

|  |
| --- |
| **FPT UNIVERSITY** |
| Capstone Project Document  Report 3  Project Management Plan |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | **Group 18** | | | **Group Members** | Đào Bảo Long – Team Leader – SE60690  Lê Phúc Lữ – Team Member – 60563  Nguyễn Thanh Tùng – Team Member – SE60609  Nguyễn Tấn Đức – Team Member – SE60660 | | **Supervisor** | Nguyễn Trọng Tài | | **Ext Supervisor** | N/A | | **Capstone Project Code** | i-Deliver |   Build a Web Application  for manages all activities of  delivery service system by coach. | |
|  |
|  |
| Hồ Chí Minh City, January 2014 |

|  |
| --- |
| *This page is intentionally left blank* |

*Acknowledgement*

After more than three months of intensive researching, working, and having fun, we have finally reached the fundamental objectives of the project *Multi-Hotel Booking*. Apart from our own efforts, the completion of this project depends largely on the encouragement and guidelines of many others. We would like to dedicate this page to expressing our gratitude to the people who have been instrumental in our project.

Firstly and most importantly, we would like to show our greatest respect to Mr. Lâm Hữu Khánh Phương, our supervisor, whose tremendous support and surprising yet innovative ideas contributed a great deal to the completion of our project. Without his encouragement and guidance, this project would probably not have materialized. More than that, we really appreciate the fact that he put us under no pressure whatsoever and always let us having fun while we were working on this project.

Secondly, we would like to thank our fellow men in Group 11, who were also working on a similar project. We will never forget the moments when we were so drawn into arguments that led to nowhere that we even thought we would ended up failing to finish our projects. It was an extraordinarily interesting experience having worked together.

Finally, we would like to express our deepest gratitude to our parents and our lecturers for always being there for us. We appreciate that they did a lot to gear us up for all the things we might encounter in our lives. We would not be who we are today if it weren’t for them.

Group 2  
*Nguyễn Phú An,  
Bùi Minh Đức,  
Nguyễn Hiếu Triệu Vỹ*

*This page is intentionally left blank*

# Table of figures

[Figure 1. Model-View-Controller Architecture 6](#_Toc380566451)

[Figure 2. Home page 41](#_Toc380566452)

[Figure 3. Search Hotel page 42](#_Toc380566453)

[Figure 4. Hotel Details page 43](#_Toc380566454)

[Figure 5. Booking Details page 44](#_Toc380566455)

[Figure 6. Dashboard page 45](#_Toc380566456)

[Figure 7. Approve Booking page 47](#_Toc380566457)

[Diagram 1. Component Diagram 7](#_Toc380566645)

[Diagram 2. State machine diagram: Request status transition 28](#_Toc380566646)

[Diagram 3. State machine diagram: Package status transition 28](#_Toc380566647)

[Diagram 4. Sequence diagram: Search Routes 29](#_Toc380566648)

[Diagram 5. Sequence diagram: Post delivery requests 30](#_Toc380566649)

[Diagram 6. Sequence diagram: Create a trip 31](#_Toc380566650)

[Diagram 7. Sequence diagram: Edit a station 32](#_Toc380566651)

[Diagram 8. Sequence diagram: Add a price 33](#_Sequence_Diagram:_Add)

[Diagram 9. Sequence diagram: Create a Staff account 34](#_Sequence_Diagram:_Create)

[Diagram 10. Sequence diagram: Approve a request 35](#_Sequence_Diagram:_Approve)

[Diagram 11. Sequence diagram: Assign package 36](#_Sequence_Diagram:_Assign)

[Diagram 16. Logical database design 49](#_Toc380566660)

[Diagram 17. Physical database design 56](#_Toc380566661)

# Software Design Description (SDD)



## Design Overview

The SDD describes the system architectural design and the detailed designs, including the user interfaces, of the system.

The components should communicate through interfaces. The detailed implementation of each component should be transparent to other components. The passive MVC III pattern is the preferable architecture for the website.

The detailed designs of the system should adopt the basic principles of software design, including “high cohesion, low coupling” and “open to extension, closed to modification” principles.

Each of the following sections is summarized below:

* Section 4.2: Gives a specification of the system architecture design, describing the overall architecture of the system and subsystems.
* Section 4.3: Specifies all the components that should be included in the system, and the communications between them.
* Section 4.4: Describes the class and package diagrams, as well as their descriptions in details, to provide a static view of the system.
* Section 4.4: Describes the sequence diagrams for the use cases specified in the SRS, to provide a dynamic view of the system.
* Section 4.5: Describes the user interfaces, to specify what will be seen by the users of the system.
* Section 4.5: Describes the Database Design, including the relations and the relationships between them.

## System Architectural Design

The MVC III (Model – View – Controller) pattern is used as the overall system architecture, because it specifies a clear distinction between the responsibilities of the components and is appropriate for developing web applications.

3. Exchange data

4. Results

**CONTROLLER**

* Intercepts user input;
* Coordinates the view and model;
* Handles communication between the model and data layer.

**VIEW**

* Binds to the model;
* Renders the UIs (HTML, CSS, JavaScript);
* Allows navigating between controllers.

**MODEL**

* Communicates with data source;
* Exposes functionalities, business logics, and data validation.

Database   
Server

Application Server

1. Request

8. Response

5. Select views

7. Changes made

2. Invoke methods

6. Query states

Method invocations

Change notifications

Figure 1.Model-View-Controller Architecture

## Component Diagram

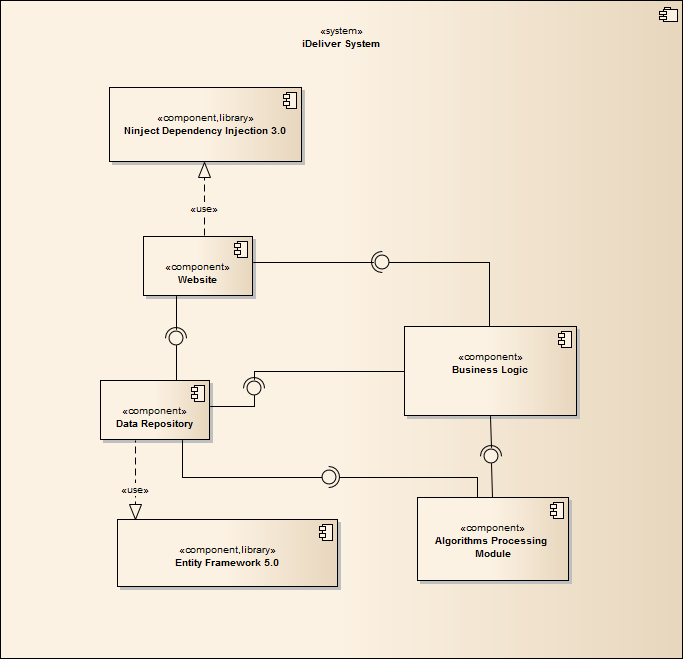


Diagram 1 Component Diagram

The MHB system includes the following major components:

* **Entity Framework 5.0:** An object-relational mapper that enables working with relational data persistence using domain-specific objects;
* **Data Repository:** A data store that provides data access functionality, using the Entity Framework to communicates with the database;
* **Data Synchronization Module:** A module that communicates with the services provided by the BMT hotel data provider and uses interfaces exposed by the Data Repository to persist the data;
* **BMT Services:** A set of web services that provide information about hotels, room types, and price changes for its consumers;
* **Business Logic:** A module that uses interfaces exposed by the Data Repository to encapsulate important businesses of in the MHB system;
* **Ninject Dependency Injection 3.0:** A lightweight dependency injection framework for .NET applications which helps split the application into a collection of loosely-coupled, highly-cohesive pieces, and then glue them back together in a flexible manner.
* **Website:** A web-application that helps exposes all the functionalities to end-users.

From the point of view of the MVC-II pattern, the components Entity Framework 5.0, Data Repository, Data Synchronization Module, BMT Services, Business Logic, and Ninject Dependency Injection 3.0 all belong to the Model part. The Controller and the View parts are wrapped in the component Website.

