

# Number Guessing Game Using Streamlit

## 1. Introduction

This project is a web-based "Number Guessing Game," where players attempt to guess a randomly selected number within a predefined range. Built using Python and Streamlit, the game offers an interactive interface, immediate feedback, and attempt tracking to enhance user engagement.

## 2. Objective

- Develop an intuitive number guessing game with an interactive UI.
- Provide real-time feedback to guide the player.
- Implement input validation for a smooth user experience.
- Track the number of attempts for better engagement.

## 3. Methodology

### Technologies Used:

- **Python:** Used for core logic and game functionality.
- **Streamlit:** Enables the creation of a web-based user interface.
- **Random Module:** Generates the target number dynamically.

### Implementation Steps:

1. Generate a random number within a specified range (1 to 100).
2. Accept user input via an interactive web interface.
3. Validate user input to ensure it falls within the correct range.
4. Compare the guessed number with the target and provide feedback.
5. Maintain a count of attempts taken by the player.
6. Display a success message when the player guesses correctly.

## 4. Code and Implementation

Below is the Python implementation using Streamlit:

```

main.py x
1 import streamlit as st
2 import random
3 usage
4 def main():
5     st.title("🎮 Guess the Number Game")
6     # Session state to store game variables
7     if 'target_number' not in st.session_state:
8         st.session_state.target_number = random.randint(a=1, b=100)
9     if 'attempts' not in st.session_state:
10        st.session_state.attempts = 0
11    if 'game_over' not in st.session_state:
12        st.session_state.game_over = False
13    st.write("I'm thinking of a number between 1 and 100. Can you guess it?")
14    guess = st.number_input("Enter your guess:", min_value=1, max_value=100, step=1, format="%d")
15    if st.button("Submit Guess") and not st.session_state.game_over:
16        st.session_state.attempts += 1
17        if guess < st.session_state.target_number:
18            st.warning("Too low! Try again.")
19        elif guess > st.session_state.target_number:
20            st.warning("Too high! Try again.")
21        else:
22            st.success(f"Congratulations! You guessed the number in {st.session_state.attempts} attempts.")
23            st.session_state.game_over = True
24    if st.session_state.game_over:
25        if st.button("Play Again"):
26            st.session_state.target_number = random.randint(a=1, b=100)
27            st.session_state.attempts = 0
28            st.session_state.game_over = False
29            st.experimental_rerun()
30 if __name__ == "__main__":
31     main()

```

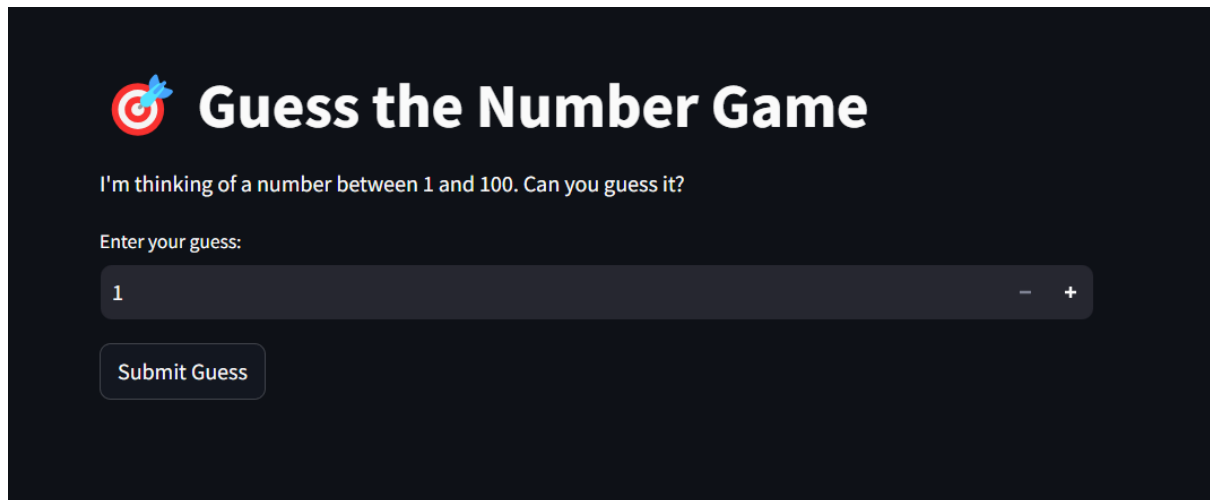
## 5. Results and Observations

### Gameplay Insights:

- The application successfully generates a random number for each session.
- Players receive instant feedback on whether their guess is too high, too low, or correct.
- The number of attempts is accurately tracked and displayed.
- Players can restart the game after completing a round.

