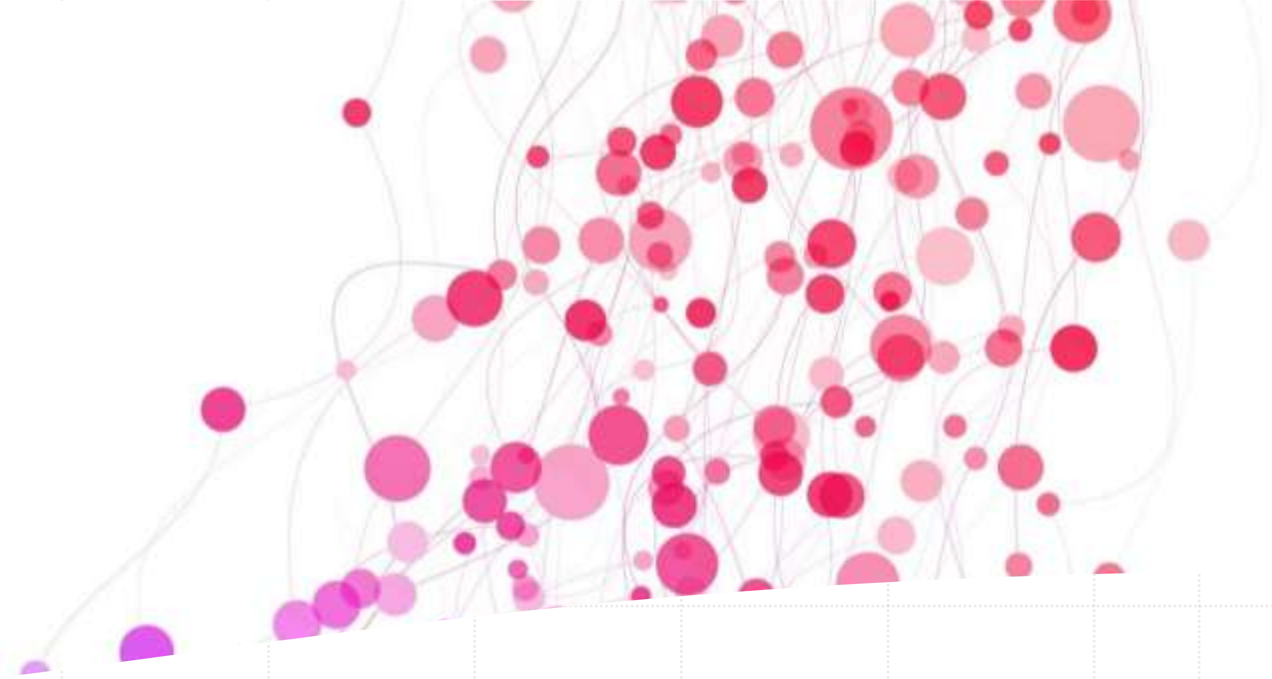


A party planner application

Elakkiya

Content

- Introduction
- Python used
- Work flow
- Module
- benifits

Introduction

- A party planner application is a tool that helps you organize events easily. It allows you to manage guest lists, track your budget, schedule tasks, and coordinate with vendors. You can send invitations, track who is coming, and make sure everything is ready on time. It also provides ideas for themes, decorations, and menus. These apps are accessible on web or mobile devices, making it convenient to plan your party from anywhere.

