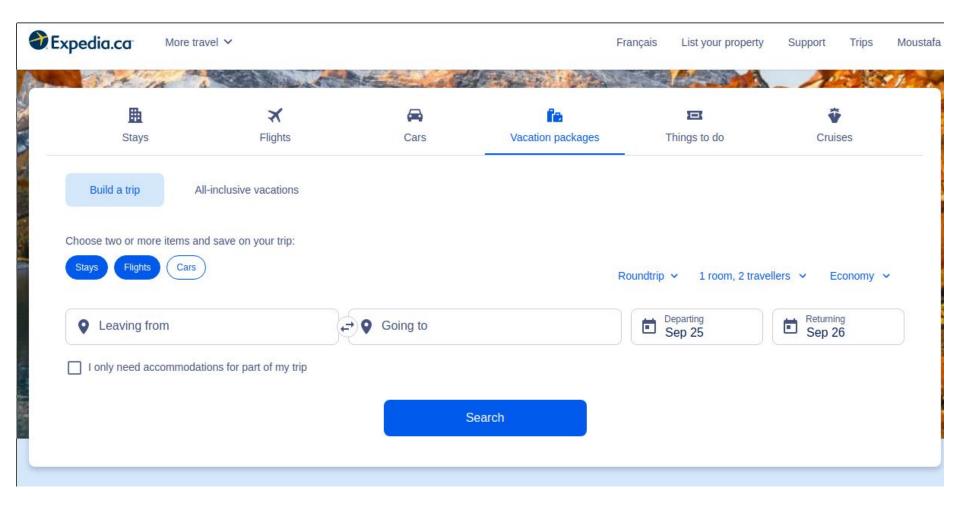# C++ Programming

Project: Expedia.com

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Stays   Flights   Cars   **Vacation packages**   Things to do   Cruises

Build a trip   All-inclusive vacations

Choose two or more items and save on your trip:

Stays   Flights   Cars

Roundtrip ⌄   1 room, 2 travellers ⌄   Economy ⌄

⚲ Leaving from   ⇄ ⚲ Going to   📅 Departing Sep 25   📅 Returning Sep 26

☐ I only need accommodations for part of my trip

Search

Functional Requirements

- In this project we will implement simple Expedia.com site
- In expedia a user have **several itineraries**, each itinerary consists of
  - 0 or more flights, hotels, cars, etc. E.g. 4 flights, 2 hotels and 2 cars.
  - For first MVP: Consider only flights (one way no transit) and hotels (but be extensible).
- An itinerary item (e.g. a flight) has:
  - Its own cost
  - Its info: E.g. A flight has request from user, and his selected flight details
    - E.g. Hotel cost: total nights x price per night
  - Itinerary cost is the **sum of all** internal items cost.
- For simplicity
  - Don't use files / Use dummy data when convenient
  - 2 types of users: Admin & customer. Focus of your code is on Customer part

Functional Requirements: User Interface

```
Menu:
        1: Login
        2: Sign Up

Enter number in range 1 - 2: 1
Enter user user_name & password: asmaa 222


Hello Asmaa Saad Ibrahim | User View

Menu:
        1: View Profile
        2: Make Itinerary
        3: List my itineraries
        4: Logout

Enter number in range 1 - 4: 2

Menu:
        1: Add Flight
        2: Add Hotel
        3: Done
        4: Cancel

Enter number in range 1 - 4:
```

Functional Requirements: User Interface

```
Enter number in range 1 - 4: 2

Menu:
        1: Add Flight
        2: Add Hotel
        3: Done
        4: Cancel

Enter number in range 1 - 4: 1
Enter From: Cairo
Enter From Date (dd-mm-yy): 10-05-20
Enter To: Vancouver
Enter To Date (dd-mm-yy): 13-06-20
Enter # of adults children (5-16) and infants: 3 2 1
Airline: Turksih Airlines Cost: 300 Departure Date 10-01-2022 Arrival date 10-02-2022
Airline: Turksih Airlines Cost: 320 Departure Date 12-01-2022 Arrival date 10-02-2022
Airline: AirCanada Airlines Cost: 200 Departure Date 25-01-2022 Arrival date 10-02-2022
Airline: AirCanada Airlines Cost: 250 Departure Date 29-01-2022 Arrival date 10-02-2022

Enter -1 to cancel or number in range 1 - 4: 3
```

