HARSHADA KENE 2441015

DIGITAL DIARY/JOURNAL APPLICATION

Description

This C++ program implements a digital diary application that allows users to add, display, and manage their personal and travel diary entries.

Classes Used

1. Diary: The base class that represents a diary entry.
2. PersonalDiary: A derived class that inherits from Diary and represents a personal diary entry.
3.TravelDiary: A derived class that inherits from Diary and represents a travel diary entry.

Data Members

- Diary: date and entry
- PersonalDiary: feelings (in addition to the inherited members)
- TravelDiary: location (in addition to the inherited members)

Member Functions

- Diary: displayEntry() (virtual)
- PersonalDiary: displayEntry() (overrides the virtual function in Diary)
- TravelDiary: displayEntry() (overrides the virtual function in Diary)

OOP Concepts Used

1. Inheritance: PersonalDiary and TravelDiary inherit from Diary.
2. Polymorphism: The displayEntry() function is overridden in PersonalDiary and TravelDiary.
3. Encapsulation: Data members are private and accessed through public member functions.

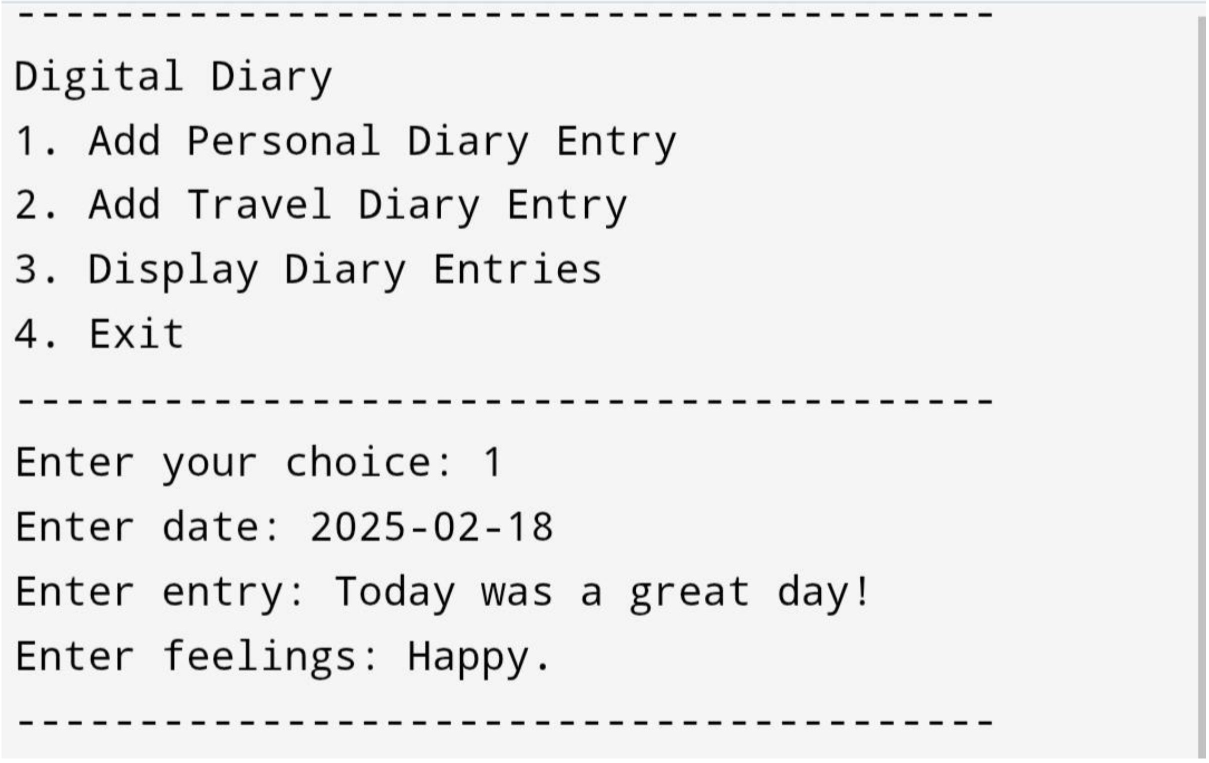
Program Flow

1. The program presents a menu to the user.
2. The user selects an option:
    - Add Personal Diary Entry
    - Add Travel Diary Entry
    - Display Diary Entries
    - Exit
3. Based on the user's selection, the program performs the corresponding action.

