AI Based Number Guessing Game

A PROJECT REPORT
for
Introduction to AI (AI101B)
Session (2024-25)

Submitted by

Ashwani Kumar Katiyar (202410116100044)
Anup Kumar Mahto (202410116100038)
Anubhav Vaish (202410116100037)
Abhi Srivastava (202410116100006)

Submitted in partial fulfilment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
Prof. Apoorv Jain
Assistant Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206

(APRIL- 2025)

CERTIFICATE

Certified that Ashwani Kumar Katiyar (202410116100044), Anup Kumar Mahto (202410116100038), Anubhav Vaish 202410116100037), Abhi Srivastava (202410116100006) has/ have carried out the project work having "**Ai Based Number Guessing Game in Introduction to AI (AI101B)** for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Mr. Apoorv Jain
Assistant Professor

Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Dr. Akash Rajak

Dean
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

AI Based Number Guessing Game

Ashwani Kumar Katiyar , Anup Kumar Mahto , Anubhav Vaish ,Abhi Srivastava

ABSTRACT

The AI-Powered Number Guessing Game is an innovative project that merges traditional gaming elements with modern Artificial Intelligence (AI) technologies to create an interactive, voice-driven user experience. This game challenges players to guess a randomly generated number between 1 and 100. What sets this game apart is its integration with Google Cloud APIs, specifically Speech-to-Text for voice recognition and Natural Language Processing (NLP) for providing AI-powered hints.

The game allows players to interact either by typing their guesses or speaking them aloud, with the system transcribing the spoken input into text. The AI then analyzes the player's guess, providing hints based on the proximity of the guess to the correct number. These hints are enhanced through sentiment analysis, which adjusts feedback based on the player’s progress. The project uses Python, Tkinter for the user interface, and Google Cloud services to power the speech recognition and intelligent hint generation. This not only makes the game more engaging but also demonstrates how AI can be used in interactive gaming applications.

This project highlights the potential of AI in creating dynamic user experiences, showcasing the integration of speech recognition, natural language processing, and interactive game design. It serves as a practical example of how modern technologies can make games more immersive, accessible, and educational. Through this project, users can experience firsthand how AI and machine learning can be applied to real-world applications in an engaging and accessible manner.

ACKNOWLEDGEMENTS

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Mr Apoorv Jain** for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Akash Rajak, Professor and Dean, Department of Computer Applications**, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Ashwani Kumar Katiyar

Anup Kumar Mahto

Anubhav Vaish

Abhi Srivastava

TABLE OF CONTENTS

Chapter 1
INTRODUCTION

Overview

The **AI-Powered Number Guessing Game** combines traditional game mechanics with advanced Artificial Intelligence (AI) to create an interactive and engaging user experience. The game challenges players to guess a randomly generated number between 1 and 100, with the added complexity of receiving **AI-generated hints** to guide them toward the correct answer. The game leverages **speech recognition** to allow voice-based interactions, enhancing accessibility and making the experience more immersive.

This project involves utilizing **Google Cloud APIs** for both **speech-to-text recognition** and **natural language processing**. These technologies help to transcribe spoken inputs into numbers and provide intelligent feedback based on the player's guesses. The use of AI in generating hints and feedback makes the game more dynamic, as it adapts to the player's actions, offering unique responses based on each guess.

Key Features

1. **Voice Interaction**: Players can speak their guesses instead of typing them, utilizing **Google Speech-to-Text API** for transcription.
2. **AI-Powered Hints**: The game provides intelligent hints using the **Google Natural Language API** to guide players towards the correct answer.
3. **Dynamic Gameplay**: The game adapts based on the player's progress, generating personalized hints for each guess.
4. **User Interface (UI)**: A clean and intuitive GUI is built using **Tkinter** in Python, allowing easy interaction for both voice and text-based inputs.
5. **Feedback Mechanism**: Real-time feedback is provided after each guess, including whether the guess is too high, too low, or correct. Upon winning, a new number is generated to start a fresh round.

