Here's a brief write-up explaining how I implemented the various functions in the Contact Management System and the challenges encountered during development:

Contact Class:

The Contact class is implemented to represent a single contact with attributes like name, phone number, email, and address. It has a straightforward constructor (__init__ method) that initializes these attributes when a new contact object is created.

ContactManagementSystem Class:

The ContactManagementSystem class is implemented to manage contacts. It has a list contacts to store all the contact objects.

Add Contact Functionality:

The "Add Contact" functionality is implemented within the main function using an if-else block. When the user chooses option "1", they are prompted to input the contact details such as name, phone number, email, and address. Appropriate validation checks are performed for the phone number and email.

If the input is invalid, an error message is displayed, and the user is prompted to enter valid details.

Once the input is validated, a new Contact object is created, and it is added to the contacts list of the ContactManagementSystem using contact_manager.contacts.append(contact).

View Contacts Functionality:

The "View Contacts" functionality is implemented within the main function using an if-else block. When the user chooses option "2", the system checks if there are any contacts in the list.

If there are no contacts, a message is displayed to inform the user. Otherwise, the details of each contact in the list are displayed using a loop.

Search Contact Functionality:

The "Search Contact" functionality is implemented within the main function using an if-else block. When the user chooses option "3", they are prompted to enter a name to search for.

The system then iterates through the contacts list and finds any contacts whose names match the search query (case-insensitive).

If matching contacts are found, their details are displayed. If no matching contacts are found, an appropriate message is displayed.

Delete Contact Functionality:

The "Delete Contact" functionality is implemented within the main function using an if-else block. When the user chooses option "4", they are prompted to enter the name of the contact to delete.

The system then searches for a contact with the provided name (case-insensitive) in the contacts list.

If a matching contact is found, it is removed from the list. If no matching contact is found, an appropriate message is displayed.

Exit Functionality:

The "Exit" functionality is implemented within the main function using an if-else block. When the user chooses option "5", the loop breaks, and the program ends, exiting the Contact Management System.

Challenges Encountered:

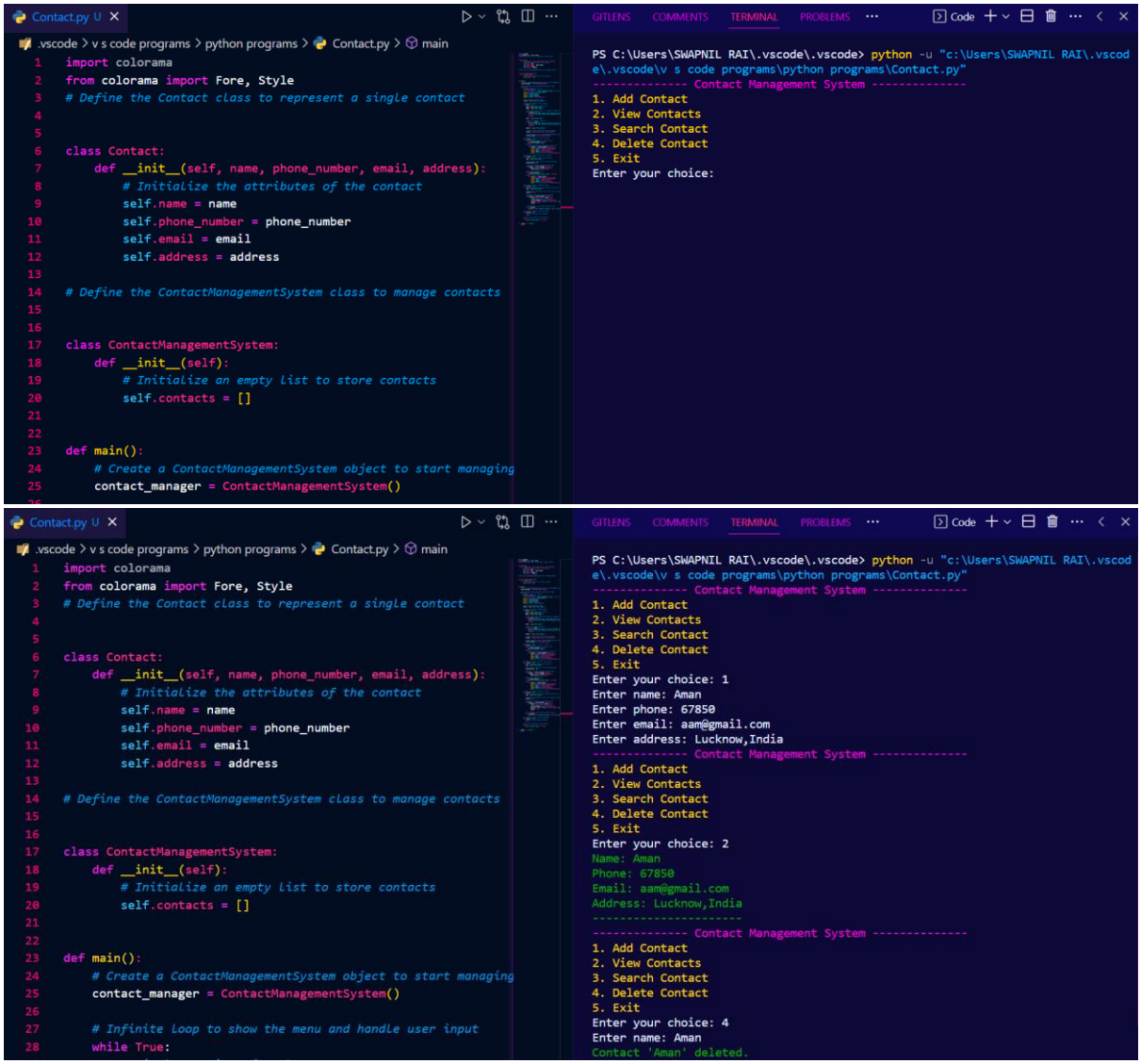
- One of the main challenges was ensuring that the user inputs were properly validated before adding a contact. I implemented checks for the phone number to be an integer and the email to contain '@'. If the inputs were invalid, the system prompted the user to provide valid details.

