****

**Advanced Object-Oriented Software Engineering**

**Group 13 – Phase 1**

**Online Tourism System**

|  |  |  |
| --- | --- | --- |
| ID | Student Name | SRS Group No. |
| 196735 | Mahmoud Assem Abdelkhalek | A3 |
| 195423 | Seifeldin Mohamed Khalil | A3 |
| 193754 | Sama Hany Hamed | A2 |
| 203398 | Maiada Khaled | A3 |
| 195948 | Abdelrahman Hagrass | A1 |

Contents

[Project Description 4](#_Toc119417287)

[Use Case Diagram 5](#_Toc119417288)

[Use Case Scenarios 6](#_Toc119417289)

[Maiada 203398: 6](#_Toc119417290)

[Create Package 6](#_Toc119417291)

[Book Package 6](#_Toc119417292)

[Add rooms 7](#_Toc119417293)

[Mahmoud Assem 196735: 8](#_Toc119417294)

[Book Flight 8](#_Toc119417295)

[Edit Flight Details 8](#_Toc119417296)

[Handle Complaint 9](#_Toc119417297)

[Seifeldin Khalil 195423: 10](#_Toc119417298)

[Rent Car 10](#_Toc119417299)

[Edit Car Details 10](#_Toc119417300)

[Make Payment 11](#_Toc119417301)

[Sama Hany 193754: 12](#_Toc119417302)

[Submit Complaint 12](#_Toc119417303)

[Add Car 12](#_Toc119417304)

[Edit Room Details 13](#_Toc119417305)

[Abdelrahman Hagrass 195948: 14](#_Toc119417306)

[Book Hotel 14](#_Toc119417307)

[Create review 14](#_Toc119417308)

[Add flight 15](#_Toc119417309)

[Package Diagram 16](#_Toc119417310)

[Quality Attribute Scenarios 17](#_Toc119417311)

[Maiada 203398: 17](#_Toc119417312)

[Fault Tolerance 17](#_Toc119417313)

[Reusability 18](#_Toc119417314)

[Modularity 19](#_Toc119417315)

[Mahmoud Assem 196735: 20](#_Toc119417316)

[Maintainability: *Modifiability*: Test functionality 20](#_Toc119417317)

[*Portability*: installability 20](#_Toc119417318)

[Compatibility: *Interoperability*: Declined 21](#_Toc119417319)

[Seifeldin Khalil 195423: 22](#_Toc119417320)

[Compatibility: Interoperability 22](#_Toc119417321)

[Maintainability: Modifiability 22](#_Toc119417322)

[Reliability: Recoverability 23](#_Toc119417323)

[Sama Hany 193754: 24](#_Toc119417324)

[Availability: Downtime 24](#_Toc119417325)

[Maintainability : Testability 24](#_Toc119417326)

[Security : Authentication: 25](#_Toc119417327)

[Abdelrahman Hagrass 195948: 26](#_Toc119417328)

[Book flights, hotels, or cars 26](#_Toc119417329)

[Entering wrong input, choosing wrong options 26](#_Toc119417330)

[Detailed class diagram 27](#_Toc119417331)

[Each pattern description 28](#_Toc119417332)

[Composite Design Pattern - Maiada 203398 28](#_Toc119417333)

[Chain of Responsibility Design Pattern - Seifeldin 195423 29](#_Toc119417334)

[Observer – Subscriber pattern – Mahmoud Assem 196735 30](#_Toc119417335)

[Data Mapper- Abdelrahman 195948 31](#_Toc119417336)

[Read Only- Abdelrahman 195948 32](#_Toc119417337)

# Project Description

The Online Tourism aims to simplify the process of planning a trip, whether the trip is for tourism or business. It helps the users by allowing them to book the most important parts of the trip, the flight and place of accommodation, all from one single application rather than having to traverse multiple websites or applications to do so. The application has several services that will help the user throughout their trip.

The application gives the user the privilege to book their flight tickets to their desired destination. The users can also book a hotel room to as an accommodation. The user can also pre-book a rental car for the duration of their trip so that they have an easy and instant method of transportation once they reach their destination. Moving on there are some packages that are available for the users to book which consist of a flight ticket, a hotel room, and a rental car all combined for a cheaper price than booking each of the components individually. The user can also subscribe to a service that would notify them when a new package has been added. Finally, the user will be able to leave a review regarding one or more of the components that they had booked, the review could either be a positive one or an inconvenience complaint which is then handled by one of the admins.

The admins are the staff that work on the application, they have a variety of privileges on the application. The first of these privileges is generating sales reports of the available services on the website, this helps them in creating packages by combining some of the most popular purchases of each component into one singular package at a discounted price for the user’s convenience. The admins also have to ability to remove these packages as well as handle any of the complaints that the users might have.