How It Works

1. **Game Initialization**:
   The game starts by generating a random number between 1 and 100. This is the number the player must guess.

2. **Guessing Mechanism**:
   Players can either enter their guess using the keyboard or speak their guess using voice commands. If using the keyboard, players enter a number, and the system checks whether the guess is correct. If using voice, the system records the audio, converts it to text using **Google's Speech-to-Text API**, and processes the result to extract the guessed number.

3. **AI-Generated Hints**:
   After every guess, the AI evaluates the player's input. Based on the comparison between the player's guess and the correct number, the AI provides a hint, guiding the player to guess higher or lower. The hint also uses sentiment analysis to provide positive feedback when the player is close and constructive feedback when far from the correct number.

4. **Winning Condition**:
   If the player guesses the correct number, a congratulatory message is shown, and a new random number is generated to restart the game. The process continues until the player decides to quit.

Technologies Used

- **Python**: The core programming language for the game logic and UI development.
- **Tkinter**: Python's standard GUI library, used to create the interactive user interface.
- **Google Cloud APIs**:
    - **Speech-to-Text API**: Converts spoken guesses into text, enabling voice-based input.
    - **Natural Language API**: Analyzes player guesses and generates intelligent hints.
- **PyAudio**: A library used for recording audio from the microphone, enabling voice interaction.

Benefits of the Project

- **Engagement**: Players enjoy a more immersive experience with voice interaction, allowing them to use natural speech instead of just typing.
- **Educational**: The game offers a fun way for users to interact with AI, helping them learn about speech recognition and natural language processing technologies.
- **Accessibility**: By incorporating voice-based input, the game becomes accessible to users with disabilities or those who prefer not to use a keyboard.

Challenges & Considerations

- **Voice Recognition Accuracy**: The accuracy of speech-to-text conversion is a critical factor, especially with noisy environments or unclear speech.
- **API Latency**: The speed of communication with Google Cloud APIs might affect the real-time feedback during gameplay, depending on network conditions.
- **Error Handling**: Handling incorrect guesses, unrecognized speech, or invalid input is important for a smooth user experience.

Chapter 2

Methodology

The methodology of this project follows a structured approach, where we combine traditional game mechanics with modern AI and machine learning technologies to create an interactive and engaging user experience. The game is developed using Python as the core programming language, and various libraries and APIs are integrated to achieve voice recognition and AI-powered hint generation. Below is a detailed breakdown of the methodology, followed by an explanation of the libraries used in the project.

Project Methodology

1. Problem Identification and Game Concept

The main objective of this project is to create an AI-based number guessing game that allows players to guess a randomly generated number between 1 and 100. The game offers the following features:

- Voice-based Input: The player can speak their guess instead of typing it.
- AI-Powered Hints: The AI provides hints based on the player's guess, guiding them toward the correct number.
- User Interaction: The game provides a simple, intuitive interface where the player can either type or speak their guess.

2. System Design

The system architecture is composed of two main components:

1. Game Logic: This includes the random number generation, checking the player's guesses, and providing hints.
2. User Interface (UI): The interface is built using Tkinter for visual interaction. The UI includes buttons, entry fields, and labels to display game instructions, user inputs, results, and feedback.

3. Speech Recognition Integration

- The system uses Google Cloud's Speech-to-Text API to convert the player’s voice input into text. This allows the user to provide their guess verbally.
- The PyAudio library is used to capture audio from the microphone and store it as a file, which is then transcribed by the Speech-to-Text API.

4. Natural Language Processing for AI Hints

- The game uses Google Cloud's Natural Language API to analyze the player’s guess and generate personalized AI-powered hints.
- Based on the player’s guess and the actual number, the system provides feedback that is more intelligent and helpful than just stating whether the guess is too high or low.