- Another challenge was handling the case-insensitive search for contacts by name. I used name.lower() to convert both the search query and the contact names to lowercase for comparison.

- Additionally, designing a user-friendly and visually appealing interface in the terminal posed a challenge. To overcome this, I used the colorama library to add colors and formatting to the text, making the system more visually appealing and easier to use.

Overall, the Contact Management System was successfully implemented using Python, and the functionalities of adding, viewing, searching, and deleting contacts were achieved while addressing the challenges encountered during development.

Sample:



```
Contact.py U
.vscode > v s code programs > python programs > Contact.py > main
import colorama
from colorama import Fore, Style
# Define the Contact class to represent a single contact


class Contact:
    def __init__(self, name, phone_number, email, address):
        # Initialize the attributes of the contact
        self.name = name
        self.phone_number = phone_number
        self.email = email
        self.address = address

# Define the ContactManagementSystem class to manage contacts


class ContactManagementSystem:
    def __init__(self):
        # Initialize an empty list to store contacts
        self.contacts = []


def main():
    # Create a ContactManagementSystem object to start managing
    contact_manager = ContactManagementSystem()
```

```
GITLENS  COMMENTS  TERMINAL  PROBLEMS  ...  Code
PS C:\Users\SWAPNIL RAI\.vscode\.vscode> python -u "c:\Users\SWAPNIL RAI\.vscode\.vscode\v s code programs\python programs\Contact.py"
-------------- Contact Management System --------------
1. Add Contact
2. View Contacts
3. Search Contact
4. Delete Contact
5. Exit
Enter your choice:
```



```
Contact.py U
.vscode > v s code programs > python programs > Contact.py > main
import colorama
from colorama import Fore, Style
# Define the Contact class to represent a single contact


class Contact:
    def __init__(self, name, phone_number, email, address):
        # Initialize the attributes of the contact
        self.name = name
        self.phone_number = phone_number
        self.email = email
        self.address = address

# Define the ContactManagementSystem class to manage contacts


class ContactManagementSystem:
    def __init__(self):
        # Initialize an empty list to store contacts
        self.contacts = []


def main():
    # Create a ContactManagementSystem object to start managing
    contact_manager = ContactManagementSystem()

    # Infinite loop to show the menu and handle user input
    while True:
```

```
GITLENS  COMMENTS  TERMINAL  PROBLEMS  ...  Code
PS C:\Users\SWAPNIL RAI\.vscode\.vscode> python -u "c:\Users\SWAPNIL RAI\.vscode\.vscode\v s code programs\python programs\Contact.py"
-------------- Contact Management System --------------
1. Add Contact
2. View Contacts
3. Search Contact
4. Delete Contact
5. Exit
Enter your choice: 1
Enter name: Aman
Enter phone: 67850
Enter email: aam@gmail.com
Enter address: Lucknow,India
-------------- Contact Management System --------------
1. Add Contact
2. View Contacts
3. Search Contact
4. Delete Contact
5. Exit
Enter your choice: 2
Name: Aman
Phone: 67850
Email: aam@gmail.com
Address: Lucknow,India
----------------------
-------------- Contact Management System --------------
1. Add Contact
2. View Contacts
3. Search Contact
4. Delete Contact
5. Exit
Enter your choice: 4
Enter name: Aman
Contact 'Aman' deleted.
```