## Detailed Designs



### Entity Classes

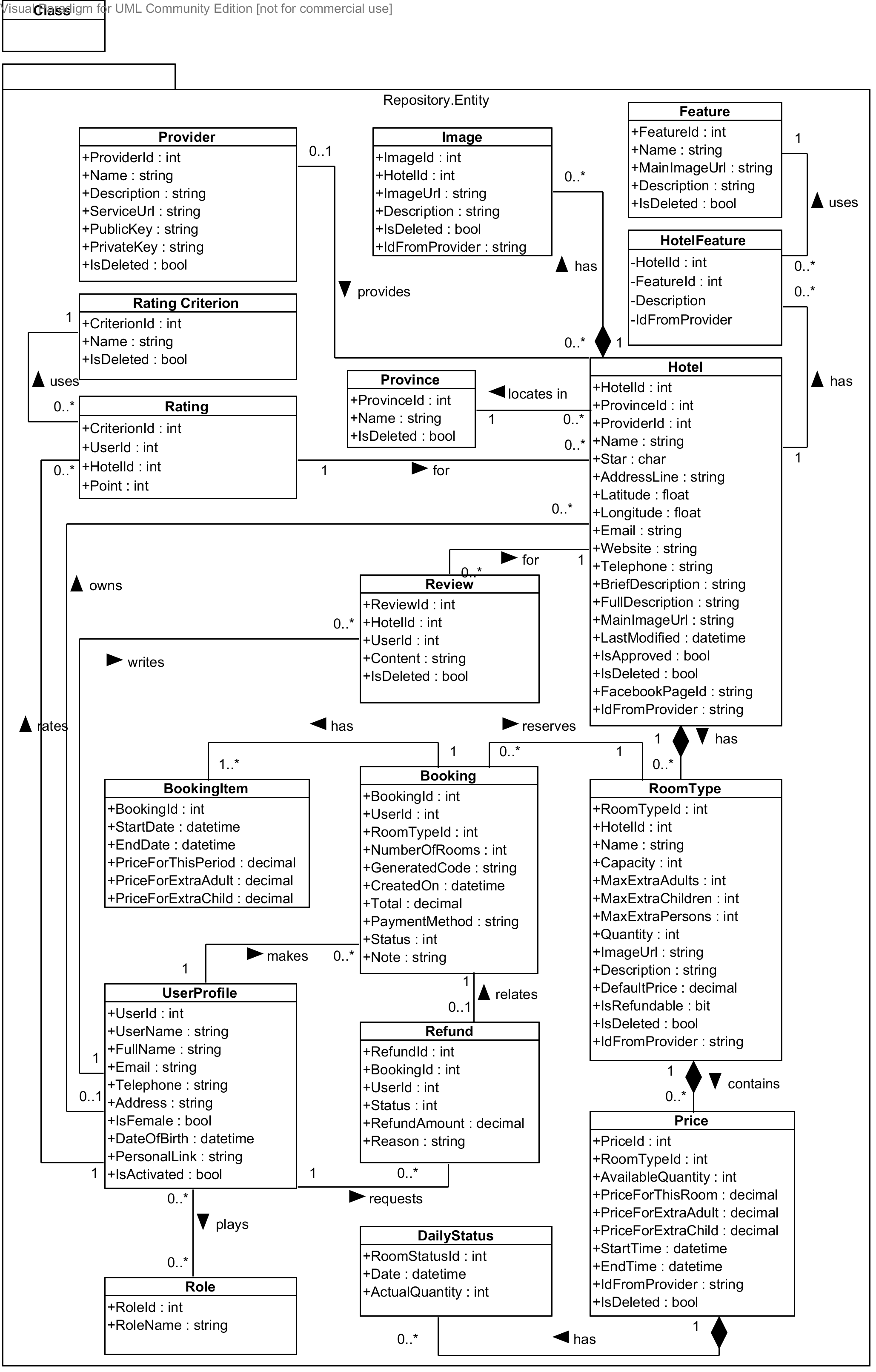


Diagram 1 Entity class diagram

#### Hotel Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *HotelId* | int | public | Unique id that differentiates a hotel from the others |
| *ProvinceId* | int | public | Id of the province where the hotel locates in |
| *ProviderId* | int | public | Id of the hotel data provider who provides this hotel |
| *OwnerId* | int | public | Id of the hotel owner who manages this hotel |
| *Name* | string | public | Official name of the hotel |
| *Star* | char | public | International quality of the hotel |
| *AddressLine* | string | public | Address line of the hotel |
| *Latitude* | float | public | Latitude of the hotel’s address line |
| *Longitude* | float | public | Longitude of the hotel’s address line |
| *Email* | string | public | Main email address of the hotel |
| *Website* | string | public | Website address of the hotel |
| *Telephone* | string | public | Main telephone number of the hotel |
| *BriefDescription* | string | public | Short introduction about the hotel |
| *FullDescription* | string | public | Full description of the hotel |
| *MainImageUrl* | string | public | Url of the main image for the hotel |
| *LastModified* | datetime | public | Last time when the hotel’s information was modified |
| *IsApproved* | bool | public | Whether the hotel is approved or not |
| *IsDeleted* | bool | public | Whether the hotel is marked deleted or not |
| *FacebookPageId* | string | public | Id of the Facebook page that has the hotel installed as a tab |
| *IdFromProvider* | string | public | Id of the hotel if it is provided by a hotel data provider |

**Operations:** N/A

#### Feature Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *FeatureId* | int | public | Unique id that differentiates a feature from the others |
| *Name* | string | public | Name of the hotel feature |
| *MainImageUrl* | string | public | Image url of the hotel feature |
| *Description* | string | public | Short description of the hotel feature |
| *IsDeleted* | bool | public | Whether the hotel feature is deleted or not |

**Operations:** N/A

#### Image Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *ImageId* | int | public | Unique id that differentiates an image from the others |
| *HotelId* | int | public | Id of the hotel which the image belongs to |
| *ImageUrl* | string | public | Url of the image |
| *Description* | string | public | Description of what the image is about |
| *IdFromProvider* | string | public | Id of the image if it is provided by a hotel data provider |

**Operations:** N/A

#### Provider Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *ProviderId* | int | public | Unique id that differentiates a provider from the others |
| *Name* | string | public | Official name of the hotel data provider |
| *Description* | string | public | Description of the services of the provider |
| *ServiceUrl* | string | public | Base url of the services provided by the hotel provider |
| *PublicKey* | string | public | Public key used to identify hotel consumer when information exchanges occur |
| *PrivateKey* | string | public | Private key that only the hotel provider and its consumer know |
| *IsDeleted* | bool | public | Where the hotel provider is deleted or not |

**Operations:** N/A

#### Province Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *ProvinceId* | int | public | Unique id that differentiates a province from the others |
| *Name* | string | public | Name of the province in 64 provinces in Vietnam |
| *IsDeleted* | bool | public | Where the hotel province is deleted or not |

**Operations:** N/A

#### RoomType Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *RoomTypeId* | int | public | Unique id that differentiates a room type from the others |
| *HotelId* | int | public | Id of the hotel which the room type belongs to |
| *Name* | string | public | Official name of the room type specified by the hotel owner |
| *Capacity* | int | public | Default value of the number of persons who can stay in the room |
| *MaxExtraAdults* | int | public | Number of maximum extra children, if any |
| *MaxExtraChildren* | int | public | Number of maximum extra adults, if any |
| *MaxExtraPersons* | int | public | Number of maximum extra person, if any |
| *Quantity* | int | public | Number of physical rooms of this type in the hotel |
| *ImageUrl* | string | public | Main url of the image for the room type |
| *Description* | string | public | Description of the room type |
| *DefaultPrice* | decimal | public | Default price for the room type |
| *IsRefundable* | bool | public | Whether the rooms of this type can be refunded or not |
| *IsDeleted* | bool | public | Whethere the room type is deleted or not |
| *IdFromProvider* | string | public | Id of the room type if it is provided by a hotel provider |

**Operations:** N/A

#### Price Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *PriceId* | int | public | Unique id that differentiates a price from the others |
| *RoomTypeId* | int | public | Id of the room type that this price belongs to |
| *AvailableQuantity* | int | public | Number of available rooms for this room type in the period specified |
| *PriceForThisRoom* | decimal | public | Price for this room in the period specified |
| *PriceForExtraAdult* | decimal | public | Price for an extra adult, if any |
| *PriceForExtraChild* | decimal | public | Price for an extra child, if any |
| *StartTime* | datetime | public | Date on which the price becomes active |
| *EndTime* | datetime | public | Date on which the price becomes inactive |
| *IdFromProvider* | string | public | Id of the price if it is provided by a hotel provider |
| *IsDeleted* | bool | public | Whethere the price is deleted or not |

**Operations:** N/A

#### DailyStatus Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *PriceId* | int | public | Id of the price that this daily status belongs to |
| *Date* | datetime | public | Date on which this daily status takes place |
| *ActualQuantity* | int | public | Actual available number of rooms on the day when the daily status takes place |

**Operations:** N/A

#### UserProfile Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *UserId* | int | public | Unique id that differentiates a user from the others |
| *Username* | string | public | Unique name that will be use to log in the system |
| *FullName* | string | public | Full name of the user |
| *Email* | string | public | Email address of the user |
| *Telephone* | string | public | Telephone number of the user |
| *Address* | string | public | Address line of the user |
| *IsFemale* | bool | public | Whether the user is female or male |
| *DateOfBirth* | datetime | public | Date of birth of the user |
| *PersonalLink* | string | public | Personal website of the user |
| *IsActivated* | bool | public | Whether the user is activated on the MHB system or not |

**Operations:** N/A

#### Booking Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *BookingId* | int | public | Unique id that differentiates a hotel from the others |
| *UserId* | int | public | Id of the user who makes the booking |
| *RoomTypeId* | int | public | Id of the room type booked by the user |
| *NumberOfRooms* | int | public | Number of rooms the user booked for this room type |
| *ExtraAdults* | int | public | Number of extra adults, if any |
| *ExtraChildren* | int | public | Number of extra children, if any |
| *GeneratedCode* | string | public | String that whill be used to verify with the hotel when the user checks in |
| *CreatedOn* | datetime | public | Time when the booking is created |
| *Total* | decimal | public | Total amount of the booking order |
| *PaymentMethod* | string | public | Method which the user uses to pay for the booking |
| *Status* | int | public | Status of the booking |
| *Note* | string | public | Additional note the user specifies for the hotel |

**Operations:** N/A

#### Refund Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *RefundId* | int | public | Unique id that differentiates a refund from the others |
| *BookingId* | int | public | Id of the booking which the refunds relates |
| *UserId* | int | public | Id of the user who requests the refund |
| *Status* | int | public | Status of the refund request |
| *RefundAmount* | decimal | public | Amount that will be refunded to the user |
| *Reason* | string | public | Reason why the user wants the hotel to refund the bookings |

**Operations:** N/A

#### Role Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *RoleId* | int | public | Unique id that differentiates a role from the others |
| *RoleName* | string | public | Name of the role |

**Operations:** N/A

#### BookingItem Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *BookingId* | int | public | The id of the booking which this booking item belongs to |
| *StartDate* | datetime | public | Date on which a booking item starts |
| *EndDate* | datetime | public | Date on which a booking item ends |
| *PriceForThisPeriod* | decimal | public | Price for the room in the period specified |
| *PriceForExtraAdult* | decimal | public | Price for an extra adult in the period specified, if any |
| *PriceForExtraChild* | decimal | public | Price for an extra child in the period specified, if any |

**Operations:** N/A

#### Review Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *ReviewId* | int | public | Unique id that differentiates a review from the others |
| *HotelId* | int | public | Id of the hotel that the review is about |
| *UserId* | int | public | Id of the user who writes the review |
| *Content* | string | public | Content of the review written by the user |
| *IsDeleted* | bool | public | Whether the review is deleted or not |

**Operations:** N/A

#### Rating Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *CriterionId* | int | public | Id of the criterion being rated by the user |
| *UserId* | int | public | Id of the user who rates the hotel |
| *HotelId* | int | public | Id of the hotel being rated by the user |
| *Point* | int | public | Point that the user gives the hotel for the selected criterion |

**Operations:** N/A

#### RatingCriterion Class

**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *CriterionId* | int | public | Unique id that differentiates a criterion from the others |
| *Name* | int | public | Name of the criterion used to rate the hotels |
| *IsDeleted* | int | public | Whether the criterion is deleted or not |