Python used

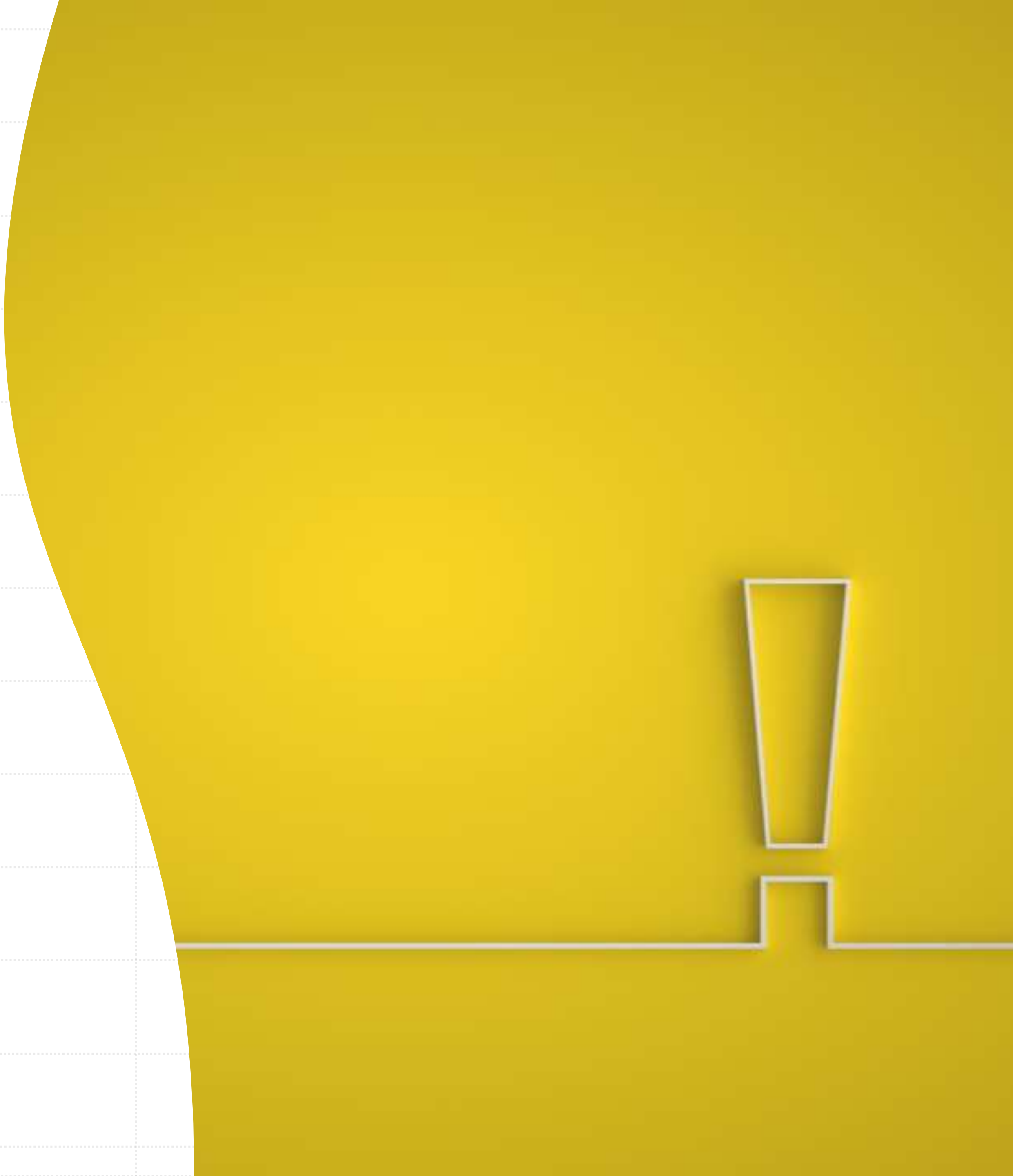
- A class is a blueprint for creating objects. Classes encapsulate data for the object and methods to manipulate that data.
- 1. Defining a ClassTo define a class, you use the class keyword followed by the class name and a colon. Inside the class, you define methods and attributes.

Work flow

- 1. Initialization
- 2. Menu Loop
- A) Add guest
- B) Remove guest
- C) View guest list
- D) Exit

