Project Overview:

The objective of this project “Implement edit book feature” is to create a simple book for users to create and add existing contacts and later to be able to edit or update name, number and email of those contacts.

Explanation:

- Firstly we are giving attributes to the contact which includes 3 attributes: “name”, “phone number”, and “email”.
- Here, the “__init__” function means initialize, its work is to initialize a “contact” with provided details.

```
    # Giving attributes for contact

class Contact:
    def __init__(self, name, phone_number, email):
        self.name = name
        self.phone_number = phone_number
        self.email = email
```

- Now creating a new contact by using “add_contact” method which adds “contact” it to the list.

```
    # creating a new contact and adding it to the list

    def add_contact(self, name, phone_number, email):
        contact = Contact(name, phone_number, email)
        self.contacts.append(contact)
        print("Contact added successfully.")
```

- The “display_contacts” method prints the details of each contact in the address book.

```
# printing details of each contact

    def display_contacts(self):
        print("Contacts:")
        for idx, contact in enumerate(self.contacts):
            print(f"{idx+1}. Name: {contact.name}, Phone: {contact.phone_number}, Email: {contact.email}")
```

- The “edit_contact” method allows the user to edit their contact by providing index or new details.
- The “len(self.contact)” is used to determine if the provided index is within the range of indices for the “contacts”.

```
34      # checking if the index is valid then update the contact details
35
36          def edit_contact(self, index, name, phone_number, email):
37              if 0 <= index < len(self.contacts):
38                  contact = self.contacts[index]
39                  contact.name = name
40                  contact.phone_number = phone_number
41                  contact.email = email
42                  print("Contact edited successfully.")
43              else:
44                  print("Invalid index.")
```

- In the “main” function, an instance “AddressBook” is created.
- Then we filled in the details or index of each contact by giving them a specific name, number, and mail by using the “ass_contact” method.

```
46      # adding sample contacts
47
48      def main():
49          address_book = AddressBook()
50
51          address_book.add_contact("haseeb", "1234567890", "haseeb@gmail.com")
52          address_book.add_contact("hamza", "9876543210", "hamza@gmail.com")
53          address_book.add_contact("basit", "123543765", "basit@gmail.com")
54          address_book.add_contact("saad", "090912333", "saad@gmail.com")
55          address_book.add_contact("shakeela", "030120213", "shakeela@gmail.com")
56
```

- Then the list of contacts is displayed by using the “display_contacts()” method.
- Ensured again that the index is within the range by using the “if” function and “index < len”.

```
57    # display contacts
58
59        address_book.display_contacts()
60
61        index = int(input("Enter index of contact to edit: ")) - 1
62        if 0 <= index < len(address_book.contacts):
63
```

- Now using a method of giving new details after selecting a specific index to edit or update any contact.

```
64    # giving new details and editing contact
65
66            name = input("Enter new name: ")
67            phone_number = input("Enter new phone number: ")
68            email = input("Enter new email: ")
69            address_book.edit_contact(index, name, phone_number, email)
70
```

- Put the “else” function to ensure if the index is good to go.
- Using the “main()” function to call the main function.

```
71    # display updated contacts
72
73            address_book.display_contacts()
74        else:
75            print("Invalid index.")
76
77    # calling the main function
78
79    main()
```

GitHub Link:
https://github.com/haseebtehsin18/BanoQabil_2.0_Python_Course/blob/main/project%20no.2.py