As for the Airline company agent, the hotel agent, and the car rental agent they all have similar privileges. Each of them is responsible for adding listings to the application based on their provided services, as well as editing and removing any of these listings. Lastly, all have to ability to view all their listings that are currently available to be booked or have already been booked.

# Use Case Diagram

# Use Case Scenarios

## Maiada 203398:

|  |  |
| --- | --- |
| **ID** | **00-pkg** |
| **Title** | Create Package |
| **Description** | The admin can create packages of (Flight ticket – Hotel stay – Car rental) for subscribes users to purchase with a discounted price |
| **Primary Actor** | Admin |
| **Secondary Actor** | NA |
| **Preconditions** | * Admin should be logged in * The package to be added must not be already added to the system before |
| **Postconditions** | * Subscribed users are notified of the created package and is listed |
| **Main Success Scenario** | 1. Admin enters the details of the flight ticket 2. Admin enters the details of hotel room 3. Admin enters the details of car rent 4. Admin reviews entered details then submits |
| **Extensions** | **4.a.** If upon submitting the admin wants to remove it, he can view list of packages added to system, select the desired one to remove, and delete it.  **4.b.** If upon submitting the admin wants to edit the details, he can view list of packages added to system, select the desired one to edit, and modify any of the fields, unless it is purchased by any user. |
| **Priority** | HIGH |

|  |  |
| --- | --- |
| **ID** | **01-pkg** |
| **Title** | Book Package |
| **Description** | The user can choose from the tour packages one to book |
| **Primary Actor** | User |
| **Secondary Actor** | NA |
| **Preconditions** | * User should be logged in * User should be a subscriber to travelling packages |
| **Postconditions** | * Purchased package is updated with the user who purchased it, and the number of purchases available yet for customers is decreased 1 |
| **Main Success Scenario** | 1. User views the list of packages 2. User chooses a certain package to purchase 3. Package details are displayed 4. User enters payment details and checkout |
| **Extensions** | **1.a.** If no packages are available at the moment, the user will get notified later upon adding any to the list.  **2.a.** If upon viewing the package details user changes his mind he can go back and choose another package from the list |
| **Priority** | MEDIUM |

|  |  |
| --- | --- |
| **ID** | **00-rooms** |
| **Title** | Add rooms |
| **Description** | The hotel agent can add details of rooms to be listed for users to book |
| **Primary Actor** | Hotel Agent |
| **Secondary Actor** | NA |
| **Preconditions** | * Hotel Agent should be logged in * The room to be added must not be already added to the system before |
| **Postconditions** | * The added room is listed for users to book * Room availability calendar is updated whenever a user book it |
| **Main Success Scenario** | 1. Hotel agent enters the details of the room 2. Hotel agent submits the new room to the hotel list |
| **Extensions** | **2.a.** If upon submitting, the hotel agent wants to remove it, he can view list of rooms added to his hotel, select the desired one to remove, and delete.  **2.b.** If upon submitting, the hotel agent wants to edit the details, he can view list of rooms added to his hotel, select the desired one to edit, and modify any of the fields, unless it is booked by any user. |
| **Priority** | HIGH |

## Mahmoud Assem 196735:

|  |  |
| --- | --- |
| **ID** | **USER-01** |
| **Title** | Book Flight |
| **Description** | The user can choose a flight to their travelling destination to book. |
| **Primary Actor** | User |
| **Secondary Actor** | NA |
| **Preconditions** | * User should be logged in |
| **Postconditions** | * The flight will have a seat reserved for the user and the flight’s available seats will decrease by 1. |
| **Main Success Scenario** | 1. The user will view the list of all available flights to their desired destination with all the details. 2. The user will choose the trip that suits them according to the date/ time and pricing. 3. The user will confirm their payment to reserve them a seat in the flight. |
| **Extensions** | **1.a** If there are now available flights now the user can return to the home menu.  **2.a** If the user enters wrong payment credentials, they will be asked to enter them again. |
| **Priority** | HIGH |

|  |  |
| --- | --- |
| **ID** | **FA-01** |
| **Title** | Edit Flight Details |
| **Description** | The Airline Agent can choose a flight to edit its details. |
| **Primary Actor** | Airline Agent |
| **Secondary Actor** | NA |
| **Preconditions** | * User Airline Agent be logged in * The selected flight should be upcoming yet. |
| **Postconditions** | * The flight’s details will be updated with the new details to the users. |
| **Main Success Scenario** | 1. The Airline Agent will view a list of all the flights that they can edit now. 2. The Airline Agent will select a flight to edits its details. 3. The Airline Agents will save the edited flight details to the system. 4. The new flight details will be available to the users to inspect. |
| **Extensions** | **N/A** |
| **Priority** | MEDIUM |

|  |  |
| --- | --- |
| **ID** | **ADMIN-01** |
| **Title** | Handle Complaint |
| **Description** | The admin can select an available complain submitted by the users to start handling it. |
| **Primary Actor** | Admin |
| **Secondary Actor** | NA |
| **Preconditions** | * User admin must be logged in * The complains list should not be empty. |
| **Postconditions** | * A message will be sent to the user who was complaining demonstrating how will the issue be handled. * The complaint will be set to handled and will be removed from the available complains list. |
| **Main Success Scenario** | 1. The admin will view a list with all available complains submitted by users. 2. The admin will select a certain complain to start handling. 3. The complaint will be removed from the list once handled. |
| **Extensions** | **N/A** |
| **Priority** | MEDIUM |

