txt[after_index:].strip()

        # remove "to" if present
        if after.startswith("to "):
            after = after[3:].strip()

        reminder_text = after or "your reminder"

        msg = add_reminder(reminder_text, minutes)
        print(msg)
        speak(msg)

    except:
        err = "Sorry, I could not understand the reminder format."
        print(err)
        speak(err)

elif "exit" in command or "quit" in command:
    speak("Assistant stopped.")
```

```
            break


        else:
            reply = "I heard you, but I do not have a command for that."
            print(reply)
            speak(reply)


if __name__ == "__main__":
    main()
```