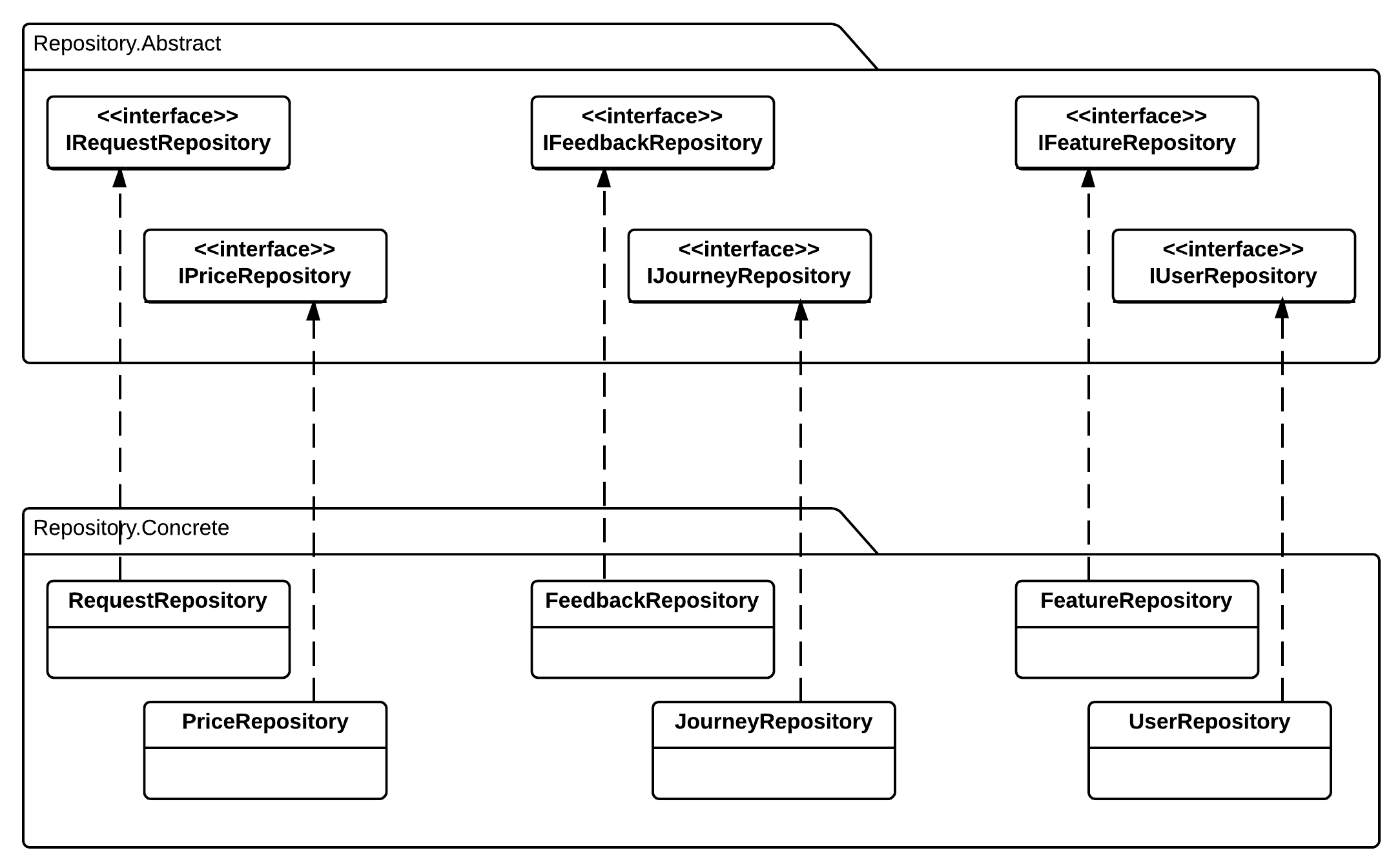
**Operations:** N/A



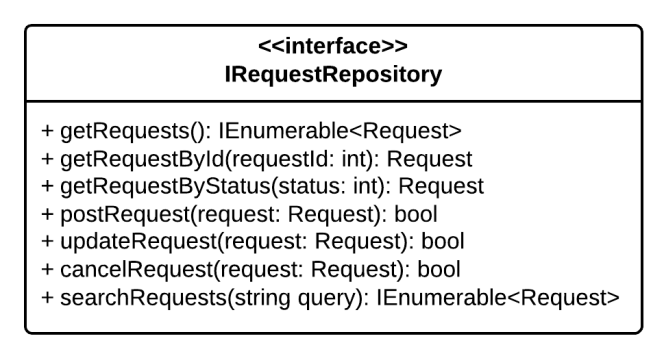
### Repository Classes



#### Class diagram



#### IRequestRepository Interface



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Request> Get Requests()  Description: Retrieves all the requests from the database | | |
| N/A | N/A | N/A |

|  |  |  |
| --- | --- | --- |
| Signature: Request GetRequestById(int requestId)  Description: Retrieves a request from the database given its id | | |
| Parameter | **Type** | **Description** |
| *requestId* | int | Id of the request to be retrieved |

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Request> GetRequestByStatus(int status)  Description: Retrieves all the request from the database given its status | | |
| Parameter | **Type** | **Description** |
| *Status* | Int | Status of the request to be retrived |

|  |  |  |
| --- | --- | --- |
| Signature: bool postRequest(Request request)  Description: post new delivery request to system and save to database | | |
| Parameter | **Type** | **Description** |
| *request* | Request | New request to be saved |

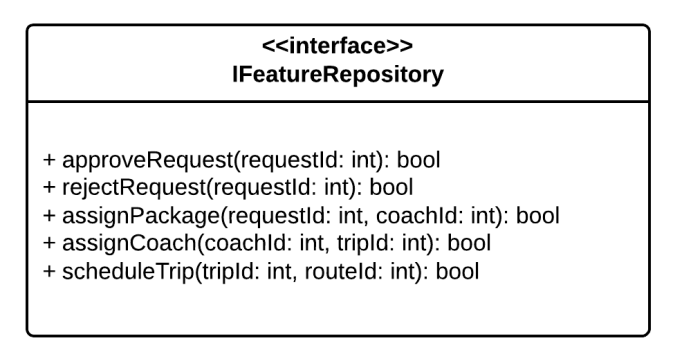
|  |  |  |
| --- | --- | --- |
| Signature: bool updateRequest(int requestId)  Description: Update existing request to the database given its id | | |
| Parameter | **Type** | **Description** |
| *requestId* | int | Id of the request to be updated |

|  |  |  |
| --- | --- | --- |
| Signature: bool cancelRequest(int requestId)  Description: Cancel/Remove existing request from the database given its id | | |
| Parameter | **Type** | **Description** |
| *requestId* | int | Id of the request to be updated |

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Request> serchRequest(string query)  Description: Retrieves all the request from the database given by query string | | |
| Parameter | **Type** | **Description** |
| *query* | String | Query string to find matching request |

|  |  |  |
| --- | --- | --- |
| Signature:  Description: | | |
| Parameter | **Type** | **Description** |
| *provinceId* | int | Id of the hotel to be checked |

#### IFeatureRepository Interface



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: bool approveRequest(int requestId)  Description: Approve pending requests | | |
| Parameter | **Type** | **Description** |
| *requestId* | int | Id of the request to be approved |

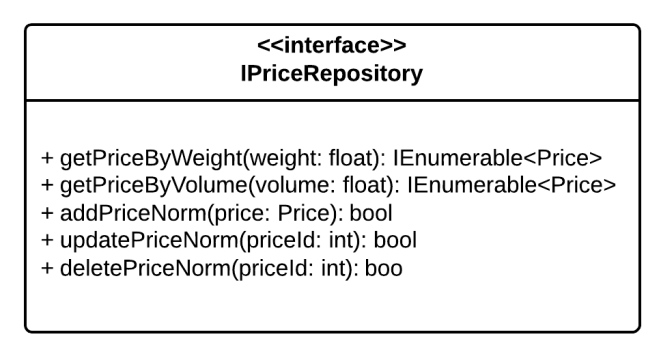
|  |  |  |
| --- | --- | --- |
| Signature: bool rejectRequest(int requestId)  Description: Reject pending requests | | |
| Parameter | **Type** | **Description** |
| *requestId* | int | Id of the request to be rejected |

|  |  |  |
| --- | --- | --- |
| Signature: bool assignPackage(int requestId, int coachId)  Description: Assign approved package for specified coach | | |
| Parameter | **Type** | **Description** |
| *requestId* | int | Id of the package to be assigned |
| *coachId* | int | Id of the coach to assign package |

|  |  |  |
| --- | --- | --- |
| Signature: bool assignCoach(int coachId, int tripId)  Description: Assign specific coach to for specified trip | | |
| Parameter | **Type** | **Description** |
| *coachId* | int | Id of the coach to be assigned |
| *tripId* | int | Id of the trip to assign coach |

|  |  |  |
| --- | --- | --- |
| Signature: bool scheduleTrip(int tripId, int routeId)  Description: Schedule trips for specified route | | |
| Parameter | **Type** | **Description** |
| *tripId* | int | Id of the trip to be schedule |
| *routeId* | int | Id of the route for trip scheduling |

#### IPriceRepository Interface



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Price> GetPriceByWeight(float weight)  Description: Get delivery price/fee based on package weight | | |
| Parameter | **Type** | **Description** |
| *weight* | float | Weight of delivered package |

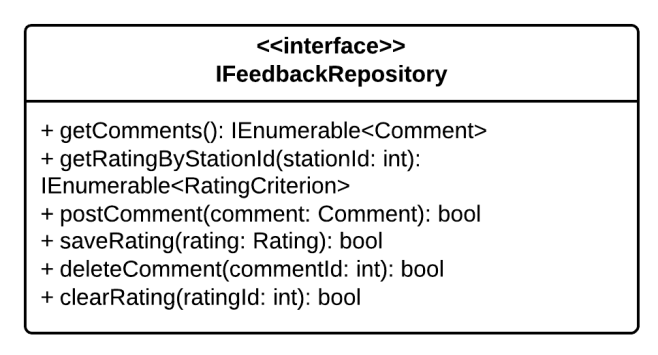
|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Price> GetPriceByVolume(float volume)  Description: Get delivery price/fee based on package volume | | |
| Parameter | **Type** | **Description** |
| *volume* | float | Volume of delivered package |

|  |  |  |
| --- | --- | --- |
| Signature: bool addPriceNorm(Price price)  Description: Add new price/fee norm to database | | |
| Parameter | **Type** | **Description** |
| *price* | Price | New price norm to be added to database |

|  |  |  |
| --- | --- | --- |
| Signature: bool updatePriceNorm(priceId: int)  Description: Update a price norm to the database | | |
| Parameter | **Type** | **Description** |
| *priceId* | int | The price norm to be saved |

|  |  |  |
| --- | --- | --- |
| Signature: bool updatePriceNorm(priceId: int)  Description: Delete/Remove a price norm from the database | | |
| Parameter | **Type** | **Description** |
| *priceId* | int | The price norm to be deleted |