## Seifeldin Khalil 195423:

|  |  |
| --- | --- |
| **ID** | **C-1** |
| **Title** | Rent Car |
| **Description** | The customer can view the cars listed available for rent and rent the car he wishes. |
| **Primary Actor** | Customer |
| **Secondary Actor** | NA |
| **Preconditions** | * Customer should be logged in * The chosen car should be available. |
| **Postconditions** | * The chosen car is removed from available cars. * Confirmation message is sent to the customer |
| **Main Success Scenario** | 1. The user selects the car he wishes to rent. 2. The car is available for rent, and the user is redirected to the payment screen. |
| **Extensions** | **A)** If the car is not available for rent, the user is asked to select a different car. |
| **Priority** | HIGH |

|  |  |
| --- | --- |
| **ID** | **C-2** |
| **Title** | Edit Car Details |
| **Description** | The agent can choose to edit the details of a listed car. |
| **Primary Actor** | Car Rental Agent |
| **Secondary Actor** | NA |
| **Preconditions** | * The agent should be logged in * The agent should select a listing to edit. |
| **Postconditions** | * The listing’s details are updated. |
| **Main Success Scenario** | 1. The agent views their listings. 2. The agent selects the listing they wish to update. 3. The agent enters the new details. 4. The agent confirms the update, and the new details are displayed on the listing. |
| **Extensions** | **NA** |
| **Priority** | LOW |

|  |  |
| --- | --- |
| **ID** | **C-3** |
| **Title** | Make Payment |
| **Description** | After the customer selects the listing they wish to book/rent from any of the 3 fields (hotels, flights, cars) he is redirected to the payment. |
| **Primary Actor** | Customer |
| **Secondary Actor** | Bank |
| **Preconditions** | * The customer should be logged in * The customer should select a listing he wishes to purchase. |
| **Postconditions** | * Confirmation is sent to the customer’s email. * Confirmation is sent to agent of the selected listing. |
| **Main Success Scenario** | 1. The customer views available listings. 2. The customer selects the listing they wish to purchase. 3. The customer enters the payment details. |
| **Extensions** | 1. **If the payment info is invalid, the customer will be asked to re-enter the details.** |
| **Priority** | High |

## Sama Hany 193754:

|  |  |
| --- | --- |
| **ID** | **S-01** |
| **Title** | Submit Complaint |
| **Description** | When a customer faces an issue, he can submit complaint to be viewed and examined by the admin. |
| **Primary Actor** | Customer |
| **Secondary Actor** | Admin |
| **Preconditions** | * Customer is logged in. * Customer has a booking in progress, or has booked a service before. |
| **Postconditions** | * Complaint is sent to admin for examining, and handling the complaint. |
| **Main Success Scenario** | 1. Customer fills a complaint. 2. The system accepts the complaint. 3. System forwards the complaint to the admin for investigation. |
| **Extensions** | N/A |
| **Priority** | Medium |

|  |  |
| --- | --- |
| **ID** | **S-02** |
| **Title** | Add Car |
| **Description** | The car rental agent is able to add new cars to the system that are newly available for rental and this is added to the listed cars for the user. |
| **Primary Actor** | Car Rental Agent |
| **Secondary Actor** | N/A |
| **Preconditions** | * Car rental agent is logged in. |
| **Postconditions** | * Listed cars for the user is updated with the newly added cars. |
| **Main Success Scenario** | 1. Car agent adds the car details. 2. The list of the available cars for rental is updated with the newly added car. |
| **Extensions** | N/A |
| **Priority** | High |

|  |  |
| --- | --- |
| **ID** | **S-03** |
| **Title** | Edit Room Details |
| **Description** | The hotel is able to update changes that may apply to the rooms, as the availability of the room, the price, number of beds, AC condition, etc. |
| **Primary Actor** | Hotel Agent |
| **Secondary Actor** | N/A |
| **Preconditions** | * Hotel Agent is logged in. * The room the agent is editing should be already added to the system. |
| **Postconditions** | * The room changes are applied to the list of the rooms in the hotel. |
| **Main Success Scenario** | 1. Hotel agent views the list of the rooms. 2. The agent chooses the room he wants to edit. 3. Edits are changed and updated in the list of rooms. |
| **Extensions** | N/A |
| **Priority** | High |