Functional Requirements: User Interface

```
Menu:
        1: Add Flight
        2: Add Hotel
        3: Done
        4: Cancel

Enter number in range 1 - 4: 2
Enter From Date (dd-mm-yy): 10-05-20
Enter From Date (dd-mm-yy): 13-06-20
Enter Country: Canada
Enter City: Vancouver
Enter # of adults and children: 3 2
Hotel: Hilton Hotel Room Type: Interior View(6) Price per night: 200 From Date 29-01-2022 to 10-02-2022
Hotel: Hilton Hotel Room Type: City View(3) Price per night: 300 From Date 29-01-2022 to 10-02-2022
Hotel: Hilton Hotel Room Type: Deluxe View(8) Price per night: 500 From Date 29-01-2022 to 10-02-2022
Hotel: Marriott Hotel Room Type: City View(8) Price per night: 320 From Date 29-01-2022 to 10-02-2022
Hotel: Marriott Hotel Room Type: Interior View(8) Price per night: 220 From Date 29-01-2022 to 10-02-2022
Hotel: Marriott Hotel Room Type: Private View(5) Price per night: 600 From Date 29-01-2022 to 10-02-2022

Enter -1 to cancel or number in range 1 - 6: 4
```

Functional Requirements: User Interface

```
Menu:
        1: Add Flight
        2: Add Hotel
        3: Done
        4: Cancel

Enter number in range 1 - 4: 3

Select a payment choice:
        1: [Debit Card]: Owner: Asmaa Saad Ibrahim Card number: 11-22-33
        2: [Credit Card]: Owner: Asmaa Saad Ibrahim Card number: 22-11-0

Enter number in range 1 - 2: 1
Money withdraw successfully
Reservation confirmed
Itinerary reserved
```

Functional Requirements: User Interface

```
Menu:
        1: View Profile
        2: Make Itinerary
        3: List my itineraries
        4: Logout

Enter number in range 1 - 4: 3
**************************************************
Itinerary of 2 sub-reservations
Airline reservation with reservation: AirCanada Airlines: From Cairo on 10-05-20
        To Vancouver on 13-06-20
        Adults: 3 children: 2 infants: 1
        Total Cost:200

Hotel reservation: Marriott Hotel: Vancouver @ Canada
        10-05-20 to 13-06-20 : 3
        Adults: 3 children 2
        Total Cost:960

Total Itinerary cost: 1160
**************************************************
```

Functional Requirements: APIs

- Expedia doesn't know
    - What are details of flights, cars, hotels, etc. This is not in Expedia database (local files)
    - It call their APIs. As developers, we visit their sites to checkout their public APIs:
        - Checkout the functions available to call remotely
    - Also expedia wants to communicate with **ONE** of Payment APIs
        - For now, we consider 2: PayPal and Stripe
            - In future more could be added
- Assume for simplicity local APIs to call for flights and hotels as following
    - During the homework we already introduced some of that to the make project easier

Functional Requirements: APIs

```cpp
16 class AirCanadaCustomerInfo {
17 };
18
19 class AirCanadaFlight {
20 public:
21     double price;
22     string date_time_from;
23     string date_time_to;
24 };
25
26 class AirCanadaOnlineAPI {
27 public:
28     static vector<AirCanadaFlight> GetFlights(string from, string from_date,
29                 string to, string to_date, int adults, int childern) {
30         vector<AirCanadaFlight> flights;
31
32         flights.push_back( { 200, "25-01-2022", "10-02-2022" });
33         flights.push_back( { 250, "29-01-2022", "10-02-2022" });
34         return flights;
35     }
36     static bool ReserveFlight(const AirCanadaFlight& flight,
37             const AirCanadaCustomerInfo &info) {
38         return true;
39     }
40     static bool CancelReserveFlight(const AirCanadaFlight& flight,
41             const AirCanadaCustomerInfo &info) {
42         return true;
43     }
44 };
45
```

