

Python

Password Generator

```
1 import random
2 import string
3
4 def generate_password(length=12):
5     # Define character pool
6     characters = string.ascii_letters + string.digits + string.punctuation
7     # Randomly select characters from the pool
8     password = ''.join(random.choice(characters) for _ in range(length))
9     return password
10
11 # Generate and display the password
12 print("Generated Password:", generate_password(12))
```

input

Generated Password: d~:[I\$zT1#.&

To-Do List (CLI)

```
tasks = []

while True:
    # Display the menu
    print("\n1. Add Task\n2. View Tasks\n3. Remove Task\n4. Exit")
    choice = input("Enter choice: ")

    if choice == "1":
        task = input("Enter task: ")
        tasks.append(task)
        print("Task added!")

    elif choice == "2":
        print("\nTo-Do List:")
        for idx, task in enumerate(tasks, 1):
            print(f"{idx}. {task}")

    elif choice == "3":
        task_num = int(input("Enter task number to remove: "))
        if 0 < task_num <= len(tasks):
            tasks.pop(task_num - 1)
            print("Task removed!")

    elif choice == "4":
        break

    else:
        print("Invalid choice. Try again.")
```



1. Add Task
2. View Tasks
3. Remove Task
4. Exit

Enter choice: 1

Enter task: Morning Exercise

Task added!

1. Add Task
2. View Tasks
3. Remove Task
4. Exit

Enter choice: 1

Enter task: sleep Early

Task added!

1. Add Task
2. View Tasks
3. Remove Task
4. Exit

Enter choice: 2

To-Do List:

1. Morning Exercise
2. sleep Early

1. Add Task
2. View Tasks
3. Remove Task
4. Exit

Enter choice: 3

Enter task number to remove: 2

Task removed!

1. Add Task
2. View Tasks
3. Remove Task
4. Exit

Enter choice: 2

To-Do List:

1. Morning Exercise

1. Add Task
2. View Tasks
3. Remove Task
4. Exit

Enter choice: 4

Weather App (API-based)

```
1 import requests
2
3 # Use your provided API key
4 API_KEY = "8f2d6822fb2e4524adf20f8132e6f463"
5 city = input("Enter city name: ")
6
7 # Construct the API URL
8 url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={API_KEY}&units=metric"
9
10 # Fetch the weather data
11 response = requests.get(url).json()
12
13 # Check if the response is valid
14 if response["cod"] == 200:
15     print(f"\nCity: {response['name']}")
16     print(f"Temperature: {response['main']['temp']}°C")
17     print(f"Weather: {response['weather'][0]['description']}")
18 else:
19     print("\nCity not found!")
20
```

input

Enter city name: London

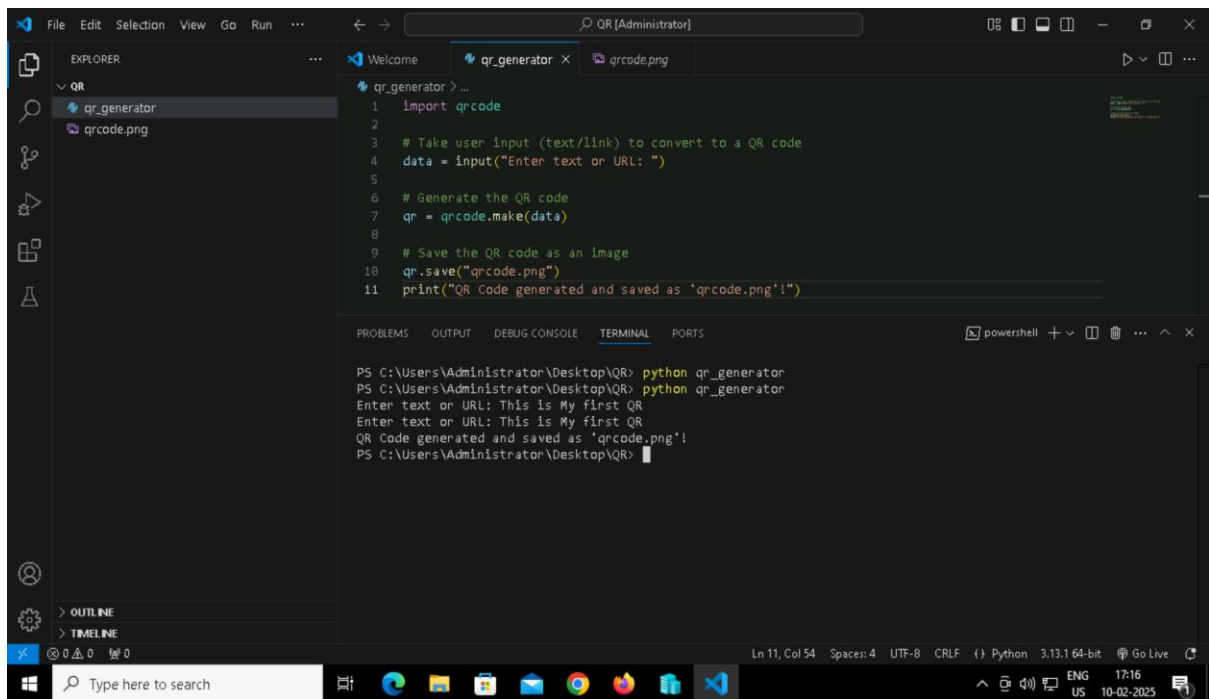
City: London
Temperature: 3.94°C
Weather: overcast clouds

Number Guessing Game

```
1 import random
2
3 # Generate a random number between 1-100
4 number = random.randint(1, 100)
5
6 while True:
7     guess = int(input("Guess the number (1-100): "))
8
9     if guess < number:
10         print("Too low! Try again.")
11     elif guess > number:
12         print("Too high! Try again.")
13     else:
14         print("Congratulations! You guessed it right.")
15         break
```

Guess the number (1-100): 12
Too low! Try again.
Guess the number (1-100): 68
Too high! Try again.
Guess the number (1-100): 50
Too low! Try again.
Guess the number (1-100): 60
Too high! Try again.
Guess the number (1-100): 55
Too high! Try again.
Guess the number (1-100): 52
Congratulations! You guessed it right.

QR Code Generator



This screenshot shows the Visual Studio Code editor with a Python script named `qr_generator.py` and its execution output in the terminal. The Explorer sidebar on the left shows the project structure with a folder named `QR` containing `qr_generator` and `qrcode.png`. The script uses the `qrcode` library to generate a QR code from user input and save it as `qrcode.png`.

```
qr_generator > ...
1  import qrcode
2
3  # Take user input (text/link) to convert to a QR code
4  data = input("Enter text or URL: ")
5
6  # Generate the QR code
7  qr = qrcode.make(data)
8
9  # Save the QR code as an image
10 qr.save("qrcode.png")
11 print("QR Code generated and saved as 'qrcode.png'!")
```

The terminal output shows the command `python qr_generator` being executed, followed by the user input `This is My first QR` and the confirmation message `QR Code generated and saved as 'qrcode.png'!`.