5. Game Flow

- A random number is generated at the start of the game.
- The player either types or speaks their guess.
- The system evaluates the guess and provides feedback, either guiding the player to guess higher or lower or congratulating them if they win.
- The game then continues with a new random number after each win.

Libraries Used

Below are the key libraries used in this project, with explanations of their roles:

1. Tkinter

- Purpose: Tkinter is the standard Python library for creating Graphical User Interfaces (GUIs).
- Role in the Project: Tkinter is used to create the visual interface of the game. It handles the main window, input fields, buttons, and labels where results and hints are displayed.

Key Components:

  - Tk() - Creates the main application window.

- Label() - Used to display instructions, hints, and results.
- Entry() - Provides an input field for users to type their guesses.
- Button() - Allows users to interact with the game by checking guesses or using the voice input option.

2. PyAudio

- Purpose: PyAudio is a Python library used for working with audio streams, particularly for recording and playing sound.
- Role in the Project: PyAudio is used to record audio from the user's microphone, saving it as an audio file (e.g., .wav format). This audio file is then passed to the Google Speech-to-Text API for transcription.

Key Components:

- pyaudio.PyAudio() - Initializes the audio stream for recording.
- stream.read() - Captures raw audio data from the microphone.
- wave.open() - Saves the recorded audio to a .wav file format for processing.

3. Random

- Purpose: The random module is used for generating random numbers.
- Role in the Project: It generates a random target number between 1 and 100 that the player must guess. This random number is changed after every successful guess to ensure the game remains challenging and fun.

Key Components:

- random.randint() - Generates a random integer between a specified range (1-100).

4. OS

- Purpose: The os module in Python provides functions for interacting with the operating system, particularly useful for handling file paths and environment variables.

- Role in the Project: It is used to set the Google Cloud API credentials and ensure that the path to the JSON key file is correctly configured before interacting with Google Cloud services.

Key Components:

  - os.environ["GOOGLE_APPLICATION_CREDENTIALS"] - Sets the environment variable that points to the location of the Google Cloud credentials file.

API:-

1. Google Cloud Speech-to-Text API

- Purpose: The Google Speech-to-Text API transcribes spoken words into written text, enabling voice input in applications.
- Role in the Project: The API is used to convert the audio input (the player's spoken guess) into text. This text is then parsed to extract the guess and compared against the actual number.

Key Components:

  - speech.SpeechClient() - Creates a client to interact with the Speech-to-Text service.
  - speech.RecognitionAudio() - Sends the audio data for transcription.
  - speech.RecognitionConfig() - Specifies the configuration, including language and audio encoding settings.
  - recognize() - Transcribes the audio to text and returns the transcription.

2. Google Cloud Natural Language API

- Purpose: The Google Natural Language API is used to analyze text and extract useful insights such as sentiment, entities, and syntax.
- Role in the Project: This API is used to generate AI-powered hints based on the player's guess. By analyzing the sentiment of the player's guess, the system can provide positive or constructive feedback, encouraging the user to try again or adjust their guess.

- Key Components:
    - language_v1.LanguageServiceClient() - Creates a client to interact with the Natural Language API.
    - language_v1.Document() - Creates a document containing the text input to be analyzed.
    - analyze_sentiment() - Analyzes the sentiment of the text and returns the sentiment score, which is used to determine the tone of the feedback .

Explanation of the Workflow

1. Initialization:
    - The game initializes by generating a random number between 1 and 100.
    - The required Google Cloud credentials are set using the os module, which ensures that the Speech-to-Text and Natural Language APIs can authenticate properly.
2. User Interaction:
    - The user can either type their guess into the Tkinter input field or speak it aloud.
    - If the user chooses to speak, PyAudio captures the audio, which is then transcribed using the Speech-to-Text API.
3. Processing the Guess:
    - The transcribed text is parsed to extract the number, which is compared with the randomly generated number.
    - The game provides feedback using AI-generated hints via the Natural Language API, determining whether the guess is too high or too low.
4. Feedback and Game Progression:
    - Positive or constructive feedback is provided based on the sentiment analysis of the guess.
    - Once the player guesses the correct number, the game congratulates the user and generates a new number to continue the game.