Functional Requirements: APIs

```cpp
46  class TurkishFlight {
47  public:
48      double cost;
49      string datetime_from;
50      string datetime_to;
51  };
52
53  class TurkishCustomerInfo {
54  };
55
56  class TurkishAirlinesOnlineAPI {
57  public:
58      void SetFromToInfo(string datetime_from, string from,
59              string datetime_to, string to) {
60      }
61      void SetPassengersInfo(int infants, int childern, int adults) {
63      vector<TurkishFlight> GetAvailableFlights() const {
64          vector<TurkishFlight> flights;
65
66          flights.push_back( { 300, "10-01-2022", "10-02-2022" });
67          flights.push_back( { 320, "12-01-2022", "10-02-2022" });
68          return flights;
69      }
70
71      static bool ReserveFlight(const TurkishCustomerInfo &info,
72              const TurkishFlight& flight) {
73          return false;
74      }
75      static bool CancelReserveFlight(const TurkishCustomerInfo &info,
76              const TurkishFlight& flight) {
77          return false;
78      }
79  };
80
```

Functional Requirements: APIs

```cpp
16  class HiltonRoom {
17  public:
18      string room_type;
19      int available;
20      double price_per_night;
21      string date_from;
22      string date_to;
23  };
24
25  class HiltonHotelAPI {
26  public:
27      static vector<HiltonRoom> SearchRooms(string country, string city, string from_date,
28              string to_date, int adults, int children, int needed_rooms) {
29          vector<HiltonRoom> rooms;
30          rooms.push_back( { "Interior View", 6, 200.0, "29-01-2022", "10-02-2022" });
31          rooms.push_back( { "City View", 3, 300.0, "29-01-2022", "10-02-2022" });
32          rooms.push_back( { "Deluxe View", 8, 500.0, "29-01-2022", "10-02-2022" });
33          return rooms;
34      }
35  };
```

Functional Requirements: APIs

```cpp
37 class MarriottFoundRoom {
38 public:
39     string room_type;
40     int available;
41     double price_per_night;
42     string date_from;
43     string date_to;
44 };
45
46 class MarriottHotelAPI {
47 public:
48     static vector<MarriottFoundRoom> FindRooms(string from_date, string to_date,
49             string country, string city, int needed_rooms, int adults, int children) {
50         vector<MarriottFoundRoom> rooms;
51         rooms.push_back( { "City View", 8, 320.0, "29-01-2022", "10-02-2022" });
52         rooms.push_back( { "Interior View", 8, 220.0, "29-01-2022", "10-02-2022" });
53         rooms.push_back( { "Private View", 5, 600.0, "29-01-2022", "10-02-2022" });
54         return rooms;
55     }
56 };
57
58 ///////////////////////////////////
```

Functional Requirements: APIs

```cpp
5⊖class PayPalCreditCard {
6  public:
7      string name;
8      string address;
9      string id;
10     string expire_date;
11     int ccv;
12 };
13
14⊖class PayPalOnlinePaymentAPI {
15 public:
16⊖     void SetCardInfo(const PayPalCreditCard* const card) {
17     }
18⊖     bool MakePayment(double money) {
19         return true;
20     }
21 };
```

- Developers investigated Paypal provided payment API
  - Seems we create object of the API
  - Set info through the PayPalCreditCard
  - Then call MakePayment
    - Bool to indicate sucess

Functional Requirements: APIs

```cpp
42  class StripeUserInfo {
43  public:
44      string name;
45      string address;
46  };
47
48  class StripeCardInfo {
49  public:
50      string id;
51      string expiry_date;
52  };
53
54  class StripePaymentAPI {
55  public:
56      bool static WithDrawMoney(StripeUserInfo user,
57              StripeCardInfo card, double money) {
58          return true;
59      }
60  };
61
62  class SquarePaymentAPI {
63  public:
64      bool static WithDrawMoney(string JsonQuery) {
65          json::JSON obj = JSON::Load(JsonQuery);
66          return true;
67      }
68  };
```

- The interface is simpler
  - The request info is represented using a **JSON** query
  - See example

```json
{
    "card_info" : {
        "CCV" : 333,
        "DATE" : "09-2021",
        "ID" : "11-22-33-44"
    },
    "money" : 20.500000,
    "user_info" : ["mostafa", "canada"]
}
```

Functional Requirements

- Think in needed business logic
    - E.g. let's say I created itinerary of 2 flights and 2 cars
    - I asked to pay
    - We confirmed first flight and first hotel, then failed in next flight
        - We should cancel the first flight and hotel
        - We should cancel payment
        - We should inform user
- Enhance the APIs with whatever missing logic
    - E.g. I did not put reserve/cancel reserve in hotels as in flights
- There is a lot of space for polymorphism and design choices
    - Think twice
    - Compare with my code on github...mine around 1500 lines of code

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."