Module

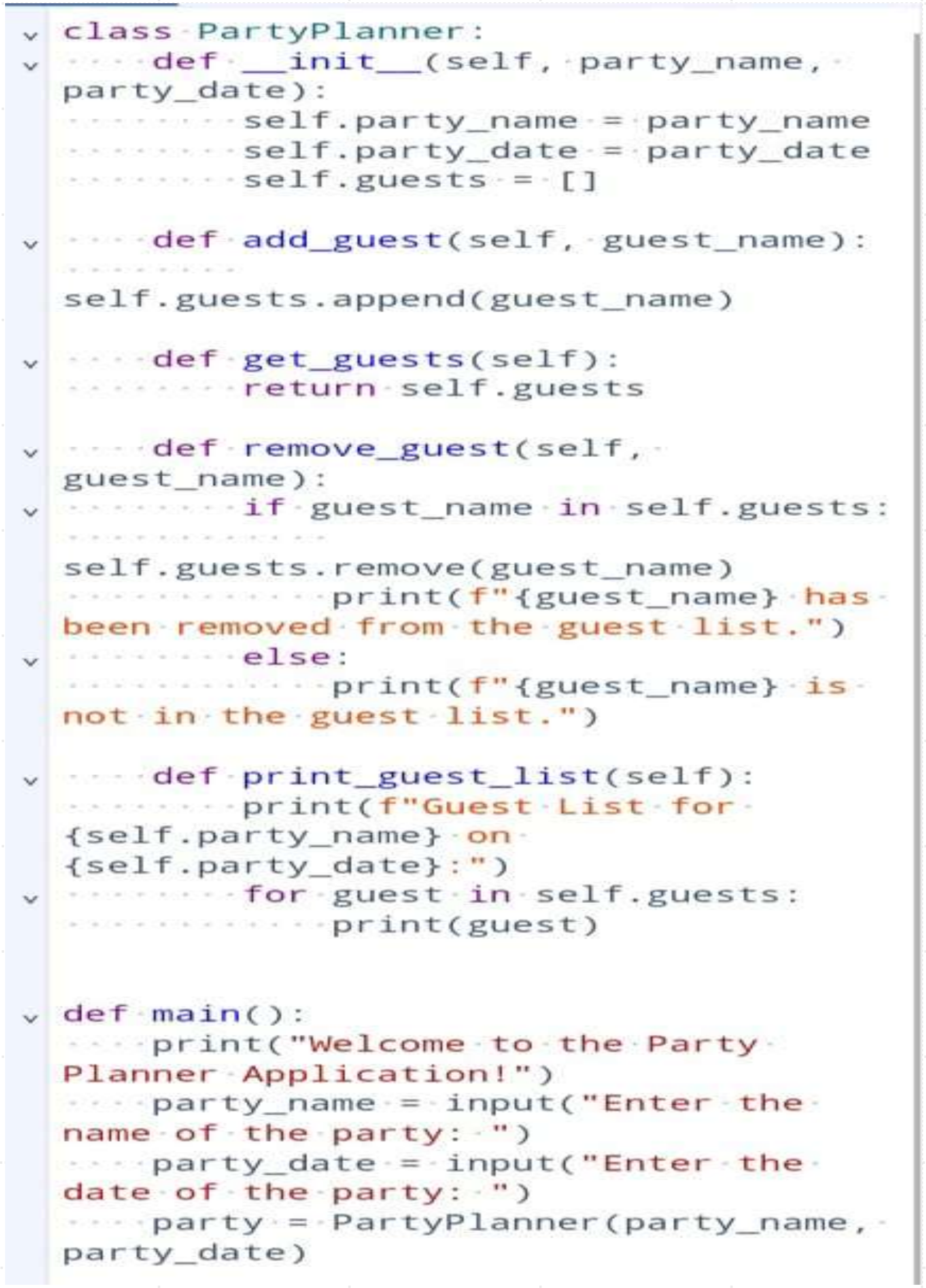
- Class `PartyPlanner`:   Initializes a party with attributes `party_name`, `party_date`, and manages a list of `guests`.Methods: `add_guest` adds guests, `get_guests` retrieves the guest list, `remove_guest` removes guests if they exist, `print_guest_list` displays guests with party details.Main Function (`main()`): Manages user interaction, initializes a `PartyPlanner` instance, and presents a menu for adding guests, removing guests, viewing the guest list, and exiting.Menu Options:
- 1. Add Guest prompts for name
- 2. Remove Guest prompts for removal
- 3. View Guest List displays current guests
- 4. Exit . Input Handling: Validates and processes user choices using `input()` function.Interaction: Coordinates user inputs with `PartyPlanner` methods to update or display guest information.Execution Flow: Starts at `main()`, loops until user exits, ensuring continuous management of the guest list for the specified party.