## Abdelrahman Hagrass 195948:

|  |  |
| --- | --- |
| **ID** | **A-01** |
| **Title** | Book Hotel |
| **Description** | A customer has the ability to book a hotel's rooms. |
| **Primary Actor** | Customer |
| **Secondary Actor** | NA |
| **Preconditions** | * Customer should be logged in. * Customer to be booked a hotel should has available rooms. * Check if Customer enters number of rooms. |
| **Postconditions** | * Alert with successful message. * Book details add to booking history. |
| **Main Success Scenario** | 1. View all hotels. 2. Choose hotels from hotel list. 3. Customer enters rooms number. |
| **Extensions** | **1a.** If the customer is not registered, create an account.  **2a.** If the hotel full, show a popup to the user that hotel is full.  **3a.** If the number of rooms exceed available hotels, show popup with max number of rooms. |
| **Priority** | HIGH |

|  |  |
| --- | --- |
| **ID** | **A-02** |
| **Title** | Create review |
| **Description** | The Customer can rate and review any kind of booking. |
| **Primary Actor** | Customer |
| **Secondary Actor** | NA |
| **Preconditions** | * Customer should be logged in. * Customer should have booking of type (flights, hotels, and renting car). * Reviewed booking should status “done”. |
| **Postconditions** | * Update the (flight, hotel, car) rating and comments * Alert with submitted successfully. |
| **Main Success Scenario** | 1. View customer’s booking history. 2. Open booking details. 3. Fill-up review form. 4. Submit the form. |
| **Extensions** | **1a.** If the customer is not registered, create an account.  **2a.** If the customer an empty form, show a popup with error message.  **3a.** if there is no rating reason, show a popup says that “enter your reason.”. |
| **Priority** | Low |

|  |  |
| --- | --- |
| **ID** | **A-03** |
| **Title** | Add flight |
| **Description** | The Airline Agent has ability to add new flight and add all new details for flights and seats. |
| **Primary Actor** | Airline Company Agent |
| **Secondary Actor** | NA |
| **Preconditions** | * Airline Agent should be logged in. * Check if there is no duplicate flights. |
| **Postconditions** | * Update the view list with new flight and available seats. * Alert that new flight added to the list. * Check if the seated available more than 0. |
| **Main Success Scenario** | 1. Open Airline agent UI. 2. Select Add flight. 3. Fill-up add flight form with seated number. 4. Submit the form. |
| **Extensions** | **1a.** If the Airline agent is not registered, create an account.  **2a.** If the Airline agent an empty form, show a popup with error message.  **3a.** if there are no available seats, refuse to add new flight. |
| **Priority** | High |

# Package Diagram

# Quality Attribute Scenarios

## Maiada 203398:

|  |  |  |
| --- | --- | --- |
| Scenario ID | 01rel-FT | |
| Scenario Description | Physical infrastructure causes an internal system failure to the servers in runtime, thus creates an emergency operational mode, in which the server replica takes on the responsibility to process booking transactions during the repair time of main server, of 24 hours. | |
| Scenario Goal | System’s ability to continue operating if exposed to a crash for a specific period of time. | |
| Quality Attribute | Reliability | |
| Attribute Concern | Fault Tolerance | |
| Scenario Refinement | Stimulus | Internal system failure |
| Stimulus Source | Physical Infrastructure damage |
| Environment | During runtime with emergency operational mode |
| Artifact | Processors / Servers |
| Response | * Switch to replica server * Repair main server |
| Response Measure | Time to switch servers must not exceed 10 seconds (repair rate = 99%) |

|  |  |  |
| --- | --- | --- |
| Scenario ID | 02maint-ruse | |
| Scenario Description | During the build time of system components, the developer is to structure the system so that its booking component can be chosen from a previously built booking component, for cars, hotels and flights bookings; to reduce effort and time. | |
| Scenario Goal | Structure the system’s components to use ready-made components. | |
| Quality Attribute | Maintainability | |
| Attribute Concern | Reusability | |
| Scenario Refinement | Stimulus | Reuse an external component for booking in our system |
| Stimulus Source | Developer |
| Environment | During build time |
| Artifact | Component |
| Response | * Booking component is implemented in our system successfully |
| Response Measure | Our system should be using the booking component without introducing defects. |

|  |  |  |
| --- | --- | --- |
| Scenario ID | 02maint-mod | |
| Scenario Description | During the design phase of our system, the developer is to divide the system into conceptual pieces (separated module for each of hotel, cars, & flights with common booking module), to lower coupling and increase cohesion; in order to ease future modifications and extensibility of the system. | |
| Scenario Goal | Modularization will ease the system’s modification without causing defects / degrading to existing product’s quality. | |
| Quality Attribute | Maintainability | |
| Attribute Concern | Modularity | |
| Scenario Refinement | Stimulus | System design of the 3 main booking services of our system (hotels, cars & flights) |
| Stimulus Source | Developer |
| Environment | Design time |
| Artifact | Our Online Tourism System |
| Response | Modular designed software of large functional independence, where each booking service resides on different module with a shared module for booking transactions, to ease extensibility and modifiability. |
| Response Measure | * High cohesion * Low coupling |

