Write C++ program using STL for sorting and searching user defined records such as personalrecords (Name, DOB, Telephone number etc) using vector container

Program-

```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>  // For sort and binary_search
#include <iterator>   // For distance

using namespace std;

// Define the structure for PersonalRecord
struct PersonalRecord {
    string name;
    string dob;      // Date of birth in the format YYYY-MM-DD
    string phone;    // Telephone number

    // Constructor to initialize a personal record
    PersonalRecord(string n, string d, string p) : name(n), dob(d), phone(p) {}

    // Overload the < operator for sorting by name
    bool operator<(const PersonalRecord& other) const {
        return name < other.name;
    }

    // Function to display a personal record
    void display() const {
        cout << "Name: " << name << ", DOB: " << dob << ", Phone: " << phone << endl;
    }
};

int main() {
    // Create a vector of PersonalRecord objects
    vector<PersonalRecord> records;

    // Add sample records
    records.push_back(PersonalRecord("Alice", "1990-05-12", "123-456-7890"));
    records.push_back(PersonalRecord("Bob", "1985-11-23", "234-567-8901"));
    records.push_back(PersonalRecord("Charlie", "1992-02-01", "345-678-9012"));
    records.push_back(PersonalRecord("Diana", "1988-08-15", "456-789-0123"));

    // Sort the records by name (uses the overloaded < operator)
    sort(records.begin(), records.end());
```

```
    // Display sorted records
    cout << "Sorted Records by Name:\n";
    for (const auto& record : records) {
        record.display();
    }

    // Searching for a record by name (using binary search)
    string searchName;
    cout << "\nEnter name to search for: ";
    getline(cin, searchName);

    // Create a dummy record to search for
    PersonalRecord searchRecord(searchName, "", "");

    // Use binary_search to find the record (binary search requires the container to be sorted)
    auto it = lower_bound(records.begin(), records.end(), searchRecord);

    if (it != records.end() && it->name == searchName) {
        cout << "\nRecord found: ";
        it->display();
    } else {
        cout << "\nRecord not found!" << endl;
    }

    return 0;
}
```