#### IFeedbackRepository Interface



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Comment> GetComments()  Description: Retrieves all the comments from the database | | |
| Parameter | **Type** | **Description** |
| N/A | N/A | N/A |

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<RatingCriterion> GetRatingByStationId(int stationId)  Description: Retrieves all the rating that belong to a given station from the database | | |
| Parameter | **Type** | **Description** |
| *stationId* | int | Id of the station that has the rating to be retrieved |

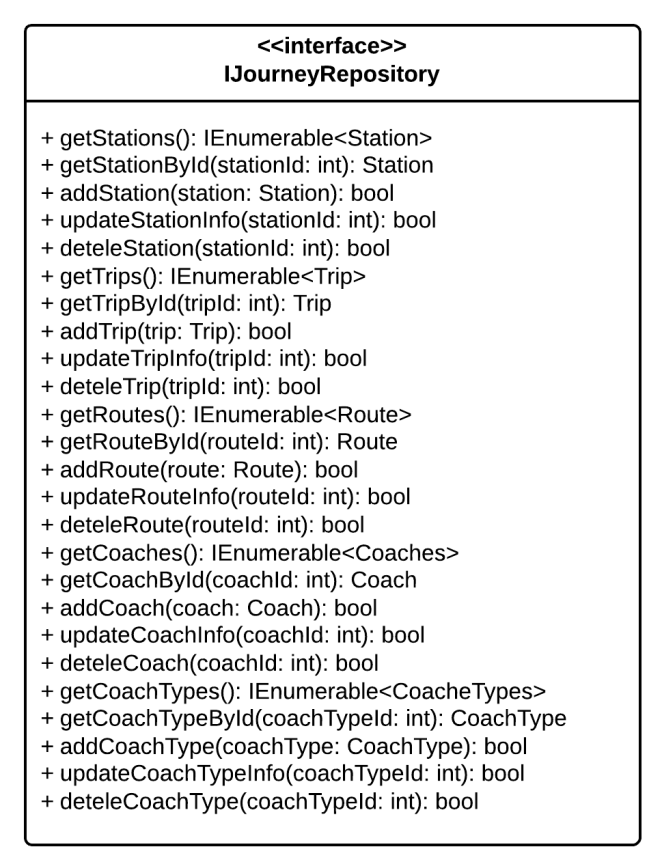
|  |  |  |
| --- | --- | --- |
| Signature: bool postComment(Comment comment)  Description: Post new comment and save to database | | |
| Parameter | **Type** | **Description** |
| *comment* | Comment | The new comment to be saved to database |

|  |  |  |
| --- | --- | --- |
| Signature: bool saveRating(Rating rating)  Description: Save station rating to database | | |
| Parameter | **Type** | **Description** |
| *rating* | Rating | The new station rating to be saved to database |

|  |  |  |
| --- | --- | --- |
| Signature: bool deleteComment(int commentId)  Description: Marks a coment as deleted in the database | | |
| Parameter | **Type** | **Description** |
| *commentId* | int | Id of the comment to be deleted |

|  |  |  |
| --- | --- | --- |
| Signature: bool clearRating(int ratingId)  Description: Marks a station rating as deleted in the database | | |
| Parameter | **Type** | **Description** |
| *ratingId* | Int | Id of the rating to be deleted |

#### IJourneyRepository Interface



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Station> GetStations()  Description: Retrieves all the stations from the database | | |
| Parameter | **Type** | **Description** |
| N/A | N/A | N/A |

|  |  |  |
| --- | --- | --- |
| Signature: Station GetStationByStationId(int stationId)  Description: Retrieves the station given its station id from the database | | |
| Parameter | **Type** | **Description** |
| *stationId* | int | Id of the station to be retrieved |

|  |  |  |
| --- | --- | --- |
| Signature: Station AddStation(Station station)  Description: Add new station to the database | | |
| Parameter | **Type** | **Description** |
| *station* | Station | The new station to be added to database |

|  |  |  |
| --- | --- | --- |
| Signature: bool UpdateStation(int stationId)  Description: Update station information from given its id from the database | | |
| Parameter | **Type** | **Description** |
| *stationId* | int | Id of the station to be updated |

|  |  |  |
| --- | --- | --- |
| Signature: bool DeleteStation(int stationId)  Description: Delete the station given its it from the database | | |
| Parameter | **Type** | **Description** |
| *stationId* | int | Id of the station to be deleted |

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Trip> GetTrips()  Description: Retrieves all the trips from the database | | |
| Parameter | **Type** | **Description** |
| N/A | N/A | N/A |

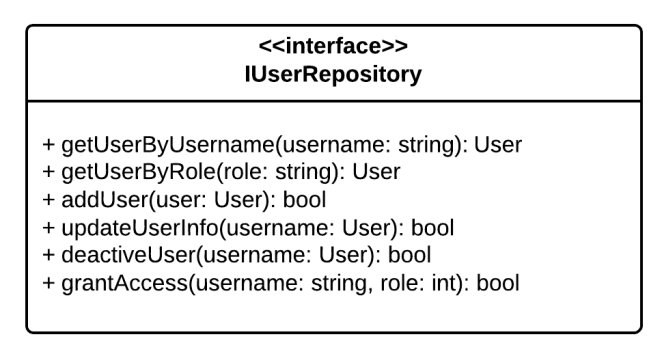
|  |  |  |
| --- | --- | --- |
| Signature: Station GetTripByTripId(int tripId)  Description: Retrieves the station given its station id from the database | | |
| Parameter | **Type** | **Description** |
| *tripId* | int | Id of the trip to be retrieved |

|  |  |  |
| --- | --- | --- |
| Signature: Station AddTrip(Trip trip)  Description: Add new trip to the database | | |
| Parameter | **Type** | **Description** |
| *trip* | Trip | The new trip to be added to database |

|  |  |  |
| --- | --- | --- |
| Signature: bool UpdateTrip(int tripId)  Description: Update trip information from given its id from the database | | |
| Parameter | **Type** | **Description** |
| *tripId* | int | Id of the trip to be updated |

|  |  |  |
| --- | --- | --- |
| Signature: bool DeleteTrip(int tripId)  Description: Delete the trip given its it from the database | | |
| Parameter | **Type** | **Description** |
| *tripId* | int | Id of the trip to be deleted |

#### IUserRepository Interface



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: User GetUserByUsername(String username)  Description: Retrieves a user given its username from the database | | |
| Parameter | **Type** | **Description** |
| *username* | String | Username of user to be retrieved |

|  |  |  |
| --- | --- | --- |
| Signature: User GetUserByRole(String role)  Description: Retrieves a user given its role from the database | | |
| Parameter | **Type** | **Description** |
| *role* | String | Role of user to be retrieved |

|  |  |  |
| --- | --- | --- |
| Signature: bool addUser(User user)  Description: Add new user to the database | | |
| Parameter | **Type** | **Description** |
| *user* | UserProfile | The user to be saved |

|  |  |  |
| --- | --- | --- |
| Signature: bool updateUser(int username)  Description: Update/Save user information to the database | | |
| Parameter | **Type** | **Description** |
| *username* | String | Username of user to be updated |

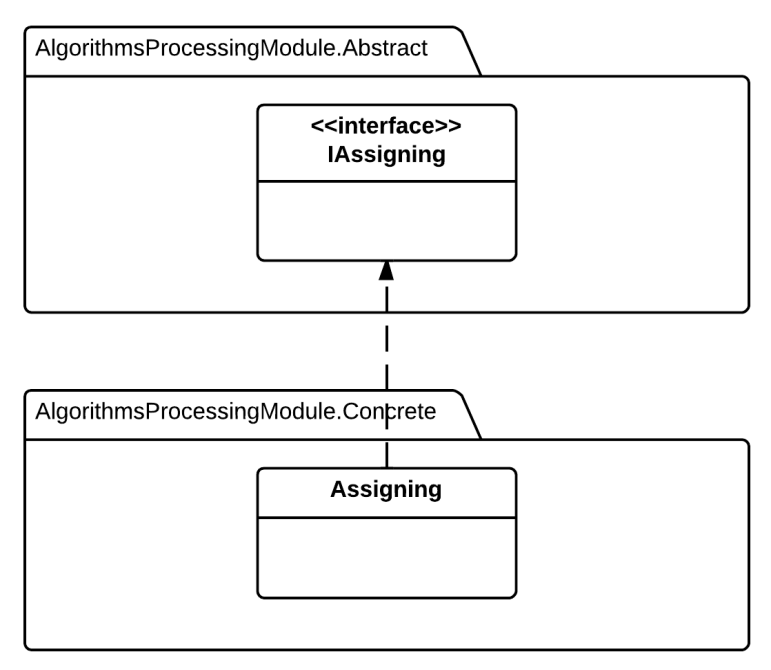
|  |  |  |
| --- | --- | --- |
| Signature: bool GrantAccess(String username, int roleId)  Description: Grants the selected role the the user with the user id | | |
| Parameter | **Type** | **Description** |
| *username* | String | Username of the user to be granted access |
| *roleId* | Role | Role that will be granted to the user |

|  |  |  |
| --- | --- | --- |
| Signature: bool DeactivateUser(String username)  Description: Marks the the user as deactivated in the database | | |
| Parameter | **Type** | **Description** |
| *username* | String | Username of the user to be marked as deactivated |

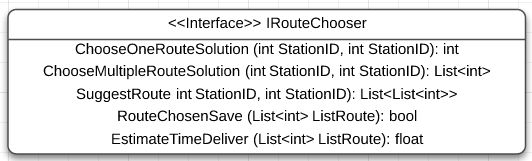
### Algoritms Processing Classes



#### Class diagram



#### IRouteChooser



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| **Signature:** int ChooseOneRouteSolution(int StartStation, int EndStation)  **Description:** return one suitable route to deliver from the start station to the end station, if the route not exsit, return -1. | | |
| **Parameter** | **Type** | **Description** |
| StartStation | int | The ID of the start station of the request |
| EndStation | int | The ID of the end station of the request |