## Mahmoud Assem 196735:

### Maintainability: *Modifiability*: Test functionality

|  |  |
| --- | --- |
| Source | Software Developer |
| Stimulus | Developer is asked to test a certain functionality operates. |
| Artifact | Online Tourism System |
| Environment | Testing phase |
| Response | Functionality is tested to make sure it is working properly. |
| Response Measure | * Cost stays within the budget. * Takes no longer than 2 days. |

### *Portability*: installability

|  |  |
| --- | --- |
| Source | Client |
| Stimulus | Developers are asked to deploy the system on several operating systems. |
| Artifact | Online Tourism System |
| Environment | Production phase |
| Response | System should operate efficiently on all operating systems. |
| Response Measure | * System should have the same speed on all operating systems. * System should have the same design in all operating systems. |

### Compatibility: *Interoperability*: Declined

|  |  |
| --- | --- |
| Source | Online tourism system |
| Stimulus | Systems requests user data to be checked from the external banking system |
| Artifact | Online tourism system, external banking system |
| Environment | Systems known prior to run time |
| Response | Banking system checks user payment details to see if it is Declined |
| Response Measure | * Information is entered correctly 100% of the time |

## Seifeldin Khalil 195423:

### Compatibility: Interoperability

|  |  |
| --- | --- |
| Source | Online tourism system |
| Stimulus | Systems requests user data to be checked from the external banking system |
| Artifact | Online tourism system, external banking system |
| Environment | Systems known prior to run time |
| Response | Banking system checks user payment details to see if it is accepted |
| Response Measure | Information is entered correctly 100% of the time |

### Maintainability: Modifiability

|  |  |
| --- | --- |
| Source | Developer |
| Stimulus | Wishes to make the UI more responsive and user friendly |
| Artifact | Code |
| Environment | Design time |
| Response | The change is made and tested to make sure it works |
| Response Measure | In less than 5 hours |

### Reliability: Recoverability

|  |  |
| --- | --- |
| Source | Server |
| Stimulus | Server crash |
| Artifact | Online tourism System database |
| Environment | Repair mode |
| Response | Recover last created backup of data stored on the server |
| Response Measure | Repair time should not exceed 1 hour |

## Sama Hany 193754:

### Availability: Downtime

|  |  |
| --- | --- |
| Source | Heartbeat Monitor |
| Stimulus | Server Down |
| Artifact | Online tourism system |
| Environment | Normal operation |
| Response | Notify developers and appropriate entities |
| Response Measure | System downtime should not exceed 3 minutes |

### Maintainability : Testability

|  |  |
| --- | --- |
| Source | Tester |
| Stimulus | Unit tester applies unit tests on the system components and functions |
| Artifact | Component in the Online tourism system |
| Environment | In the testing phase |
| Response | Random tests are applied to each component and output is compared to what is expected |
| Response Measure | The component should pass the unit tests applied, with maximum 1% fault rate |