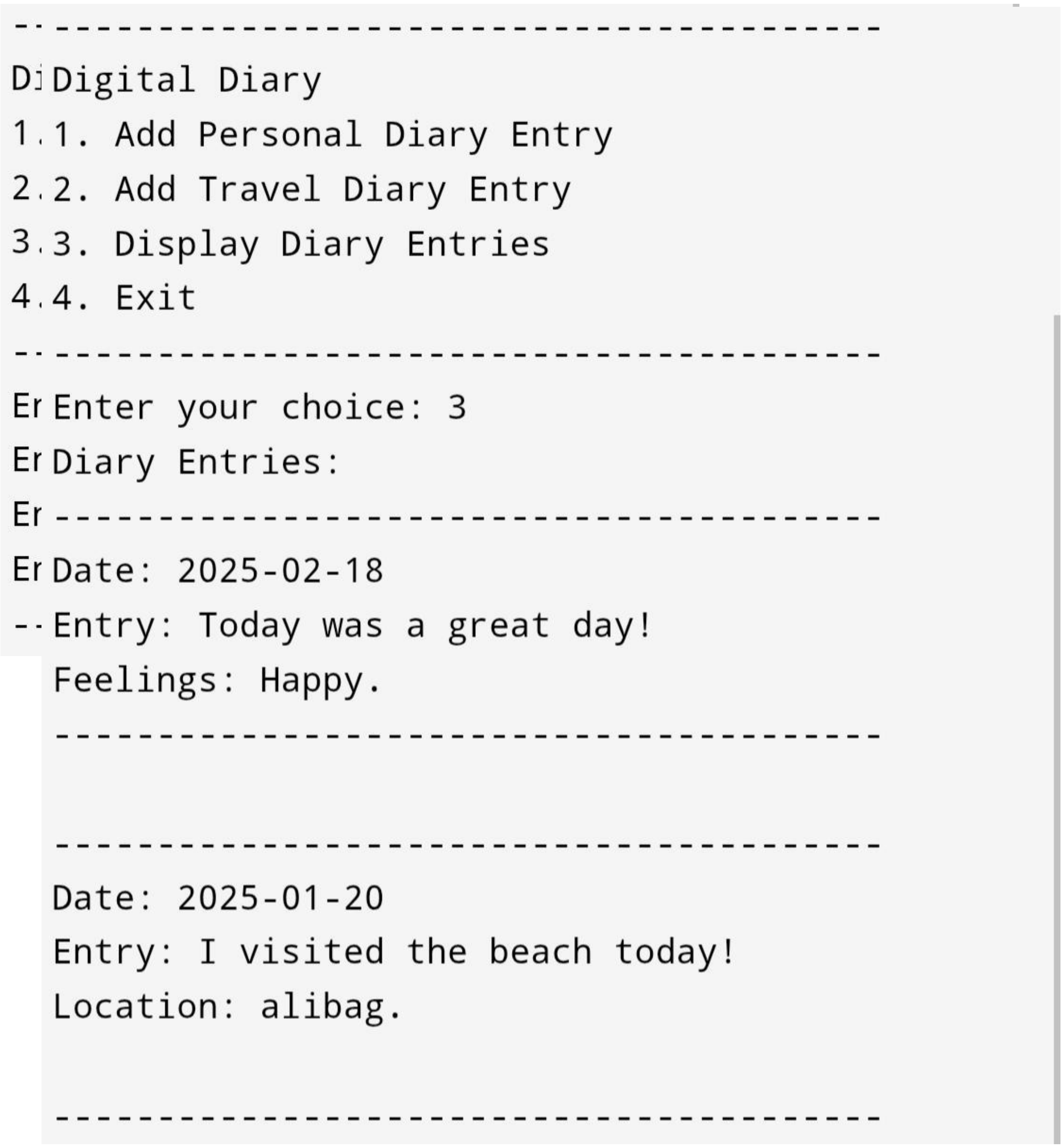
Code

```
#include <iostream>
#include <string>
using namespace std;
// Base class: Diary
class Diary {
protected:
    string date;
    string entry;
public:
    Diary(string date, string entry) : date(date),
entry(entry) {}
    virtual void displayEntry() const {
        cout<<"--------------------------------------"<<endl;
        cout << "Date: " << date << endl;
        cout << "Entry: " << entry << endl;
    }
};

// Derived class: PersonalDiary
class PersonalDiary : public Diary {
private:
    string feelings;
public:
    PersonalDiary(string date, string entry, string
feelings) : Diary(date, entry), feelings(feelings) {}
    void displayEntry() const override {
        Diary::displayEntry();
        cout << "Feelings: " << feelings << endl;
cout<<"----------------------------------------"<<endl;
    }
};
// Derived class: TravelDiary
class TravelDiary : public Diary {
private:
    string location;
public:
    TravelDiary(string date, string entry, string location) :
Diary(date, entry), location(location) {}
    void displayEntry() const override {
        Diary::displayEntry();
        cout << "Location: " << location << endl;
    }
};

int main() {
    int choice;
    string date;
    string entry;
    string feelings;
    string location;

    Diary* diary[10];
    int numEntries = 0;

    while (true) {
        cout<<"--------------------------------------"<<endl;
        cout << "Digital Diary" << endl;
        cout << "1. Add Personal Diary Entry" << endl;
        cout << "2. Add Travel Diary Entry" << endl;
        cout << "3. Display Diary Entries" << endl;
        cout << "4. Exit" << endl;
        cout<<"--------------------------------------"<<endl;
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1:
                cout << "Enter date: ";
                cin.ignore();
                getline(cin, date);
                cout << "Enter entry: ";
                getline(cin, entry);
                cout << "Enter feelings: ";
                getline(cin, feelings);
                diary[numEntries] = new PersonalDiary(date,
    entry, feelings);
                numEntries++;
                break;

            case 2:
                cout << "Enter date: ";
                cin.ignore();
                getline(cin, date);
                cout << "Enter entry: ";
                getline(cin, entry);
                cout << "Enter location: ";
                getline(cin, location);
                diary[numEntries] = new TravelDiary(date,
    entry, location);
                numEntries++;
                break;
             case 3:
                cout << "Diary Entries:" << endl;
                for (int i = 0; i < numEntries; i++) {
                   diary[i]->displayEntry();
                   cout << endl;
                }
                break;
             case 4:
                return 0;
                default:
            cout << "Invalid choice. Please try again." << endl;
              }
          }
        return 0;
    }
```