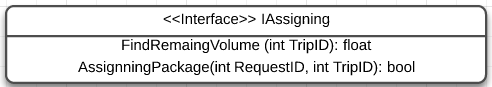
|  |  |  |
| --- | --- | --- |
| **Signature:** List <int> ChooseMultipleRouteSolution(int StartStation, int EndStation)  **Description:** return the list of suitable route to deliver from the start station to the end station. | | |
| **Parameter** | **Type** | **Description** |
| StartStation | int | The ID of the start station of the request |
| EndStation | int | The ID of the end station of the request |

|  |  |  |
| --- | --- | --- |
| **Signature:** List<int> SuggestRoute(int StartStation, int EndStation)  **Description:** Suggest for staff the best suitable route or list of routes base on the input start and end station of request. | | |
| **Parameter** | **Type** | **Description** |
| StartStation | int | The ID of the start station of the request |
| EndStation | int | The ID of the end station of the request |

|  |  |  |
| --- | --- | --- |
| **Signature:** bool RouteChosenSave (List<int> RouteList)  **Description:** Retrieves the list of route and save them to the table with exactly indicate order. | | |
| **Parameter** | **Type** | **Description** |
| RouteList | List<int> | List of route that staff choose or system suggest |

|  |  |  |
| --- | --- | --- |
| **Signature:** float EstimateTimeDeliver (List<int> RouteList)  **Description:** Find the estimate time to deliver the package base on the input route list and their relative information. | | |
| **Parameter** | **Type** | **Description** |
| RouteList | List<int> | List of route that staff choose or system suggest |

#### IAssigning



**Properties:** N/A

**Operations**

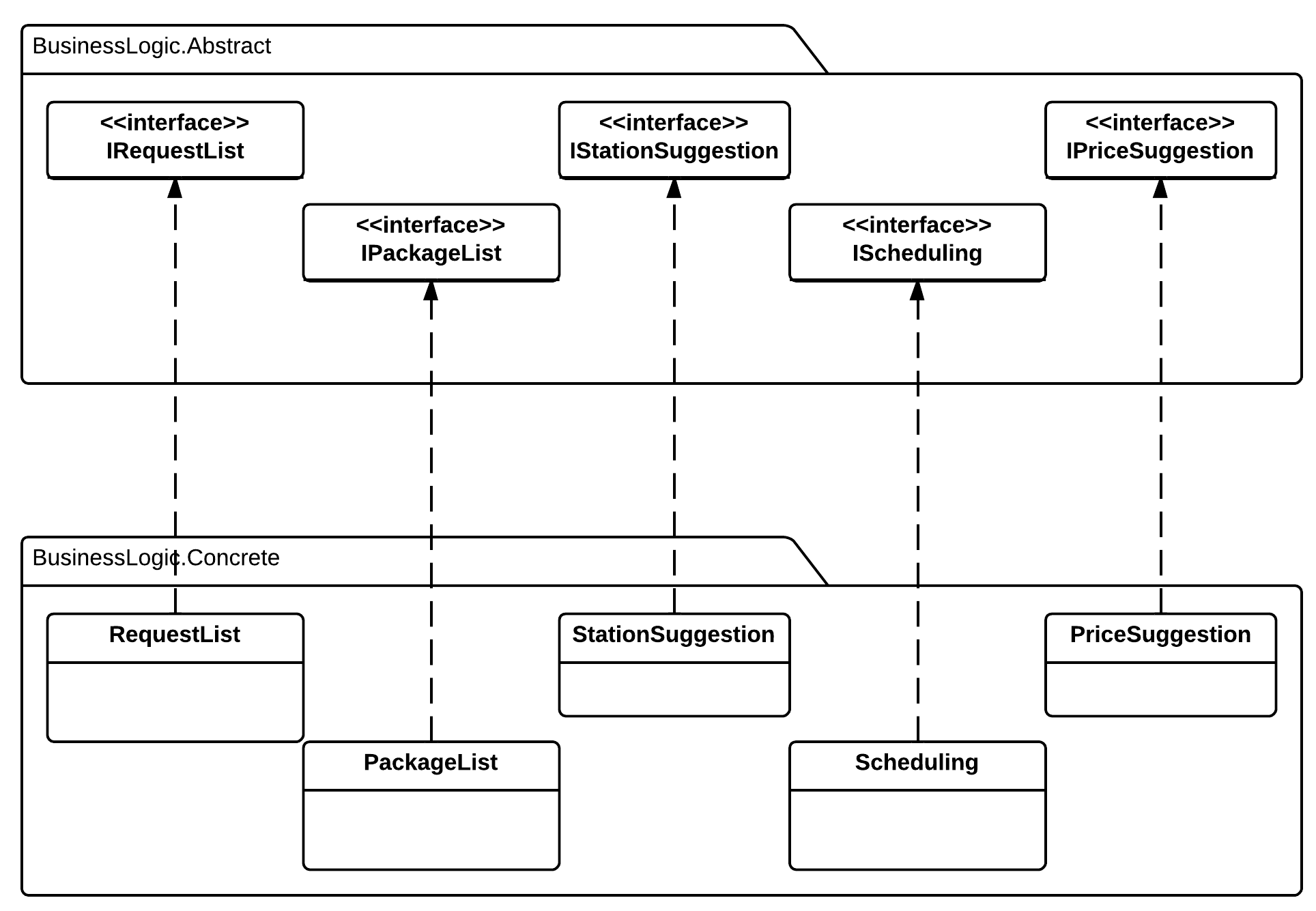
|  |  |  |
| --- | --- | --- |
| **Signature:** float FindRemaingVolume (int TripID)  **Description:** Return the remaining volume that the coach on input trip can be serve. | | |
| **Parameter** | **Type** | **Description** |
| TripID | int | The ID of trip that staff want to estimate. |

|  |  |  |
| --- | --- | --- |
| **Signature:** List <int> AssigningPackage (int RequestID)  **Description:** Automatically find the list of trip suitable for deliver the package. | | |
| **Parameter** | **Type** | **Description** |
| RequestID | int | The ID of request that staff processing. |

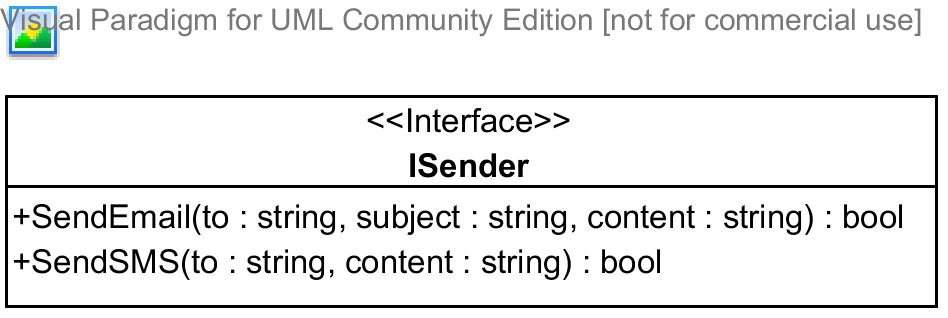
### Business Logic Classes



#### Class diagram



#### IRequestList Interface



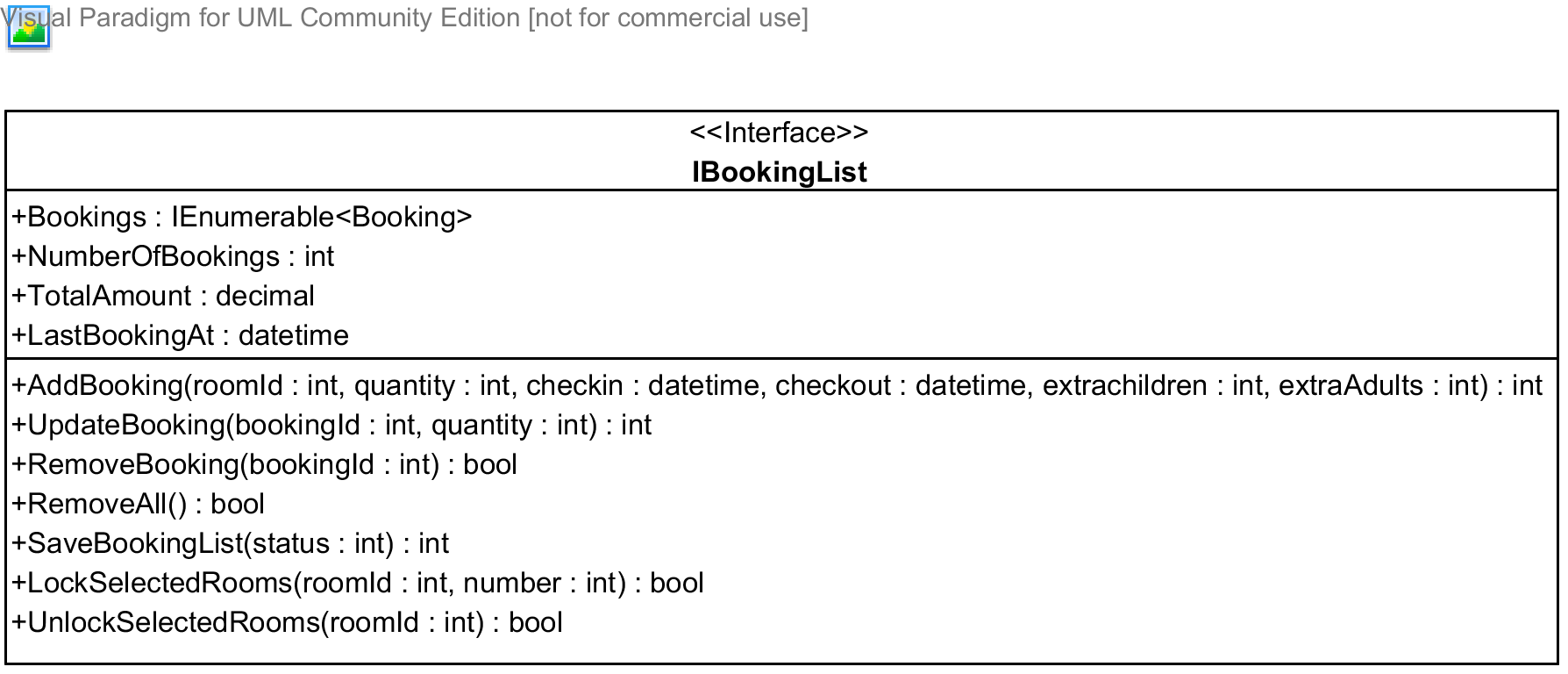
**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: bool SendEmail(string to, string subject, string content)  Description: Sends an email to the address specified | | |
| Parameter | **Type** | **Description** |
| *to* | string | The email address to send to |
| *subject* | string | The subject of the email |
| *content* | string | The content of the email |

|  |  |  |
| --- | --- | --- |
| Signature: bool SendSMS(string to, string content)  Description: Sends an SMS message to the address specified | | |
| Parameter | **Type** | **Description** |
| *to* | string | The telephone number to send to |
| *content* | string | The content of the message |

