МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ

ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ

ТЕХНОЛОГІЙ

Проект №6

З дисципліни

«Штучний інтелект»

Виконав:

Студент групи ПД-44

Солов'ян Арсен

Київ – 2025

Опис проекту:
Розробити голосового бота, який дозволяє взаємодіяти з користувачами через голосові команди для виконання таких завдань:

- Надання інформації (наприклад, погода, час, новини, довільна предметна область, тощо).
- Підтримка користувачів (пошук рішень для питань або проблем).
- Автоматизація процесів (оформлення замовлень, нагадування, виконання транзакцій).

Основні вимоги

1. Розпізнавання голосу (ASR): Реалізовано. Використовується бібліотека speech_recognition для захоплення аудіо з мікрофона та звертається до API OpenAI (модель Whisper) для перетворення голосу в текст (client.audio.transcriptions.create). Модель Whisper забезпечує високу точність розпізнавання української мови (language="uk").
2. Обробка природної мови (NLP): Реалізовано. Текст, отриманий після розпізнавання, надсилається до моделі GPT-4o через API OpenAI (client.chat.completions.create). GPT-4o аналізує зміст запиту користувача, визначає його намір та генерує змістовну відповідь, враховуючи попередню історію діалогу.
3. Синтез голосу (TTS): Реалізовано. Текстова відповідь, згенерована GPT-4o, передається до сервісу OpenAI TTS (client.audio.speech.create) для перетворення на аудіо. Для відтворення отриманого аудіофайлу використовується бібліотека pygame.mixer.
4. Логічна обробка запитів: Частково реалізовано. Основна логіка розуміння запиту та формування відповіді делегована моделі GPT-4o. У Python коді реалізовано базову логіку циклу взаємодії (очікування Enter -> запис -> розпізнавання -> обробка -> відповідь) та перевірку наявності команд для завершення роботи ("прощавай", "бувай", "до побачення").

C: > Users > Kroll > Desktop > voice.py > ...

```python
import speech_recognition as sr
import openai
import io
import pygame
import time


OPENAI_API_KEY = "kroll"

client = openai.OpenAI(api_key=OPENAI_API_KEY)

recognizer = sr.Recognizer()
microphone = sr.Microphone()

conversation_history = [
    {"role": "system", "content": "Ти - корисний голосовий асистент, що відповідає українською мовою."}
]


def listen_for_audio():
    """Захоплює аудіо з мікрофона і повертає аудіо дані."""
    with microphone as source:
        print("Налаштування на фоновий шум...")
        recognizer.adjust_for_ambient_noise(source, duration=1)
        print("ЗАПИС: Будь ласка, говоріть...")
        try:
            audio_data = recognizer.listen(source, timeout=5, phrase_time_limit=10)
            print("Аудіо отримано, розпізнавання...")
            return audio_data
        except sr.WaitTimeoutError:
            print("Час очікування вичерпано, мовлення не виявлено.")
            return None
        except Exception as e:
            print(f"Помилка під час прослуховування: {e}")
            return None

def transcribe_audio(audio_data):
    """Надсилає аудіо дані до OpenAI Whisper API для розпізнавання."""
    if not audio_data:
        return None
    try:
        wav_bytes = audio_data.get_wav_data()
        audio_file = io.BytesIO(wav_bytes)
        audio_file.name = "audio.wav"

        transcript = client.audio.transcriptions.create(
            model="whisper-1",
            file=audio_file,
            language="uk"
        )
        print(f"Розпізнано: {transcript.text}")
        return transcript.text
    except openai.APIError as e:
        print(f"Помилка OpenAI API (Whisper): {e}")
        return None
    except Exception as e:
        print(f"Не вдалося розпізнати мову: {e}")
        return None

def get_gpt_response(text):
    """Надсилає текст до GPT-4o і повертає відповідь українською."""
    if not text:
        return "Вибачте, я не розчув ваш запит."

    conversation_history.append({"role": "user", "content": text})

    try:
        response = client.chat.completions.create(
            model="gpt-4o",
            messages=conversation_history,
            max_tokens=150
        )
        assistant_response = response.choices[0].message.content
        print(f"GPT Відповідь: {assistant_response}")

        conversation_history.append({"role": "assistant", "content": assistant_response})
        return assistant_response

    except openai.APIError as e:
        print(f"Помилка OpenAI API (GPT): {e}")
        conversation_history.pop()
        return "Вибачте, сталася помилка при обробці вашого запиту."
    except Exception as e:
        print(f"Інша помилка при взаємодії з GPT: {e}")
        conversation_history.pop()
        return "Вибачте, непередбачувана помилка."
```

