AI-Driven Personal Assistant

Overview

The AI-Driven Personal Assistant is a Python-based application that integrates Twilio APIs and an AI assistant to handle incoming calls, summarize messages, and send them to recipients. This application is ideal for situations where the recipient is unavailable, allowing the caller's message to be summarized by an AI model and relayed via WhatsApp.

Table of Contents

Code Workflow

1. **Handle Incoming Call**
    - Checks the call status.
    - Redirects the caller to the AI assistant if the recipient is unavailable.
2. **Interact with AI**
    - Captures and summarizes the caller's message using a local AI model via the Ollama API.
3. **Send Message Summary**
    - Sends the summarized message to the recipient via WhatsApp using Twilio's API.

Environment Setup

Prerequisites

- Python 3.9 or later
- Twilio account (with WhatsApp sandbox setup)
- Ollama AI API running locally
- Required Python packages:

```
pip install requests twilio
```

Environment Variables

Set the following environment variables for Twilio:

- `TWILIO_ACCOUNT_SID`: Your Twilio Account SID.
- `TWILIO_AUTH_TOKEN`: Your Twilio Auth Token.
- `TWILIO_WHATSAPP_NUMBER`: Twilio sandbox number for WhatsApp.

Code Documentation

`handle_incoming_call`

Description

Handles incoming calls and directs them to the AI assistant if the recipient is unavailable.

Parameters

- `call_status` (str): Status of the recipient's phone (e.g., "busy", "unavailable").
- `caller_number` (str): Phone number of the caller.
- `recipient_number` (str): Phone number of the recipient.

Workflow

1. If the call status is in ["busy", "unavailable", "no-answer"], initiate AI interaction.
2. Summarize the caller's message.
3. Send the summary to the recipient.

---

interact_with_ai

Description

Uses a conversational AI to capture and summarize the caller's message.

Parameters

- `caller_number` (str): Phone number of the caller.

Workflow

1. Send a POST request to Ollama's local API with the caller's message.
2. Extract the summarized response.
3. Handle connection errors gracefully.

Example Payload

```
{
  "model": "llama3.2:latest",
  "prompt": "Hello, I need assistance with billing.",
  "stream": false
}
```

Returns

- (str): Summarized message or an error message if the AI interaction fails.

---

send_message_summary

Description

Sends a summarized message to the recipient via WhatsApp or Telegram.

Parameters

- `recipient_number` (str): Phone number of the recipient.
- `summary` (str): Summarized message to send.

Workflow

1. Retrieve Twilio credentials.
2. Use Twilio's API to send a WhatsApp message with the summary.
3. Handle errors during the message-sending process.

Test Suite

Unit Tests

```
import unittest
from unittest.mock import patch, MagicMock
import requests

class TestPersonalAssistant(unittest.TestCase):

    @patch('requests.post')
    def test_interact_with_ai_success(self, mock_post):
        mock_response = MagicMock()
        mock_response.json.return_value = {"response": "Billing assistance requested."}
        mock_response.status_code = 200
        mock_post.return_value = mock_response

        from main import interact_with_ai
        summary = interact_with_ai("+1234567890")
        self.assertEqual(summary, "Billing assistance requested.")

    @patch('requests.post')
    def test_interact_with_ai_failure(self, mock_post):
        mock_post.side_effect = requests.exceptions.RequestException("API Error")

        from main import interact_with_ai
        summary = interact_with_ai("+1234567890")
        self.assertEqual(summary, "Error generating summary. Please check the AI assistant.")

    @patch('twilio.rest.Client.messages.create')
    def test_send_message_summary_success(self, mock_create):
        mock_create.return_value = None

        from main import send_message_summary
        send_message_summary("+1234567890", "Test Summary")
        mock_create.assert_called_once()
```

Integration Tests (Robot Framework)

Test Case File: `assistant_test.robot`

```
*** Settings ***
Library    RequestsLibrary
Library    OperatingSystem
Library    Process

*** Variables ***
${AI_API_URL}    http://localhost:11434/api/generate

*** Test Cases ***
Verify AI Interaction
    [Documentation]  Verify that the AI interaction endpoint processes and returns a summary.
    Create Session    Ollama    ${AI_API_URL}
    ${payload}=    Create Dictionary    model=llama3.2:latest    prompt=Hello, I need assistance.
    ${response}=    Post Request    Ollama    /generate    json=${payload}
    Should Be Equal As Strings    ${response.json()['response']}    Assistance needed.

Verify Twilio Message Sending
    [Documentation]  Verify that Twilio API sends a WhatsApp message successfully.
    Run Process    python    -c    "from twilio.rest import Client; client = Client('sid', 'token'); client.messages.create(from_='wh
```

Future Enhancements

- Add support for multiple AI models.
- Integrate Telegram messaging as an alternative.
- Enhance error logging and monitoring.
- Expand the test suite to include performance tests.