#### IStationSuggestion Interface



**Properties**

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Type | Visibility | Description |
| *Bookings* | IEnumerable<Booking> | public | A list of all bookings in the booking list for the current user |
| *NumberOfBooking* | int | public | Number of bookings in the booking list |
| *TotalAmount* | decimal | public | Total amount of all the booking |
| *LastBookingAt* | datetime | public | Time when the last booking was added to the booking list |

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: int AddBooking(int roomId, int quantity, datetime checkin, datetime checkout, int extraChildren, int extraAdults)  Description: Adds a bookings made by the current user to the booking list | | |
| Parameter | **Type** | **Description** |
| *roomId* | int | Id of room type being booked |
| *quantity* | int | Numbers of rooms needed |
| *checkin* | datetime | Check-in date |
| *checkout* | datetime | Check-out date |
| *extraChildren* | int | Number of extra children, if allowed by the hotel owner |
| *extraAdults* | int | Number of extra adults, if allowed by the hotel owner |

|  |  |  |
| --- | --- | --- |
| Signature: int UpdateBooking(int bookingId, int quantity)  Description: Update a booking in the booking list given its id | | |
| Parameter | **Type** | **Description** |
| *bookingId* | int | Id of the booking to be updated |
| *quantity* | int | Number of rooms to be updated |

|  |  |  |
| --- | --- | --- |
| Signature: bool RemoveBooking(int bookingId)  Description: Removed a booking in the booking list given its id | | |
| Parameter | **Type** | **Description** |
| *bookingId* | int | Id of the booking to be removed |

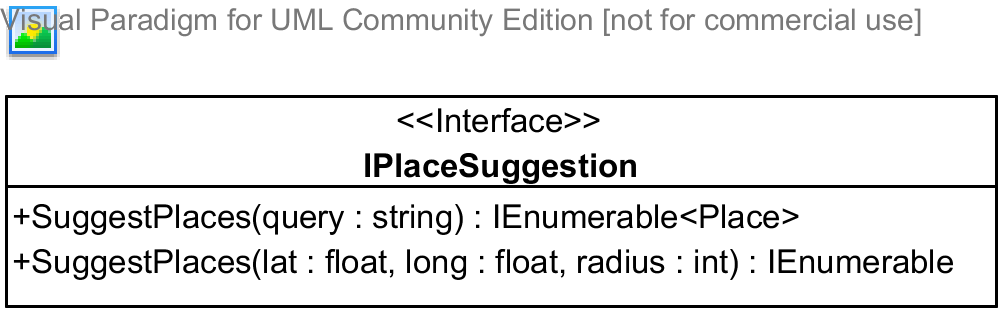
|  |  |  |
| --- | --- | --- |
| Signature: bool RemoveAll(int bookingId)  Description: Removed all the bookings int the booking list | | |
| Parameter | **Type** | **Description** |

|  |  |  |
| --- | --- | --- |
| Signature: bool SaveBookingList(int status)  Description: Saves the booking list to the database | | |
| Parameter | **Type** | **Description** |
| *status* | int | Status of the bookings to be saved |

|  |  |  |
| --- | --- | --- |
| Signature: bool LockSelectedRooms(int roomId, int number)  Description: Locks the rooms selected so that other user cannot book these rooms | | |
| Parameter | **Type** | **Description** |
| *roomId* | int | Id of the room type to be locked |
| *number* | int | Number of rooms to be locked for this type |

|  |  |  |
| --- | --- | --- |
| Signature: bool UnlockSelectedRooms(int roomId)  Description: Unlocks the rooms selected so that other user can book these rooms | | |
| Parameter | **Type** | **Description** |
| *roomId* | int | Id of the room type to be unlocked |

#### IPriceSuggestion Interface



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Place> SuggestPlaces(string query)  Description: Suggests places(hotel address, provinces, and other type of places,...) given a query string | | |
| Parameter | **Type** | **Description** |
| *query* | string | A string that will be used to search and suggest places |

|  |  |  |
| --- | --- | --- |
| Signature: IEnumerable<Place> SuggestPlaces(float lat, float long, int radius)  Description: Suggests places(hotel address, provinces, and other type of places,...) given the center of the destination and the radius to be searched | | |
| Parameter | **Type** | **Description** |
| *lat* | float | Latitude of the destination center |
| *long* | float | Longitude of the destination center |
| *radius* | int | Radius to limit the searching |

#### IPackageList Interface



**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: bool AddPageTabToHotel(string pageId, int hotelId)  Description: Adds a page tab to the hotel | | |
| Parameter | **Type** | **Description** |
| *pageId* | string | Id of the Facebook page to be added to the hotel |
| *hotelId* | int | Id of the hotel to be added |

|  |  |  |
| --- | --- | --- |
| Signature: Hotel RetrieveHotelForPage(string pageId)  Description: Retrieves a hotel given its pageId | | |
| Parameter | **Type** | **Description** |
| *pageId* | string | Facebook page id of the hotel to be retrieved |

|  |  |  |
| --- | --- | --- |
| Signature: bool RemovePage(int hotelId)  Description: Removes the Facebook page from the hotel given its id | | |
| Parameter | **Type** | **Description** |
| *hotelId* | int | Id of the hotel to remove the Facebook page |

#### IScheduling Interface

****

**Properties:** N/A

**Operations**

|  |  |  |
| --- | --- | --- |
| Signature: bool AddPageTabToHotel(string pageId, int hotelId)  Description: Adds a page tab to the hotel | | |
| Parameter | **Type** | **Description** |
| *pageId* | string | Id of the Facebook page to be added to the hotel |
| *hotelId* | int | Id of the hotel to be added |

|  |  |  |
| --- | --- | --- |
| Signature: Hotel RetrieveHotelForPage(string pageId)  Description: Retrieves a hotel given its pageId | | |
| Parameter | **Type** | **Description** |
| *pageId* | string | Facebook page id of the hotel to be retrieved |

|  |  |  |
| --- | --- | --- |
| Signature: bool RemovePage(int hotelId)  Description: Removes the Facebook page from the hotel given its id | | |
| Parameter | **Type** | **Description** |
| *hotelId* | int | Id of the hotel to remove the Facebook page |