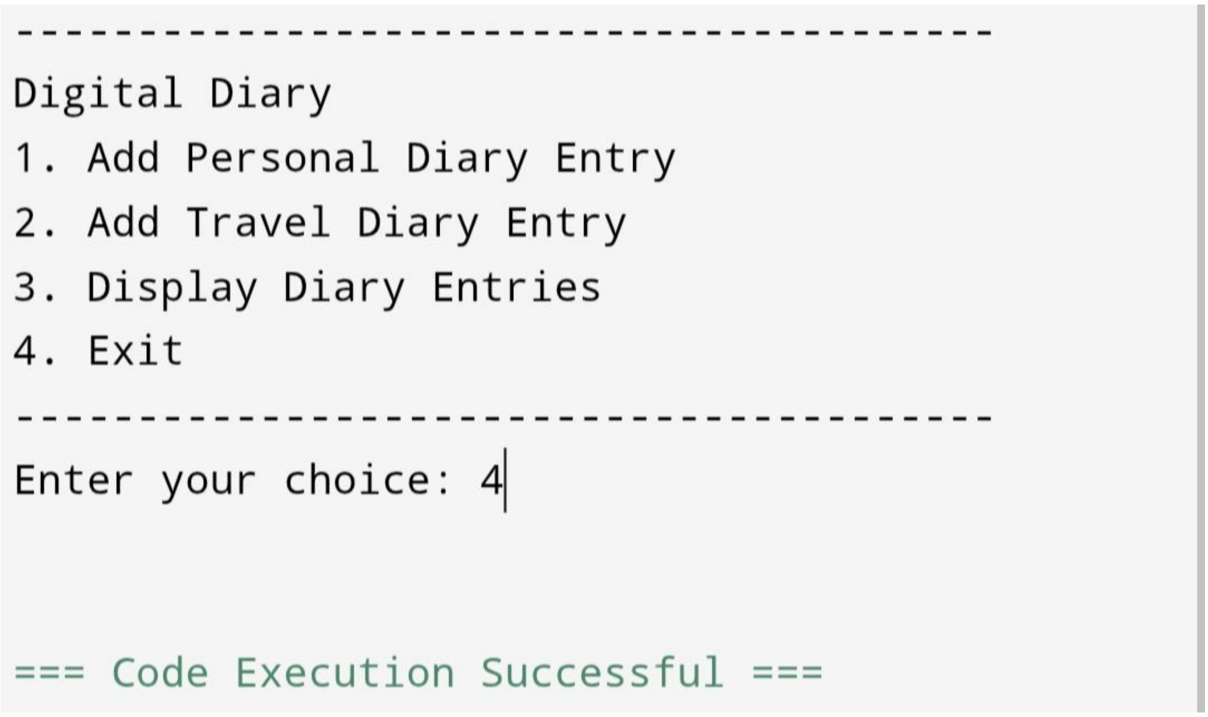
OUTPUT



```
----------------------------------------
Digital Diary
1. Add Personal Diary Entry
2. Add Travel Diary Entry
3. Display Diary Entries
4. Exit
----------------------------------------
Enter your choice: 1
Enter date: 2025-02-18
Enter entry: Today was a great day!
Enter feelings: Happy.
----------------------------------------
```



```
----------------------------------------
Digital Diary
1. Add Personal Diary Entry
2. Add Travel Diary Entry
3. Display Diary Entries
4. Exit
----------------------------------------
Enter your choice: 3
Diary Entries:
----------------------------------------
Date: 2025-02-18
Entry: Today was a great day!
Feelings: Happy.
----------------------------------------


----------------------------------------
Date: 2025-01-20
Entry: I visited the beach today!
Location: alibag.


----------------------------------------
```



```
----------------------------------------
Digital Diary
1. Add Personal Diary Entry
2. Add Travel Diary Entry
3. Display Diary Entries
4. Exit
----------------------------------------
Enter your choice: 4


=== Code Execution Successful ===
```

Conclusion

In conclusion, the digital diary application implemented in C++ demonstrates the effective use of object-oriented programming (OOP) concepts such as inheritance, polymorphism, and encapsulation. The program provides a user-friendly interface for managing personal and travel diary entries, showcasing the power and flexibility of C++ in building robust and efficient applications.

Through this project, we have gained hands-on experience with:

- Designing and implementing classes and objects

- Utilizing inheritance to create a hierarchy of classes
- Applying polymorphism to override functions and provide specific implementations
- Encapsulating data and behavior within classes

This project serves as a solid foundation for further exploration of C++ and its applications in software development.