Chapter 3

CODE

File: main.py

```
import os
import random
import pyaudio
import wave
import tkinter as tk
from tkinter import messagebox
from google.cloud import language_v1, speech

# ======== Step 1: Set Up Google Cloud Authentication =========
json_key_filename = "ai-powered-key.json"
json_key_path = os.path.join(os.getcwd(), json_key_filename)

if not os.path.exists(json_key_path):
    messagebox.showerror("Error", f"Google Cloud key file '{json_key_filename}' not found!")
    exit()

os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = json_key_path

# ======== Step 2: Initialize Google Cloud Clients =========
try:
    language_client = language_v1.LanguageServiceClient()
    speech_client = speech.SpeechClient()
except Exception as e:
```

```
    messagebox.showerror("Error", f"Failed to initialize Google Cloud clients: {e}")
    exit()

# ======== Step 3: Generate Random Number to Guess =========
actual_number = random.randint(1, 100)
print(f"🔢 Correct Number (for testing/debugging): {actual_number}")  # You can remove this

# ======== Step 4: AI-Powered Hint Generation =========
def generate_ai_hint(guess, actual_number):
    if guess == actual_number:
        return f"🎉 Congratulations! You guessed the correct number: {actual_number}!"

    direction = "higher" if guess < actual_number else "lower"
    return f"Try a {direction} number!"

# ======== Step 5: Voice Recording and Speech Recognition =========
def record_audio(filename, duration=3, rate=16000, channels=1):
    """Records an audio file from the microphone."""
    try:
        p = pyaudio.PyAudio()
        stream = p.open(format=pyaudio.paInt16, channels=channels, rate=rate, input=True, frames_per_buffer=1024)
        frames = [stream.read(1024) for _ in range(0, int(rate / 1024 * duration))]
        stream.stop_stream()
        stream.close()
        p.terminate()
```

```
        with wave.open(filename, 'wb') as wf:
            wf.setnchannels(channels)
            wf.setsampwidth(p.get_sample_size(pyaudio.paInt16))
            wf.setframerate(rate)
            wf.writeframes(b".join(frames))
    except Exception as e:
        messagebox.showerror("Error", f"Failed to record audio: {e}")

def transcribe_audio(filename):
    """Transcribes an audio file using Google Speech-to-Text API."""
    try:
        with open(filename, 'rb') as audio_file:
            content = audio_file.read()
        audio = speech.RecognitionAudio(content=content)
        config = speech.RecognitionConfig(
            encoding=speech.RecognitionConfig.AudioEncoding.LINEAR16,
            sample_rate_hertz=16000,
            language_code="en-US",
        )
        response = speech_client.recognize(config=config, audio=audio)

        if response.results:
            transcript = response.results[0].alternatives[0].transcript
            print(f" 🎙️ Detected Speech: {transcript}")  # Debugging
            return transcript
        else:
            return None
```

```
    except Exception as e:
        messagebox.showerror("Error", f"Speech recognition failed: {e}")
        return None

def extract_number_from_speech(transcript):
    """Extracts numbers from transcribed text (fixes recognition issues)."""
    words = transcript.split()
    for word in words:
        if word.isdigit():
            return int(word)  # Convert valid numbers
    return None  # If no valid number is found

# ======== Step 6: Handling Voice-Based Number Guessing =========
def voice_guess():
    """Handles voice input and extracts the guessed number."""
    audio_filename = "user_guess.wav"
    record_audio(audio_filename)
    transcription = transcribe_audio(audio_filename)
    if transcription:
        guess = extract_number_from_speech(transcription)
        return guess
    return None

# ======== Step 7: GUI Implementation =========
def check_guess():
    global actual_number
    try:
```

```
        guess = int(entry.get())
        hint = generate_ai_hint(guess, actual_number)
        result_label.config(text=hint)

        if guess == actual_number:
            messagebox.showinfo("🎉 Congratulations", f"You guessed the correct number: {actual_number}!\nStarting a new round.")
            actual_number = random.randint(1, 100)  # Restart game
            print(f"🔢 New Correct Number: {actual_number}")  # Debugging
            entry.delete(0, tk.END)  # Clear input
            result_label.config(text="New number generated. Try again!")
    except ValueError:
        messagebox.showerror("Invalid Input", "Please enter a valid number.")

def voice_guess_handler():
    """Handles the voice-based guessing."""
    guess = voice_guess()
    if guess is not None:
        entry.delete(0, tk.END)
        entry.insert(0, str(guess))
        check_guess()
    else:
        messagebox.showerror("Error", "Could not understand the voice input. Please try again.")

# ======== Step 8: Create GUI Window =========
root = tk.Tk()
root.title("🎤 AI Number Guessing Game")
```