## Behavioral Diagrams



### State Machine Diagram: Request Status Transition

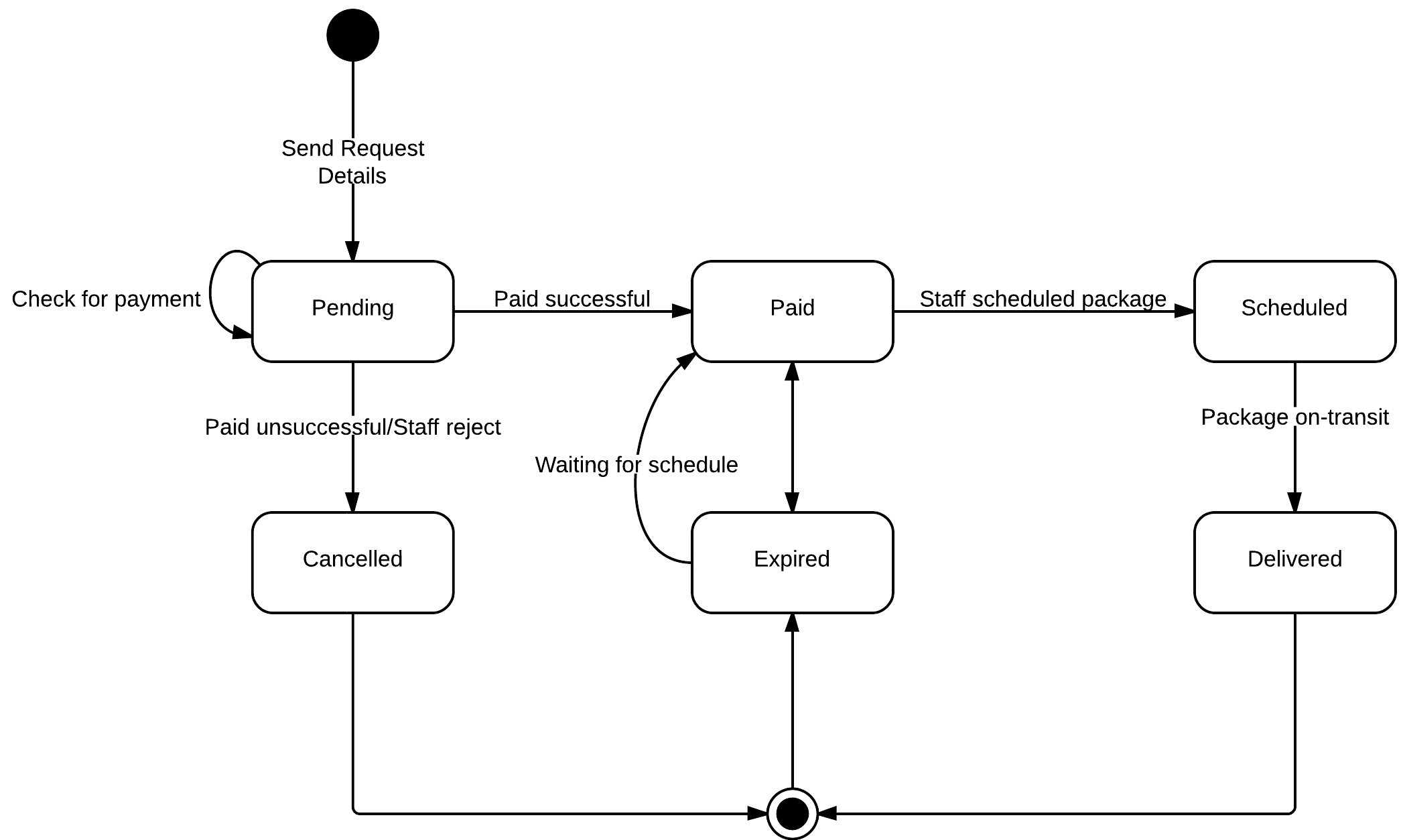


Diagram 2 State machine diagram: Requst status transition

### State Machine Diagram: Package Status Transition

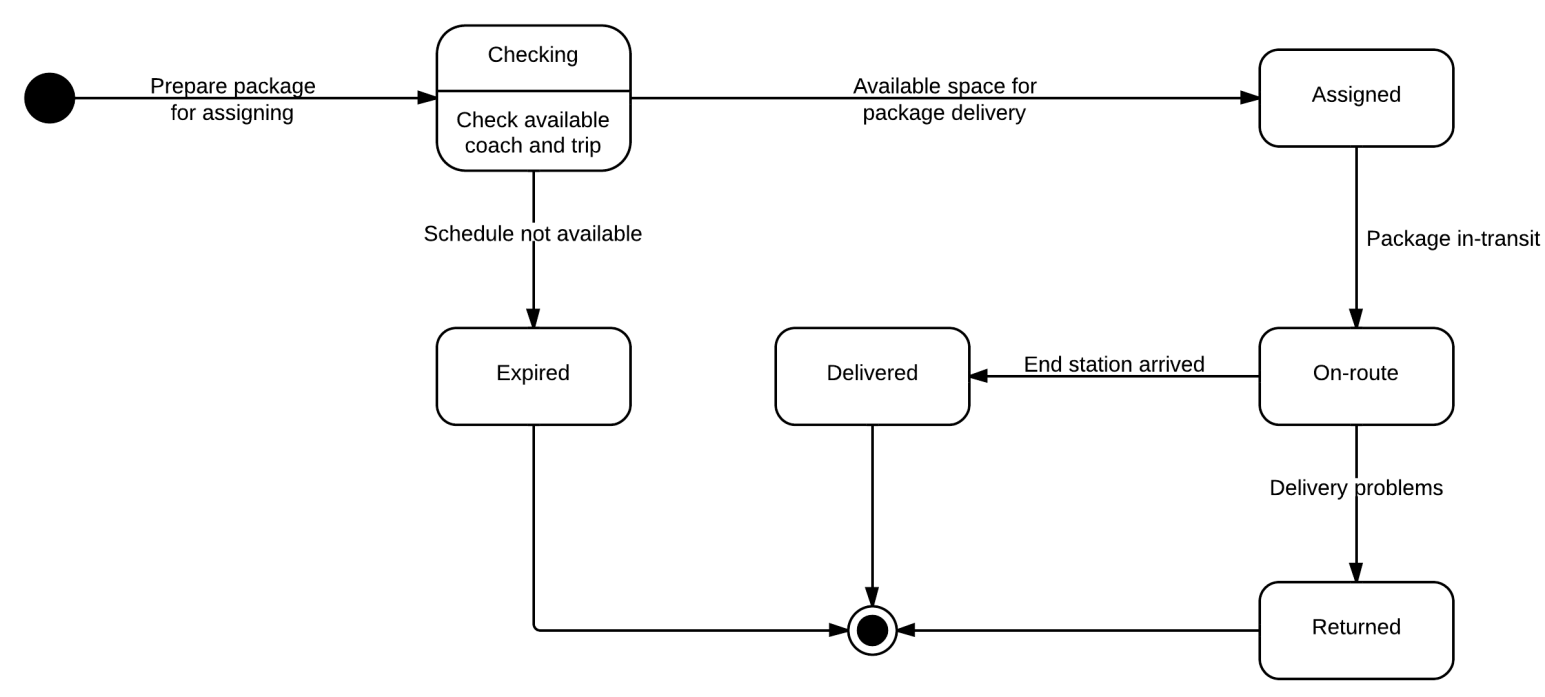


Diagram 3 State machine diagram: Package status transition

### Sequence Diagram: Search Routes

Diagram 4 Sequence diagram: Search Routes

### Sequence Diagram: Post Delivery Requests

Diagram 5 Sequence diagram: Post delivery requests

### Sequence Diagram: Create a Trip

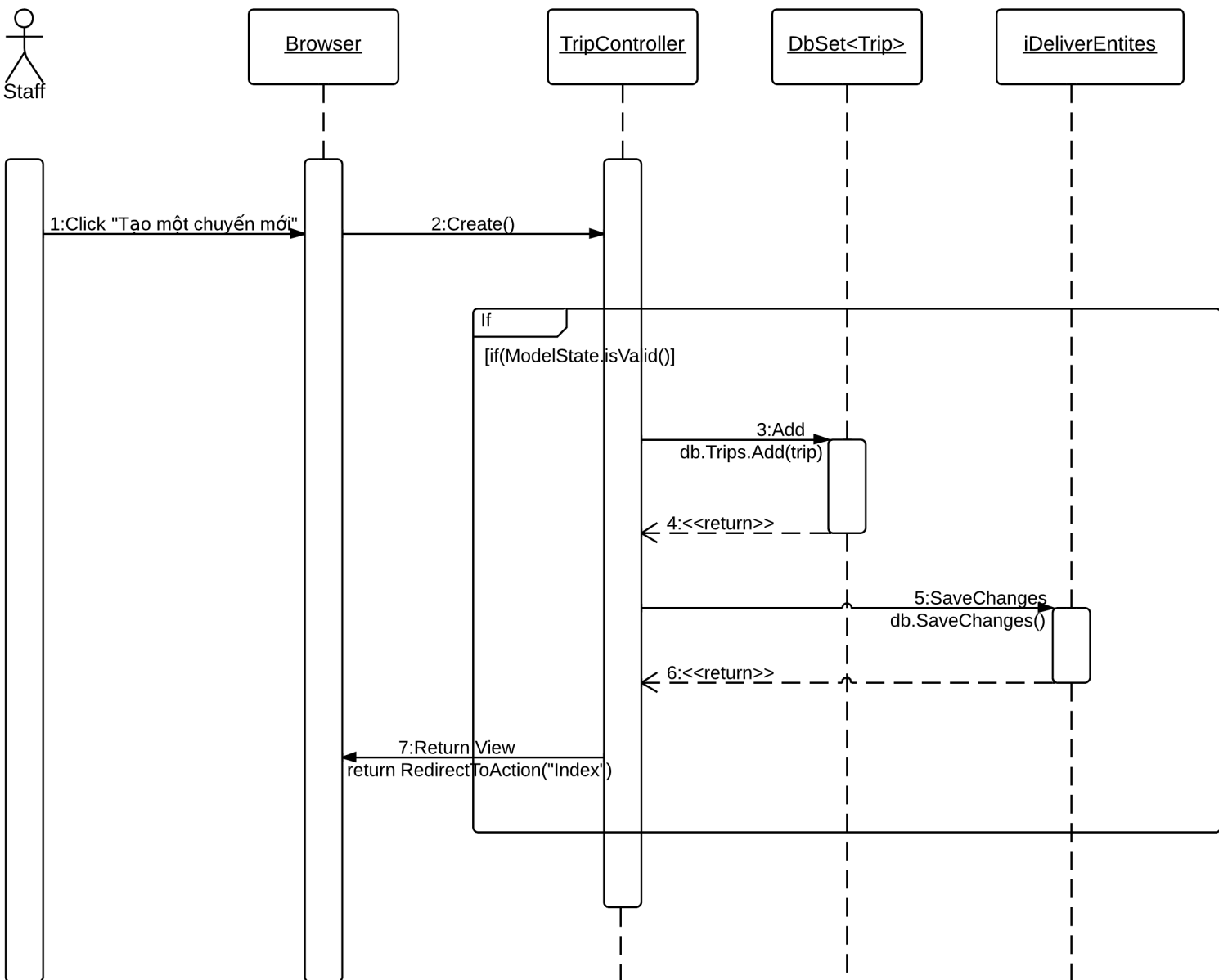


Diagram 6 Sequence diagram: Create a trip