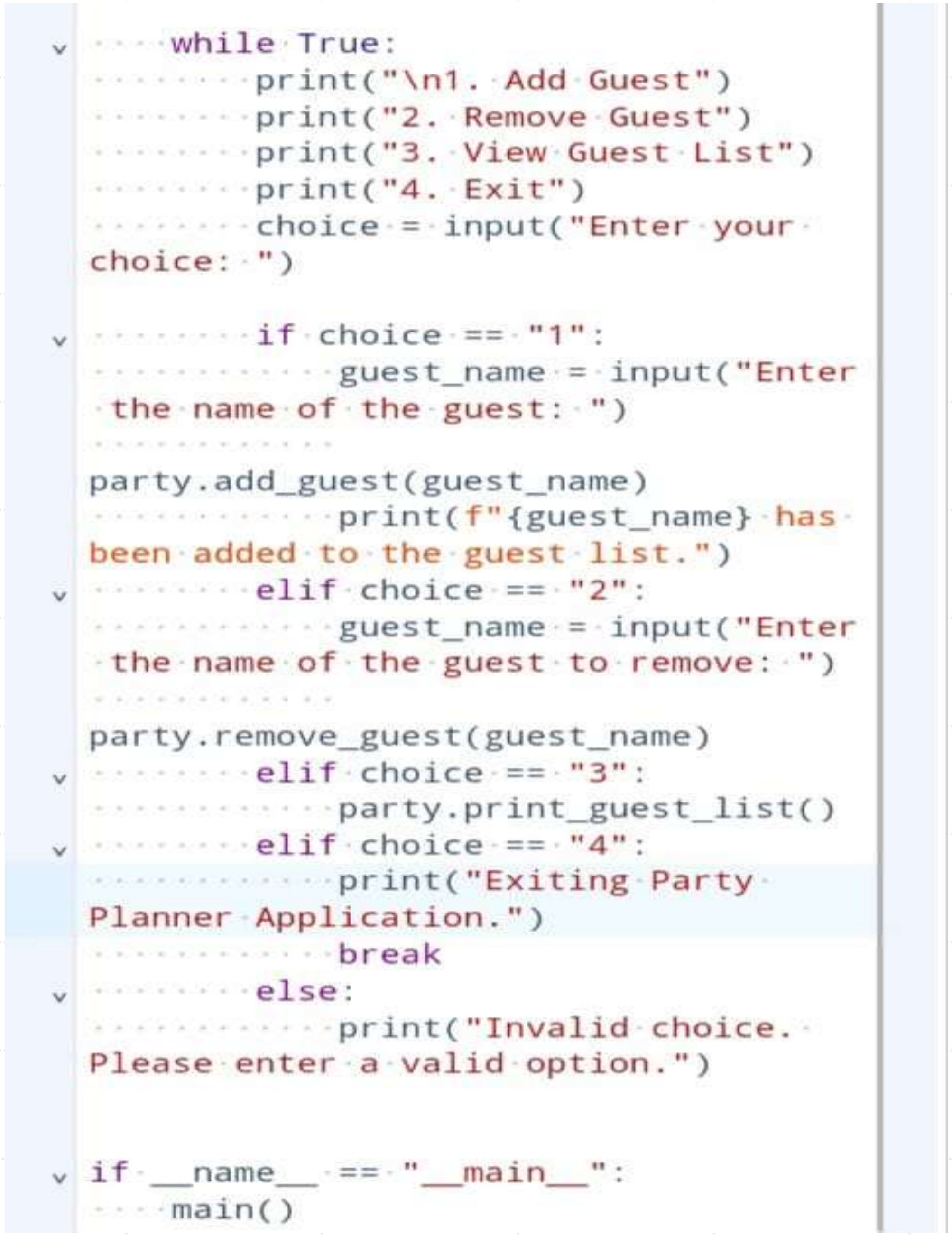
Benifits

- 1 ORGANIZATION: It helps keep all party details in one place, making it easier to manage tasks like guest lists, invitations, and schedules.
- 2. EFFICIENCY: Automates tasks such as sending invitations, tracking RSVPs, and managing budgets, saving time and reducing stress.
- 3. COORDINATION: Facilitates communication with vendors and collaborators, ensuring everything from catering to decorations is well-coordinated.
- 4. CUSTOMIZATION: Provides templates and ideas for themes, decorations, and menus, allowing for personalized and creative event planning.
- 5. ACCESSIBILITY: Can be accessed from anywhere via mobile devices or computers, enabling easy updates and collaboration among organizers.
- 6. BUDGET MANAGEMENT: Helps set and track budgets, ensuring expenses stay within limits and providing financial clarity.
- 7.GUEST MANAGEMENT: Simplifies managing guest lists, dietary preferences, and special requests, enhancing the guest experience.

