```
/// <summary>
    /// Starts recording from the first available microphone.
    /// </summary>
    public void StartRecording()
    {
        if (Microphone.devices.Length == 0)
        {
            Debug.LogWarning("No microphone detected.");
            return;
        }

        microphoneDevice = Microphone.devices[0];
        recordingClip = Microphone.Start(microphoneDevice, false, recordingLength, sampleRate);
        IsRecording = true;
        Debug.Log("🎤 Recording started...");
    }

    /// <summary>
    /// Stops recording and begins sending the audio to Whisper for transcription.
    /// </summary>
    public void StopRecording()
    {
        if (!IsRecording) return;

        Microphone.End(microphoneDevice);
        IsRecording = false;
        Debug.Log("🔴 Recording stopped. Sending to AI...");

        string tempFilePath = Path.Combine(Application.temporaryCachePath, "recorded_audio.wav");
        SaveWavFile(recordingClip, tempFilePath);

        StartCoroutine(SendToWhisper(tempFilePath));
    }

    /// <summary>
    /// Sends the recorded WAV file to OpenAI Whisper for transcription.
    /// </summary>
    private IEnumerator SendToWhisper(string filePath)
    {
        byte[] audioData = File.ReadAllBytes(filePath);

        WWWForm form = new WWWForm();
        form.AddBinaryData("file", audioData, "audio.wav", "audio/wav");
        form.AddField("model", "whisper-1");
        form.AddField("language", "ro"); // force Romanian

        using (UnityWebRequest request =
UnityWebRequest.Post("https://api.openai.com/v1/audio/transcriptions", form))
        {
            request.SetRequestHeader("Authorization", "Bearer " + openAiApiKey);

            yield return request.SendWebRequest();

            if (request.result == UnityWebRequest.Result.Success)
            {
                ParseWhisperResponse(request.downloadHandler.text);
                ForwardTextToAI();
            }
            else
            {
                Debug.LogError("❌ Whisper API error: " + request.error + "\n" +
request.downloadHandler.text);
            }
        }
    }

    /// <summary>
    /// Extracts the text result from Whisper's JSON response.
    /// </summary>
    private void ParseWhisperResponse(string jsonResponse)
    {
        try
        {
            var json = SimpleJSON.JSON.Parse(jsonResponse);
            ConvertedText = json["text"].Value.Trim();
            Debug.Log("✅ Transcribed text: " + ConvertedText);
        }
        catch
        {
            Debug.LogError("❌ Failed to parse Whisper response.");
            ConvertedText = "";
        }
    }

    /// <summary>
    /// Sends the converted text to the AI NPC as a question.
    /// </summary>
    private void ForwardTextToAI()
    {
        if (string.IsNullOrWhiteSpace(ConvertedText)) return;

        npcController.SetInputText(ConvertedText);
        aiConnection.SetQuestion(ConvertedText);
        aiConnection.AskAI();
    }

    /// <summary>
    /// Saves the recorded AudioClip to a WAV file.
    /// </summary>
    private void SaveWavFile(AudioClip clip, string filePath)
    {
        using (FileStream fileStream = new FileStream(filePath, FileMode.Create))
        {
            int samples = clip.samples * clip.channels;
            float[] data = new float[samples];
            clip.GetData(data, 0);

            byte[] wavBytes = ConvertAudioClipToWav(data, clip.frequency, clip.channels);
            fileStream.Write(wavBytes, 0, wavBytes.Length);
        }
    }

    /// <summary>
    /// Converts float audio samples to WAV format.
    /// </summary>
    private byte[] ConvertAudioClipToWav(float[] samples, int sampleRate, int channels)
    {
        // Normalize audio
        float max = 0f;
        foreach (float sample in samples) max = Mathf.Max(max, Mathf.Abs(sample));
        if (max > 0f)
            for (int i = 0; i < samples.Length; i++) samples[i] /= max;

        using (MemoryStream stream = new MemoryStream())
        using (BinaryWriter writer = new BinaryWriter(stream))
        {
            int sampleCount = samples.Length;
            int byteCount = sampleCount * 2;

            // WAV header
            writer.Write(System.Text.Encoding.UTF8.GetBytes("RIFF"));
            writer.Write(36 + byteCount);
            writer.Write(System.Text.Encoding.UTF8.GetBytes("WAVE"));
            writer.Write(System.Text.Encoding.UTF8.GetBytes("fmt "));
            writer.Write(16);
            writer.Write((short)1); // PCM
            writer.Write((short)channels);
            writer.Write(sampleRate);
            writer.Write(sampleRate * channels * 2);
            writer.Write((short)(channels * 2));
            writer.Write((short)16);
            writer.Write(System.Text.Encoding.UTF8.GetBytes("data"));
            writer.Write(byteCount);

            // Audio data
            foreach (float sample in samples)
            {
                short s = (short)(Mathf.Clamp(sample, -1f, 1f) * short.MaxValue);
                writer.Write(s);
            }

            return stream.ToArray();
        }
    }
```