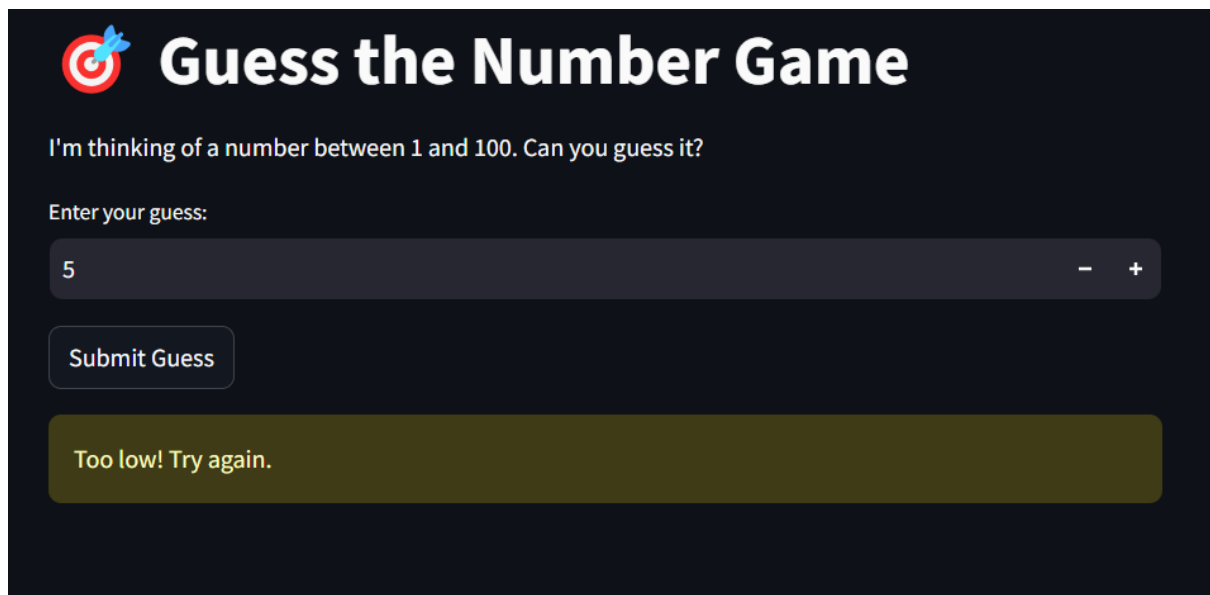
### Screenshots:

1. **Start Screen:** The interface prompts the player to input a guess.



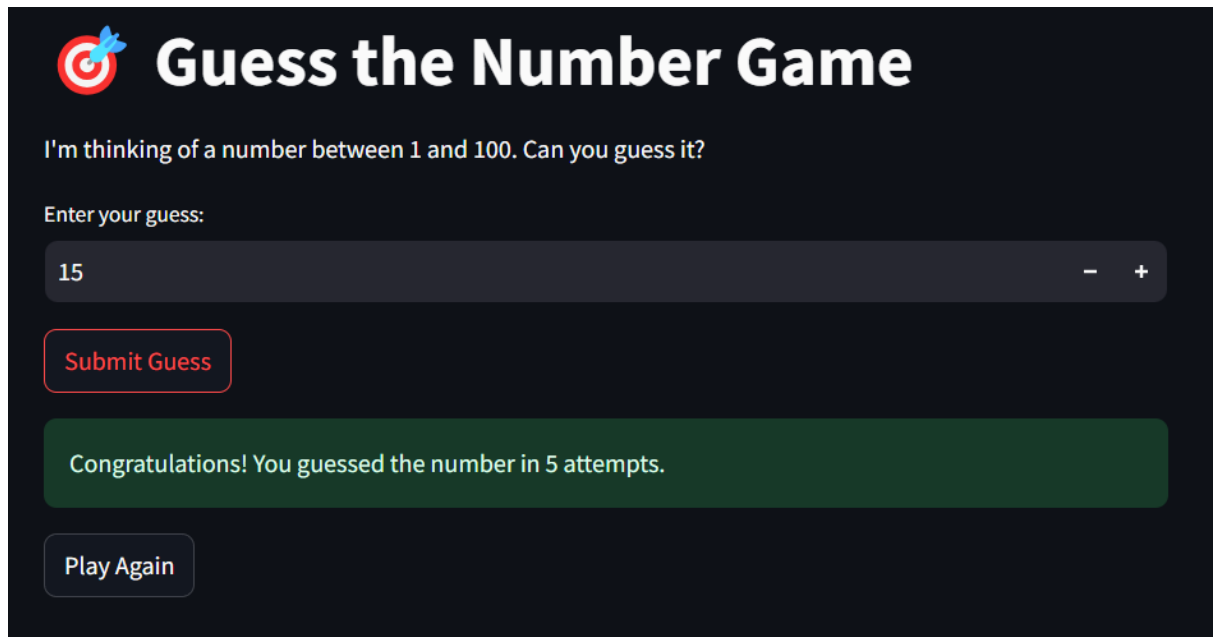
The image shows the initial state of a web application titled "Guess the Number Game". It features a dark blue background. At the top left is a logo consisting of a red bullseye with a blue arrow hitting the center. To the right of the logo is the title "Guess the Number Game" in a large, bold, white sans-serif font. Below the title is a line of white text: "I'm thinking of a number between 1 and 100. Can you guess it?". Underneath this is a label "Enter your guess:" in a smaller white font. Below the label is a dark grey input field containing the number "1". To the right of the input field are two small white icons: a minus sign and a plus sign. Below the input field is a rounded rectangular button with the text "Submit Guess" in white.

2. **Hint Messages:** Provides guidance based on the guessed number.



The image shows the same web application as before, but now with a hint message. The input field now contains the number "5". Below the input field and the "Submit Guess" button is a new element: a horizontal bar with a dark olive green background and rounded corners. Inside this bar, the text "Too low! Try again." is written in a light yellow-green font.

3. **Success Message:** Displays the number of attempts taken to guess correctly.



## 6. Conclusion

This project successfully demonstrates the use of Python and Streamlit to create an interactive and engaging number guessing game. By implementing real-time feedback, input validation, and session state tracking, the game provides an enjoyable experience for users. The project meets all the objectives outlined and serves as an example of simple yet effective web-based game development.

## 7. Project Files and Resources

- **Main Code File:** `main.py`
- **Required Libraries:** `streamlit`, `random`
- **Dataset:** Not applicable (random number generation used)
- **Installation Instructions:**

```
1.pip install streamlit
2.streamlit run main.py
```