Technical Documentation for "Voice-Activated Personal Assistant with Arduino Integration(RISE BUJJI)”

TEAM Dept of CSE-DS

RISE KRISHNA SAI GANDHI GROUP OF INSTITUTIONS , Valluru.

Technical Documentation for “Voice-Activated Personal Assistant with Arduino Integration” in Python

---

Project Overview:

The Voice-Activated Personal Assistant is an application that integrates voice commands, Arduino control, PDF reading, MP3 playback, and weather reporting. It leverages various Python libraries to listen to voice commands, interpret them, and perform tasks such as controlling hardware (via Arduino), fetching web results, reading PDFs aloud, and playing audio files.

Features:

1. **Voice Activation**: The assistant listens for a wake-up word (e.g., "Siri") and performs actions based on voice commands.
2. **Arduino Integration**: Allows control of Arduino for specific tasks (e.g., turning on/off LEDs, shaking hands, walking).
3. **PDF Reader**: Reads the text from PDF files and converts it to speech.
4. **MP3 Playback**: Plays specified MP3 files using the pygame library.
5. **Web Search**: Fetches quick web results using the DuckDuckGo API.
6. **Weather Reporting**: Provides weather updates for a specified city by querying the Open Weather API.
7. **Basic Math Operations**: Processes simple math expressions from voice commands and calculates results.
8. **Real-time Time Reporting**: Reports the current time based on user request.

List of Functions:

1. **speak(text)**: Converts a given text to speech using the pyttsx3 engine.
2. **play_mp3(file_path)**: Plays an MP3 file located at the given file path.
3. **turn_led_on()**: Sends a command to Arduino to turn an LED on.
4. **turn_led_off()**: Sends a command to Arduino to turn the LED off.
5. **take_command(idle_mode=True)**: Listens for a voice command, optionally waiting for a wake-up word, and returns the recognized text.
6. **listen_for_wakeup_word()**: Continuously listens for the wake-up word (e.g., "Siri") and initiates the main function when triggered.
7. **read_pdf(filepath)**: Reads text from a PDF file and converts it to speech.

8. **truncate_text(text, word_limit=30)**: Truncates a given text to a specific word limit.
9. **search_duckduckgo(query)**: Searches DuckDuckGo for a given query and returns an abstract of the results.
10. **get_weather(city)**: Fetches the weather information for a given city using the OpenWeather API.
11. **listen_for_city()**: Listens for a city name from the user's speech input.
12. **main()**: The main loop that processes user commands and triggers appropriate functions.
13. **Arduino control functions**: shake_hand, walk, turn_left, turn_right, move_back send specific commands to the Arduino for motor control actions.

Python Packages Used:

1. **serial**: Used for communication between the Python program and the Arduino.
    - Allows the program to send and receive commands to control hardware like LEDs or motors.
2. **pyttsx3**: Text-to-speech conversion library.
    - Used to generate spoken responses from text (e.g., reading PDF, reporting time, weather, etc.).
3. **pygame**: Used for playing MP3 files.
    - This library initializes the mixer and plays audio files while handling errors if the file is missing or corrupt.
4. **os**: For file path operations.
    - Used to check if an MP3 file exists before attempting playback.
5. **PyPDF2**: Library to read and extract text from PDF files.
    - Processes PDF pages and extracts text to be read aloud by the assistant.
6. **requests**: Used to fetch data from web APIs like OpenWeather and DuckDuckGo.
    - Queries external APIs and processes JSON responses to retrieve weather and search information.
7. **speech_recognition (sr)**: Recognizes voice input.
    - Used for converting speech to text, enabling voice commands to control the assistant.
8. **datetime (dt)**: Fetches the current date and time.
    - Used for reporting the current time on request.

Detailed Explanation of Python Code:

1. Arduino Integration:

The serial package is used to communicate with Arduino. Commands such as arduino.write(b,'0') turn the LED on or off, or trigger other physical actions (e.g., walking or shaking hands) using different command codes.

2. Text-to-Speech (speak):

The speak function is central to converting text responses into speech. It uses pyttsx3, where properties like voice rate and type of voice (male/female) can be customized.

python

Copy code

```
def speak(text):
    speaker = pyttsx3.init()
    speaker.setProperty('rate', 150)  # Adjusts speech rate
    speaker.say(text)
    speaker.runAndWait()
```

3. MP3 Playback (play_mp3):

The pygame library is initialized for playing audio files. It checks if the file exists before playing and handles exceptions during playback.

python

Copy code

```
def play_mp3(file_path):
    pygame.mixer.init()  # Initializes the mixer module for sound
    pygame.mixer.music.load(file_path)
    pygame.mixer.music.play()
```

4. Voice Command Recognition (take_command):

This function listens to the user's voice and processes it using the speech_recognition library.

python

Copy code

```
def take_command(idle_mode=True):
    with sr.Microphone() as source:
        voice = listener.listen(source)
        command = listener.recognize_google(voice)  # Converts speech to text
```

5. PDF Reader (read_pdf):

The PyPDF2 library extracts text from PDF files, which is then passed to the speak function to read aloud.

python

Copy code

```
def read_pdf(filepath):
    pdf_reader = PyPDF2.PdfReader(pdffile)
    for page in pdf_reader.pages:
        text = page.extract_text()
        speak(text)  # Reads each page's text aloud
```

6. Weather Reporting:

The get_weather function sends a request to the OpenWeather API using the requests library. It processes the response and retrieves weather information for the specified city.

python

Copy code

```
def get_weather(city):
    params = {'q': city, 'appid': API_KEY, 'units': 'metric'}
    response = requests.get(WEATHER_URL, params=params)
    data = response.json()
    return f"The weather in {data['name']} is {data['weather'][0]['description']}"
```

7. Main Function:

The main loop continuously listens for commands and directs them to the corresponding function (e.g., weather report, PDF reading, playing music, etc.).

python

Copy code

```
def main():
    while True:
        user_command = take_command(idle_mode=False)
        if 'weather in' in user_command:
            city = user_command.replace('weather in', '')
            weather_report = get_weather(city)
            speak(weather_report)
```

Conclusion:

This documentation outlines the core functionalities, libraries, and flow of the "Voice-Activated Personal Assistant with Arduino and PDF Reader." Each component is modular, handling specific tasks such as voice recognition, file handling, hardware interaction, and web-based functionalities.