Функціональні можливості

1. Основні функції:
Аналіз голосових команд для визначення завдання виконується моделлю GPT-4o під час обробки запиту.
2. Відповіді на питання користувачів: текстова відповідь виводиться в консоль (GPT Відповідь:), а голосова відповідь відтворюється звуком.

Архітектура системи:

1. Клієнтська частина:
Мікрофон для запису голосу (використовується speech_recognition та sr.Microphone).
2. Модуль візуалізації (базовий): Текстові відповіді та службові повідомлення виводяться у консоль.
3. Серверна частина (використовувані сервіси API):
Модуль ASR: OpenAI Whisper (через API).
Модуль NLP: OpenAI GPT-4o (через API).
Модуль логіки (частково): Обробка сценаріїв виконується GPT-4o, базова логіка циклу - у Python.
Модуль TTS: OpenAI TTS (через API).
Відтворення - локально через pygame.mixer.

Фреймворки для обробки мовлення: speech_recognition (для запису), pygame (для відтворення).
Хмарні сервіси: OpenAI API (для Whisper, GPT-4o, TTS).

У результаті бот:
1. Активується натисканням Enter.
2. Записує голос користувача.
3. Розпізнає українську мову за допомогою OpenAI Whisper.
4. Обробляє запит та генерує відповідь українською мовою за допомогою OpenAI GPT-4o, підтримуючи контекст розмови.
5. Синтезує голосову відповідь за допомогою OpenAI TTS.
6. Відтворює відповідь користувачеві за допомогою pygame.

```python

def speak_text(text):
    """Перетворює текст на мову за допомогою OpenAI TTS і відтворює його через pygame.mixer."""
    if not text:
        print("Немає тексту для озвучення.")
        return

    try:
        response = client.audio.speech.create(
            model="tts-1",
            voice="nova",
            input=text
        )
        audio_bytes = response.content
        print("Обробка аудіо відповіді...")
        try:
            pygame.mixer.music.load(io.BytesIO(audio_bytes))
            print("Відтворення відповіді...")
            pygame.mixer.music.play()
            while pygame.mixer.music.get_busy():
                pygame.time.Clock().tick(10)
        except pygame.error as e:
            print(f"Помилка Pygame під час завантаження/відтворення: {e}")

    except openai.APIError as e:
        print(f"Помилка OpenAI API (TTS): {e}")
    except Exception as e:
        print(f"Помилка під час синтезу або відтворення мови: {e}")


if __name__ == "__main__":
    print("Голосовий бот запущено. Скажіть 'прощавай' або 'бувай', щоб завершити.")

    try:
        pygame.init()
        pygame.mixer.init()
        mixer_initialized = True
    except Exception as e:
        print(f"Не вдалося ініціалізувати Pygame: {e}")
        print("Відтворення звуку може не працювати.")
        mixer_initialized = False

    try:
        while True:
            print("\n=== Готовий до запису ===")
            print("Натисніть Enter, щоб почати запис вашого голосового запиту...")
            input()

            audio = listen_for_audio()

            if audio:
                user_text = transcribe_audio(audio)
                if user_text:
                    if any(word in user_text.lower() for word in ["прощавай", "бувай", "до побачення"]):
                        if mixer_initialized:
                            speak_text("До побачення!")
                        else:
                            print("До побачення!")
                        break

                    bot_response = get_gpt_response(user_text)
                    if mixer_initialized:
                        speak_text(bot_response)
                    else:
                        print(f"Відповідь (без звуку): {bot_response}")

                else:
                    print("Не вдалося розпізнати мову в записі.")

    except KeyboardInterrupt:
        print("\nЗавершення роботи бота (Ctrl+C).")
    finally:
        if pygame.get_init():
            pygame.quit()
        print("Роботу бота завершено.")
```