### Security : Authentication:

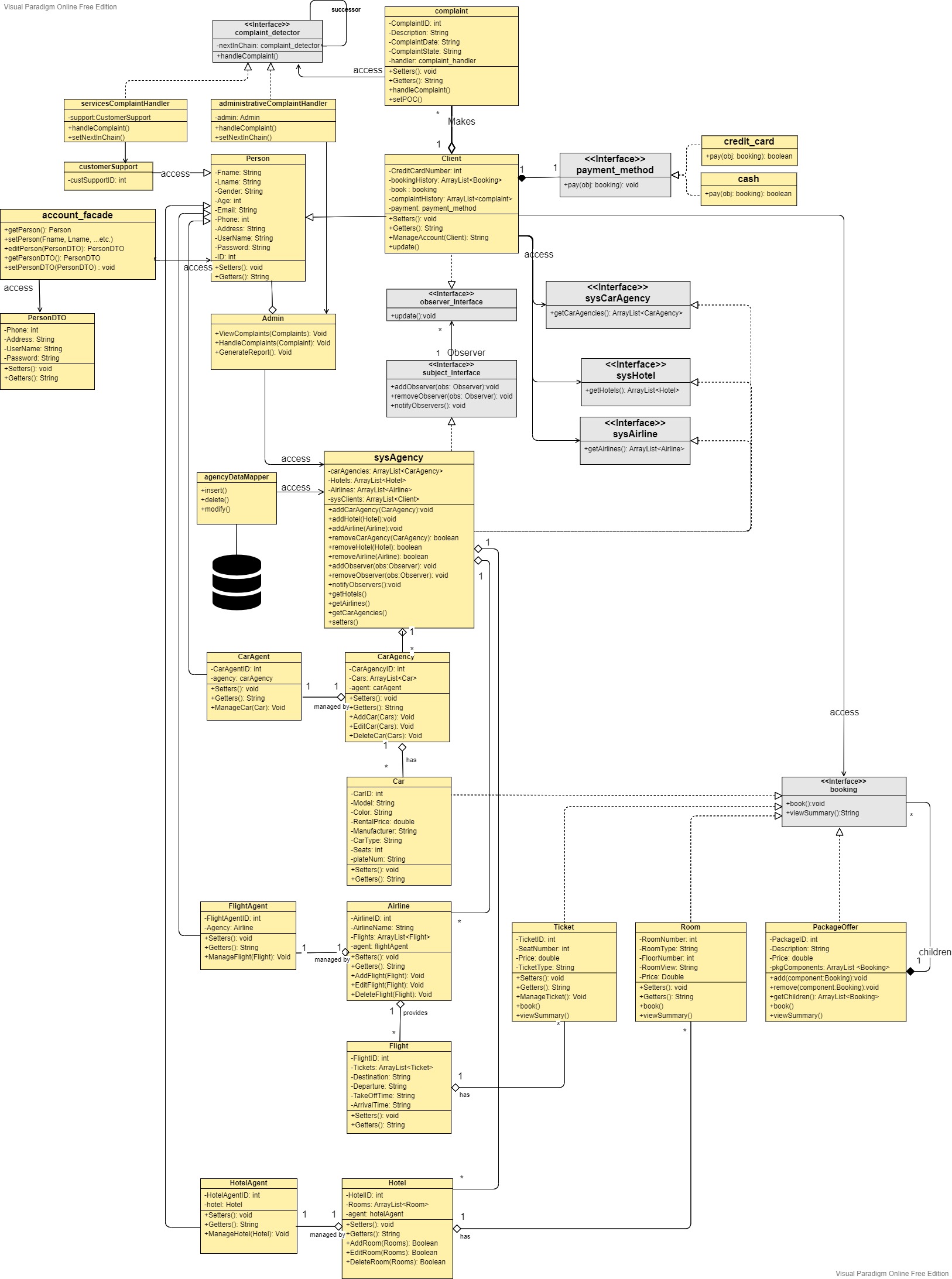
|  |  |
| --- | --- |
| Source | User |
| Stimulus | User tries to access system services |
| Artifact | Data in the Online tourism system |
| Environment | Normal operations |
| Response | Authenticates the user, if he’s not the user, services are blocked and hide any personal information until user is authenticated. |
| Response Measure | User is expected to get all the personal data correctly entered in less than 2 trials only |

## Abdelrahman Hagrass 195948:

|  |  |
| --- | --- |
| **Source of stimulus** | User |
| **Stimulus** | Book flights, hotels, or cars |
| **Artifact** | System |
| **Environment** | Normal operation |
| **Response** | Booking is processed and send a confirmation message. |
| **Response measure** | Average acceptance is 1 min |

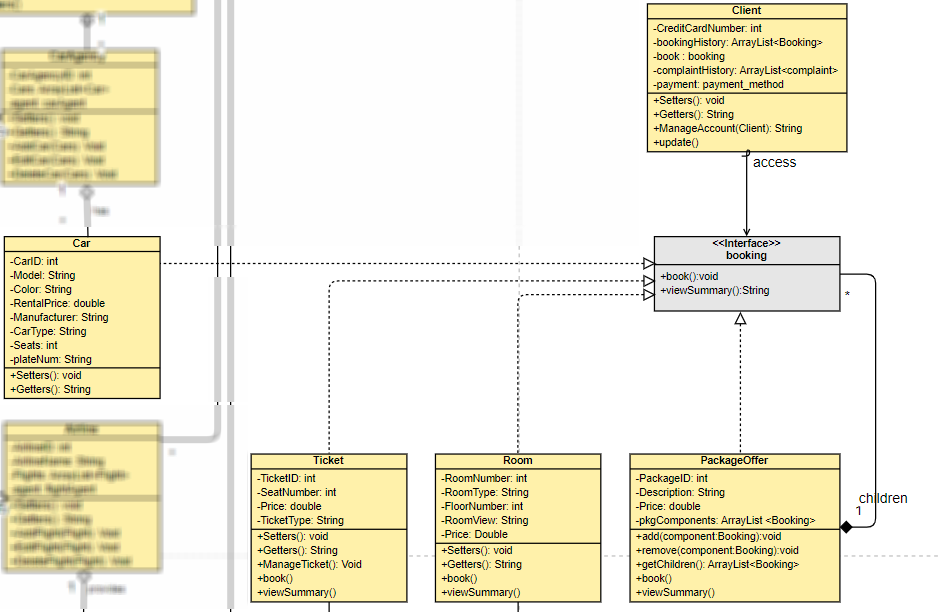
|  |  |
| --- | --- |
| **Source of stimulus** | User |
| **Stimulus** | Entering wrong input, choosing wrong options |
| **Artifact** | System |
| **Environment** | Normal operation |
| **Response** | Alert user with right input or option |
| **Response measure** | * Alerts must be in all system. * Alert must appear within 2 sec. |