Program:

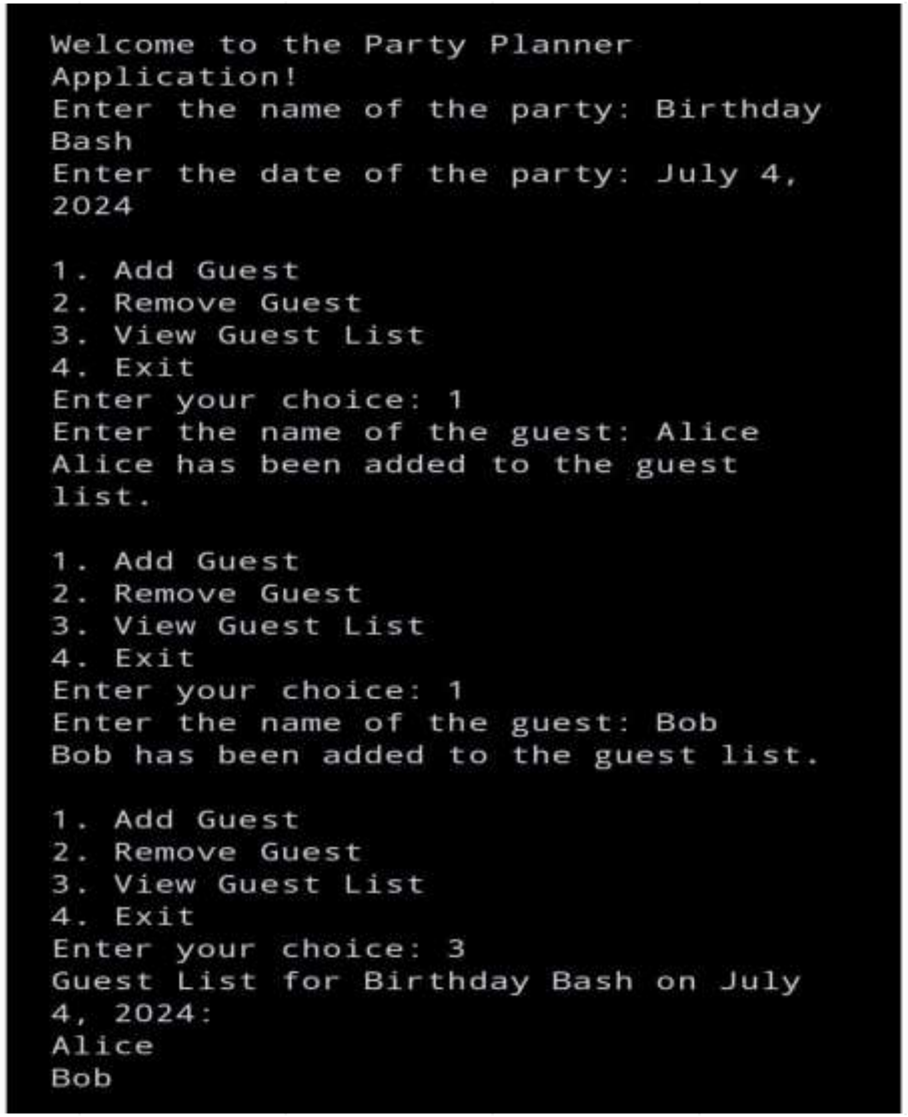


```python
class PartyPlanner:
    def __init__(self, party_name, party_date):
        self.party_name = party_name
        self.party_date = party_date
        self.guests = []

    def add_guest(self, guest_name):
        self.guests.append(guest_name)

    def get_guests(self):
        return self.guests

    def remove_guest(self, guest_name):
        if guest_name in self.guests:
            self.guests.remove(guest_name)
            print(f"{guest_name} has been removed from the guest list.")
        else:
            print(f"{guest_name} is not in the guest list.")

    def print_guest_list(self):
        print(f"Guest List for {self.party_name} on {self.party_date}:")
        for guest in self.guests:
            print(guest)


def main():
    print("Welcome to the Party Planner Application!")
    party_name = input("Enter the name of the party: ")
    party_date = input("Enter the date of the party: ")
    party = PartyPlanner(party_name, party_date)
```