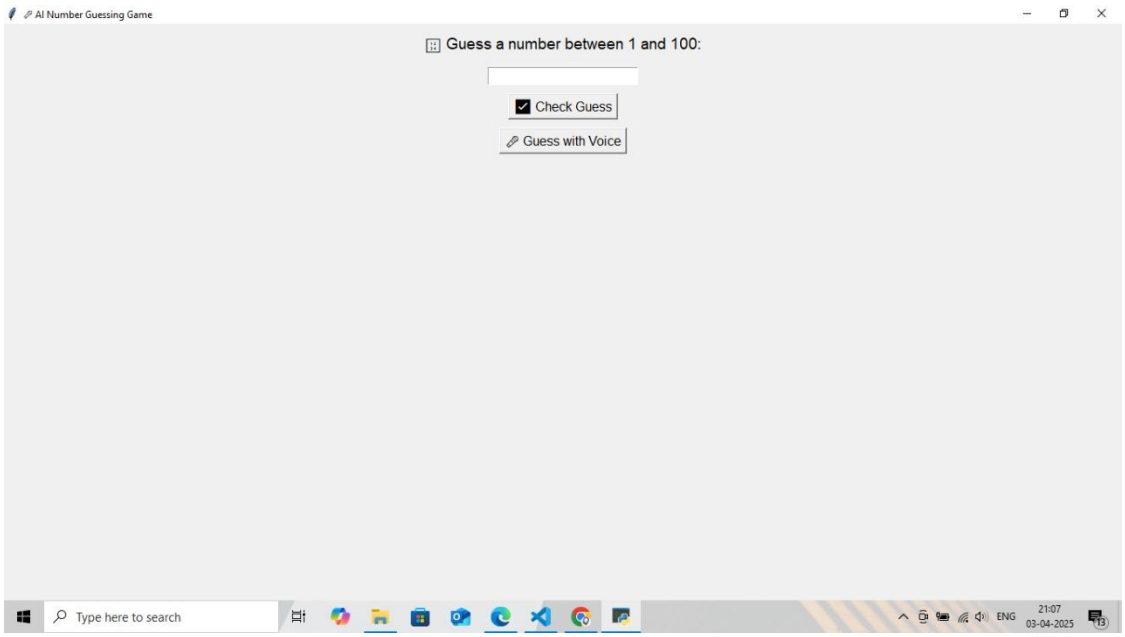
```
root.geometry("500x400")


tk.Label(root, text="🔢 Guess a number between 1 and 100:", font=("Arial", 14)).pack(pady=10)

entry = tk.Entry(root, font=("Arial", 12))

entry.pack(pady=5)


check_button = tk.Button(root, text="✅ Check Guess", font=("Arial", 12), command=check_guess)

check_button.pack(pady=5)


voice_button = tk.Button(root, text="🎤 Guess with Voice", font=("Arial", 12), command=voice_guess_handler)

voice_button.pack(pady=5)


result_label = tk.Label(root, text="", font=("Arial", 12), fg="blue")

result_label.pack(pady=10)


root.mainloop()



{
```