# Detailed class diagram

please check the class diagram image from our submitted folder for clearer view.

## Each pattern description

### Composite Design Pattern - Maiada 203398



Composite pattern is handy in our case, where clients might want to book more than one service, to make use of the discount available (-20% of the total bookings invoice).

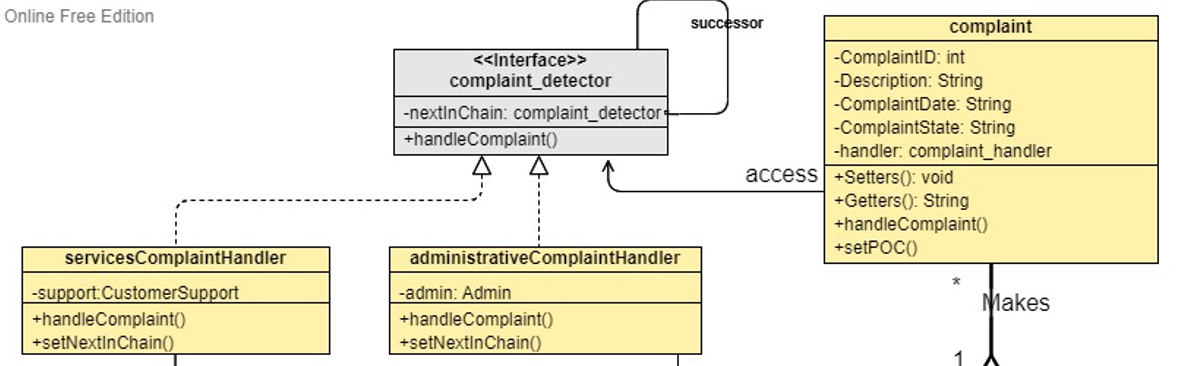
Thus, an interface named “booking” is created as the **component interface class**, for as many services (aka: leaves) to implement from it in future.

Also, a **composite class** named “packageOffer” is created to hold in the chosen leaves in an array named pkgComponents.

The above-mentioned **leaf classes** are represented in our system as “Car”, flight “ticket”, or “room” classes.

Packages can sometimes include other packages as well besides the separate leaves, for example, a client can book a package of room and flight ticket and have another package of room and car plus any separate service all in one booking.

### Chain of Responsibility Design Pattern - Seifeldin 195423



Our systems use the Chain of Responsibility design pattern for handling user complaints. Since our system has different types of complaints.

When the customer files a complaint, he fills the data required in the **complaint** **class**. The **complaint\_detector interface** passes the complaint firstly to the **adminstrativeComplaintHandler class**. This class has an object of type admin who is responsible for handling any problems that the customer might have faced with one of the many agencies available in our application. If the complaint falls under this category it is handled in the **adminstrativeComplaintHandler class,** if not is passed to the next handler class.

The other handler class is the **serviceComplaintHandler.** If the complaint could not be handled by the previous handle, it is passed to this handler. It has an object of type CustomerSupport which is responsible for handling customer complaints that could be caused by some sort of technical issue the customer is having with the application, login errors, etc.

### Observer – Subscriber pattern – Mahmoud Assem 196735

Diagram

Description automatically generated

The observer – subscriber pattern works in our system whenever a new agency of any type is added to the system. Users are notified each time a new agency is added to the system. Class sysAgency is a class that carries all the information about all types of agencies operating within the system. The class also contains all functions/operations regarding the agencies’ functionalities.

The subscription process takes place once the user has signed up to the system. Therefore, there is no actual subscription process that the user performs. All users within the system are added to the observer list, and each new user registers to the system are added to the observer list. The admins are responsible for adding new agencies to the system.

Since that class sysAgency carries the information about all agencies through the system. So, whenever a particular type of agency is added to the system. The subscriber interface uses the subject interface to loop over all users within the system to notify them that a new agency was added to the system.