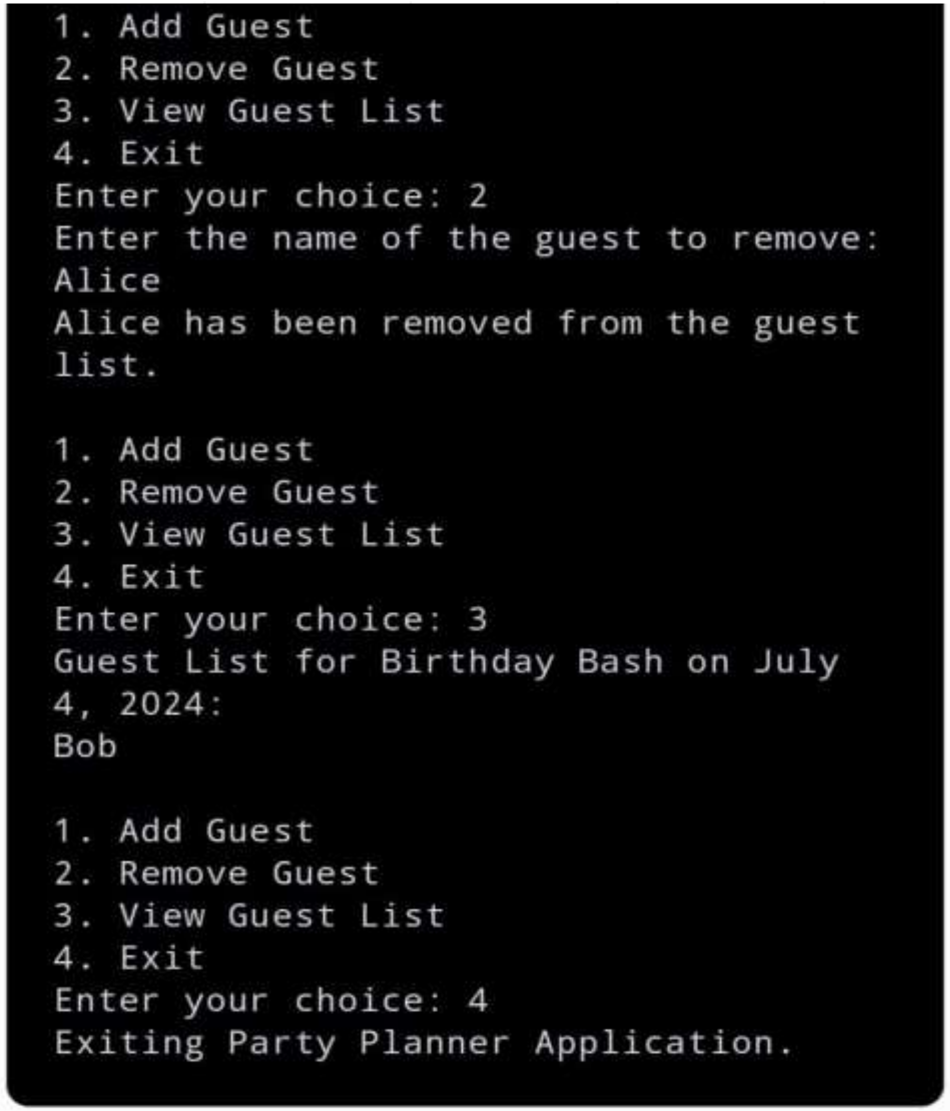


```
    while True:
        print("\n1. Add Guest")
        print("2. Remove Guest")
        print("3. View Guest List")
        print("4. Exit")
        choice = input("Enter your choice: ")

        if choice == "1":
            guest_name = input("Enter the name of the guest: ")
            party.add_guest(guest_name)
            print(f"{guest_name} has been added to the guest list.")
        elif choice == "2":
            guest_name = input("Enter the name of the guest to remove: ")
            party.remove_guest(guest_name)
        elif choice == "3":
            party.print_guest_list()
        elif choice == "4":
            print("Exiting Party Planner Application.")
            break
        else:
            print("Invalid choice. Please enter a valid option.")


if __name__ == "__main__":
    main()
```

Output:



```
Welcome to the Party Planner
Application!
Enter the name of the party: Birthday
Bash
Enter the date of the party: July 4,
2024

1. Add Guest
2. Remove Guest
3. View Guest List
4. Exit
Enter your choice: 1
Enter the name of the guest: Alice
Alice has been added to the guest
list.

1. Add Guest
2. Remove Guest
3. View Guest List
4. Exit
Enter your choice: 1
Enter the name of the guest: Bob
Bob has been added to the guest list.

1. Add Guest
2. Remove Guest
3. View Guest List
4. Exit
Enter your choice: 3
Guest List for Birthday Bash on July
4, 2024:
Alice
Bob
```



```
1. Add Guest
2. Remove Guest
3. View Guest List
4. Exit
Enter your choice: 2
Enter the name of the guest to remove:
Alice
Alice has been removed from the guest
list.

1. Add Guest
2. Remove Guest
3. View Guest List
4. Exit
Enter your choice: 3
Guest List for Birthday Bash on July
4, 2024:
Bob

1. Add Guest
2. Remove Guest
3. View Guest List
4. Exit
Enter your choice: 4
Exiting Party Planner Application.
```

Thank you