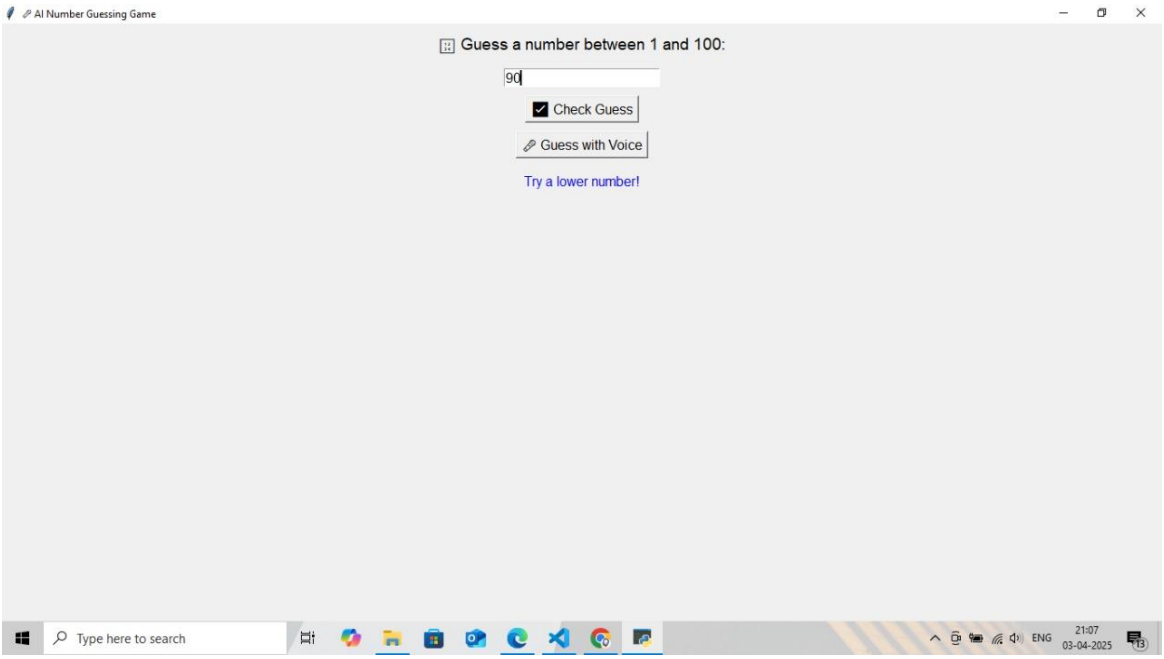
Chapter 4

Outcome:-

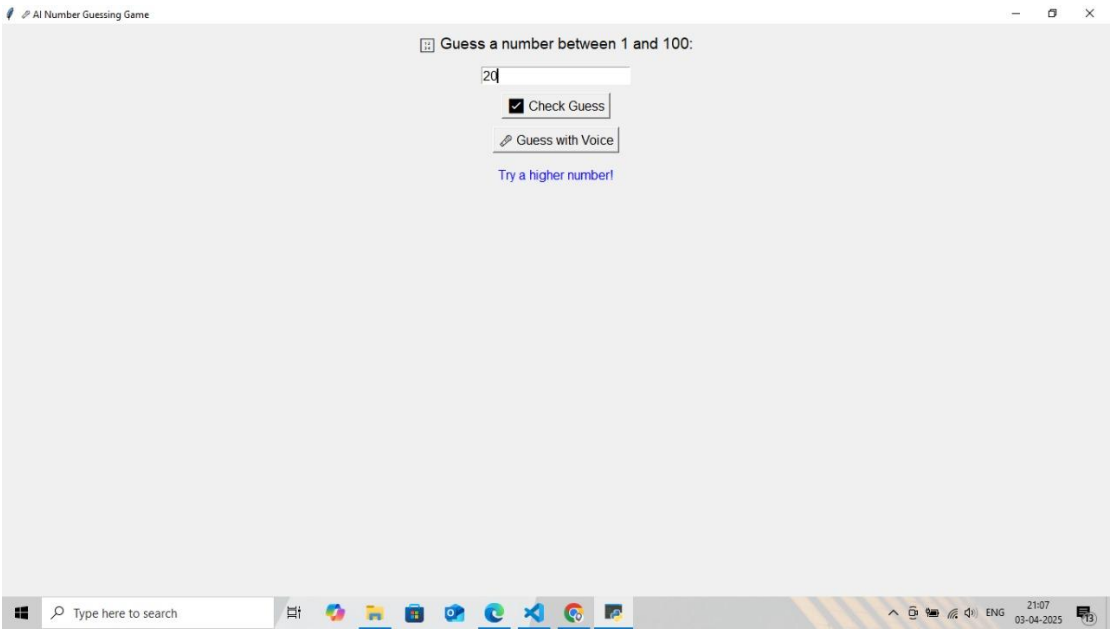
Running **main.py** code it will open an window.

This image shows an AI-powered Number Guessing Game interface. The game prompts the user to guess a number between 1 and 100. There are two buttons: "Check Guess" for verifying the number and "Guess with Voice" for voice input.