### Data Mapper- Abdelrahman 195948

Diagram

Description automatically generated

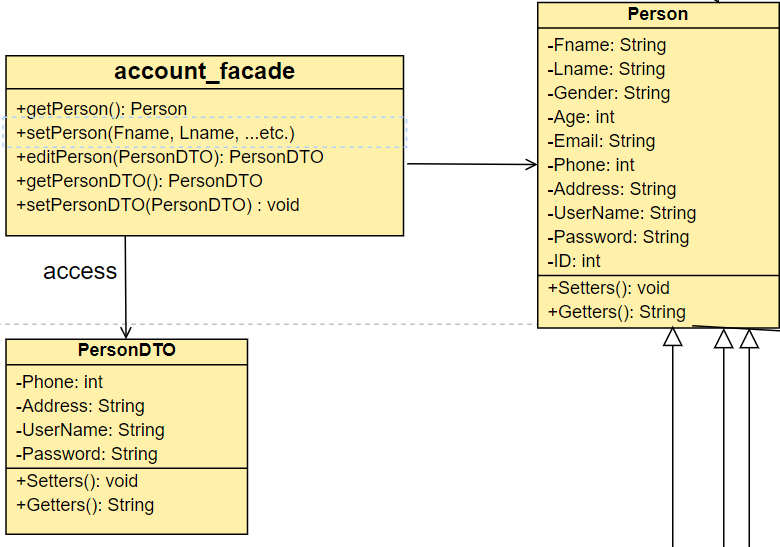
SysAgencies is a class that collects our agencies (hotels, airlines, and carAgencies), which is an important class on our system that needs to be separated from the data layer (database). We need a faster system because database queries take some time to execute. We applied the Data Mapper in our system to this important class. There is a class named agenciesDataMapper that contains functions for our database (insert, delete, and modify). And this class can access SysAgencies in order to process its data for the database.

### Read Only- Abdelrahman 195948

Diagram

Description automatically generated

Admins and clients both have access to and can read sysAgencies (hotels, airlines, and car rental agencies). Admin, on the other hand, has access to the entire class and can add, edit, and update agencies it, so we provide a read-only pattern that allows the client to only get and read from this class for booking purpose. To implement this pattern in our system, we create three interface classes; each of them has a function that reads from class SysAgency and gets its data to the client. Clients can use these three interfaces to view and obtain information about each class.

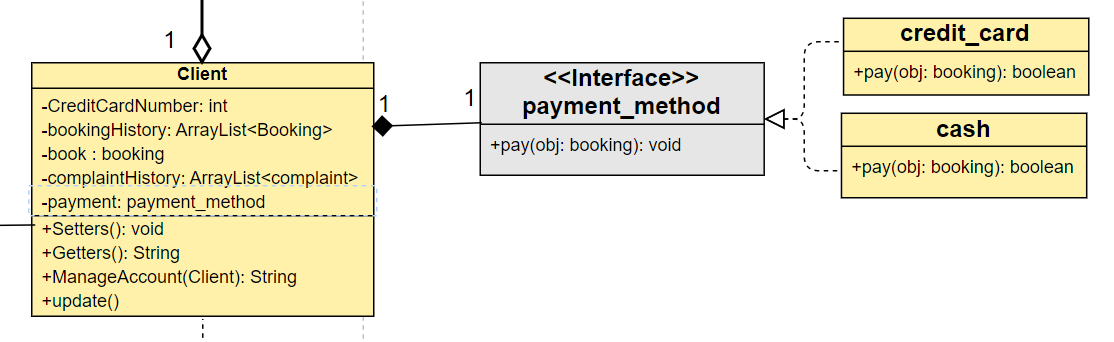
Data Transfer Object Patter (DTO) – Sama193754:

In our system a DTO pattern should be applied for the data to be secure and to enhance the system performance. The intent of the DTO is to create object that carries some data of a class certain class to reduce the method calls and parameters.

In our application, a person after signing up some data needs to be secure of changing as the person’s first and last name, gender, email and ID. Also there are a lot of changeable data, as a result, the DTO pattern is applied through the class **PersonDTO**, which carries the changeable data only of the class Person. And instead of calling only the changeable attributes from person, an object of the **PersonDTO** is passed to the parameters, more secure instead of accessing all the attributes of Person, setters and getters take the object in their parameters.

In the **account\_facade**, the functions **getPersonDTO** and **setPersonDTO** take an object of the **PersonDTO** in their parameters, and the **getPersonDTO** returns the **PersonDTO**.

Strategy Pattern – Sama193754:



The strategy pattern is a behavioral pattern that defines a family of different strategies and put each of them in a separate class, and make each object interchangeable.

In our system, the client have different payment methods according to what he chooses.

The **Client** is given two payment options; **credit\_card** or **cash.** According to what the client chooses the class relative to the choice is called and implemented, each of them has different body of the function **pay().** **Credit\_card** and **cash** class both implement the interface class **payment\_method.**