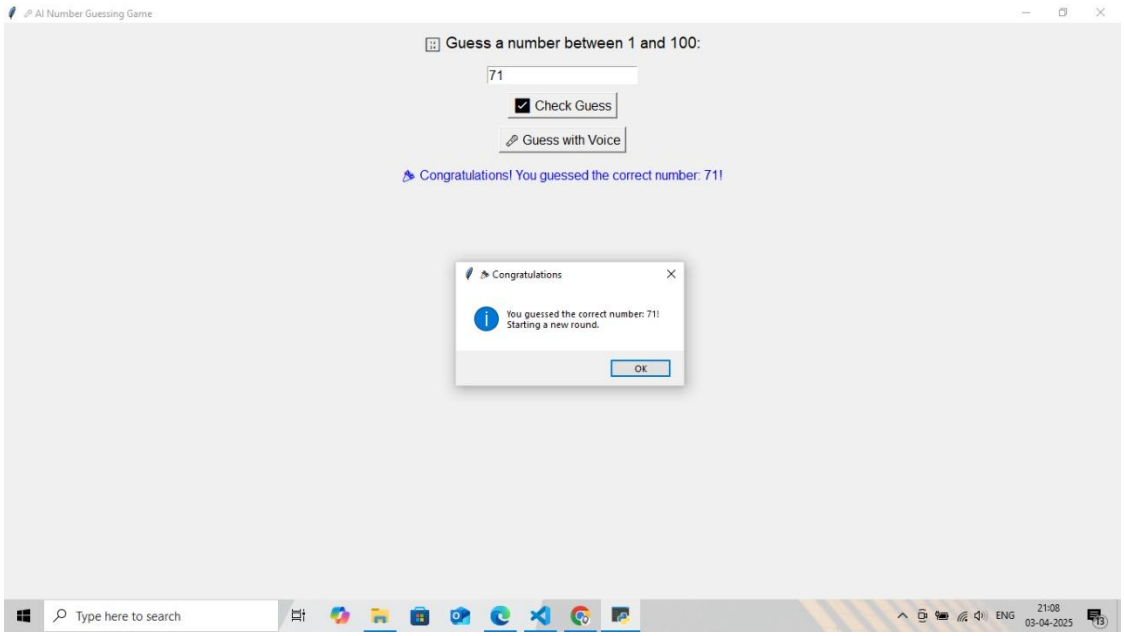




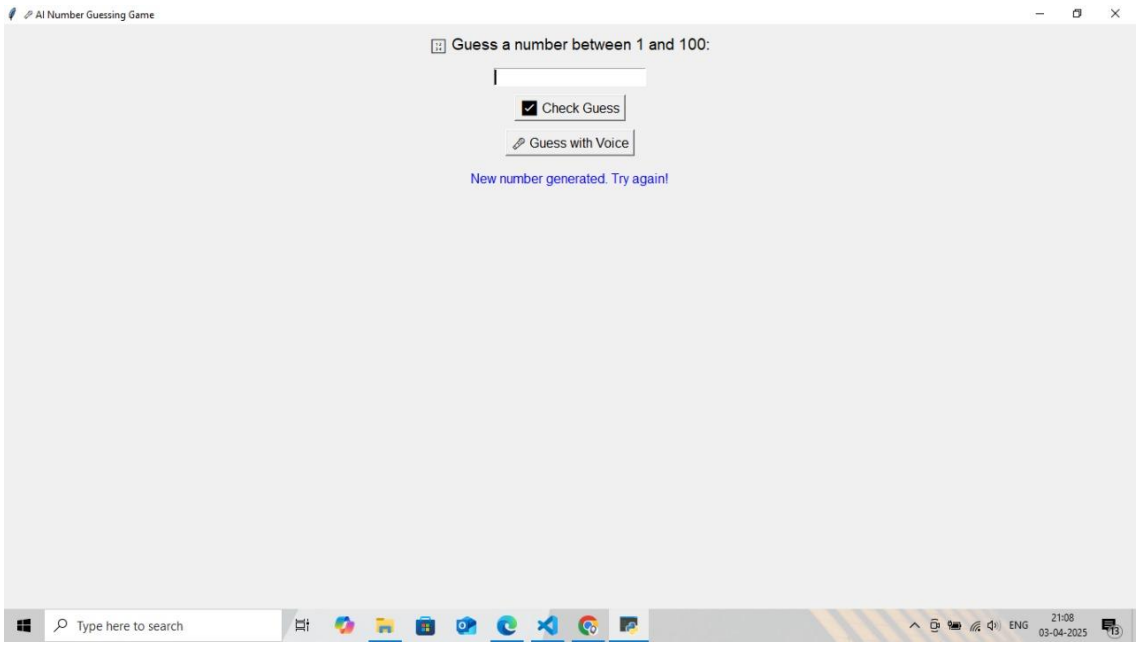
Number Entered by User is 90. But this is not that number which is generated by AI . AI number is less than the number entered by User.



Number Entered by User is 20. But this is not that number which is generated by AI . AI number is greater than the number entered by User.



Number Entered by User 71. And the number generated by AI is also 71. So the user has guessed the right number. So a dialog box will be opened if the number is right , will congratulate the user.



After Correctly Guess ,again a game will start , AI will generate the new Number and user have to guess the Number

References

For your AI Number Guessing Game project, here are some useful references:

1. Python Tkinter (GUI Development)
   - Python's official documentation on Tkinter: https://docs.python.org/3/library/tkinter.html
   - Tkinter tutorial: https://realpython.com/python-gui-tkinter/

2. Number Guessing Game in Python
   - Simple number guessing game in Python: https://www.geeksforgeeks.org/number-guessing-game-in-python/
   - Python random module (for generating random numbers): https://docs.python.org/3/library/random.html

3. Speech Recognition in Python (for Voice Input)
   - Python SpeechRecognition library: https://pypi.org/project/SpeechRecognition/
   - Google Speech API integration: https://realpython.com/python-speech-recognition/

4. Game Logic & AI for Number Guessing
   - Implementing AI-based number guessing: https://www.analyticsvidhya.com/blog/
   - Binary search algorithm for optimized guessing: https://www.geeksforgeeks.org/binary-search/