### Sequence Diagram: Edit a Station

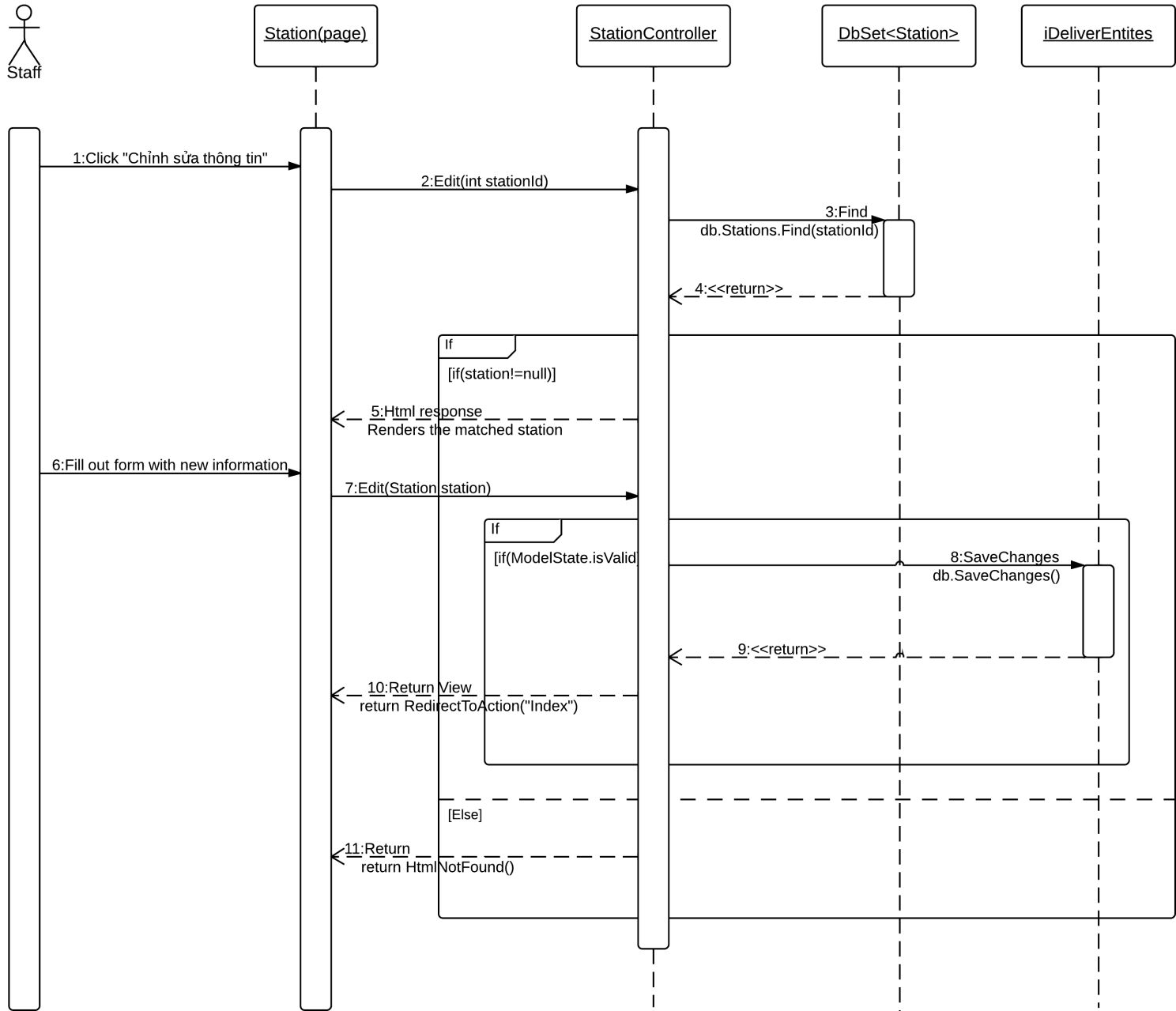


Diagram 7 Sequence diagram: Edit a station

### Sequence Diagram: Add a Price

Diagram 8 Sequence diagram: Add a price

### Sequence Diagram: Create a Staff Account

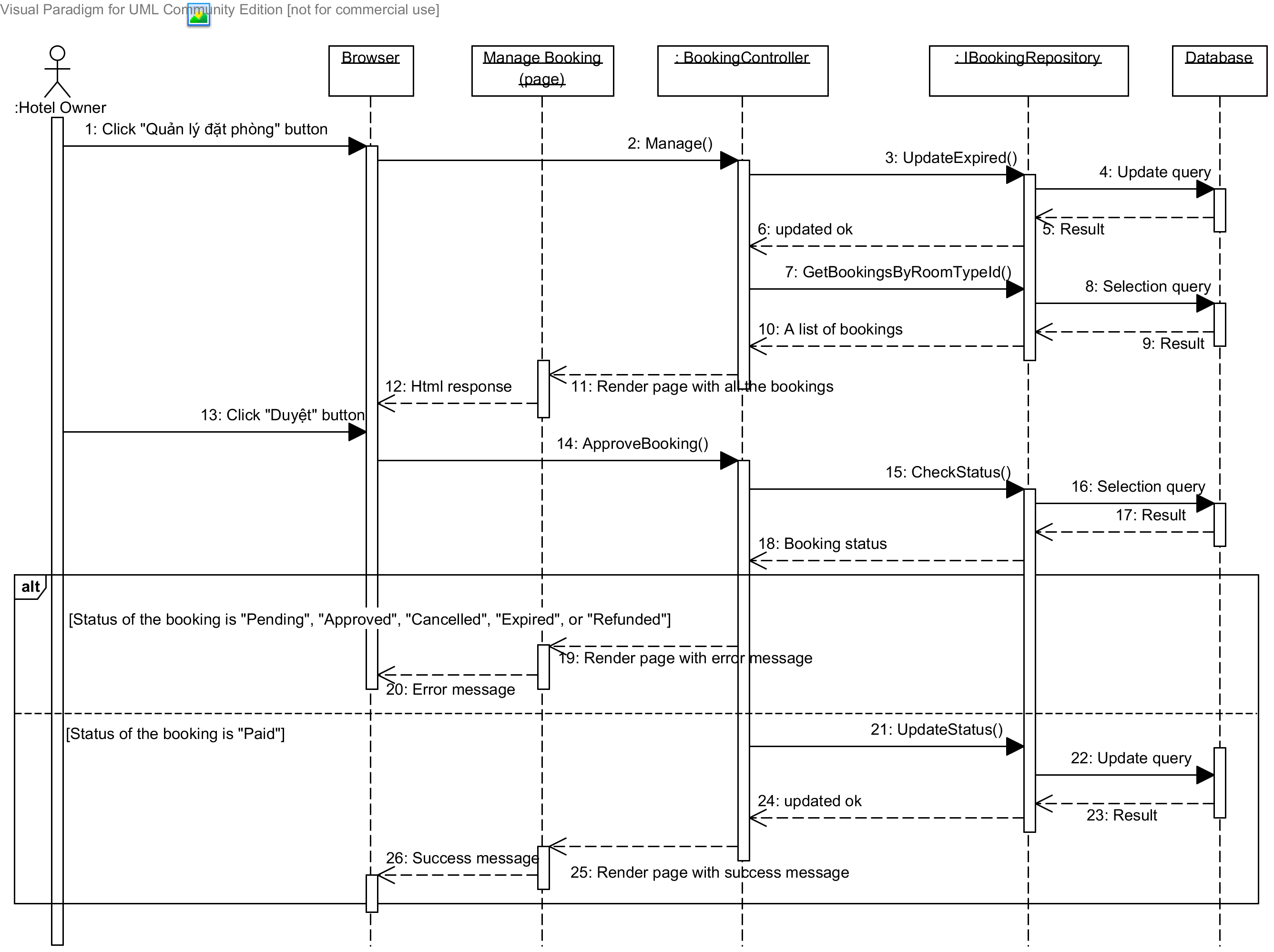


Diagram 9 Sequence diagram: Create a Staff Account

### Sequence Diagram: Approve Requests

Diagram 10 Sequence diagram: Approve requests

### Sequence Diagram: Assign Package

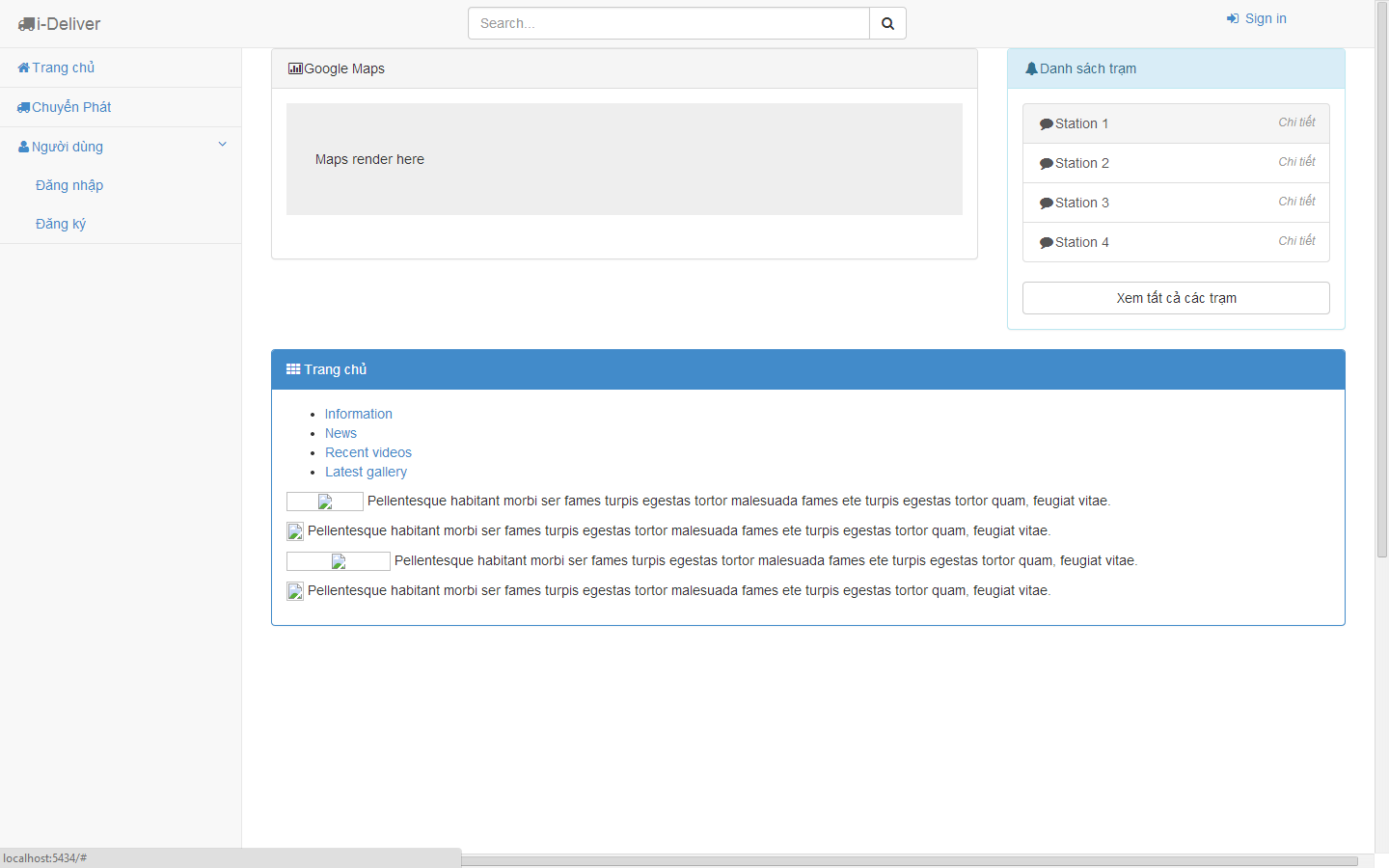
Diagram 11 Sequence diagram: Assign package

## User Interface Design

Below are some important screen designs for the MHB systems.



### Home Page



1

10

9

8

7

6

3

5

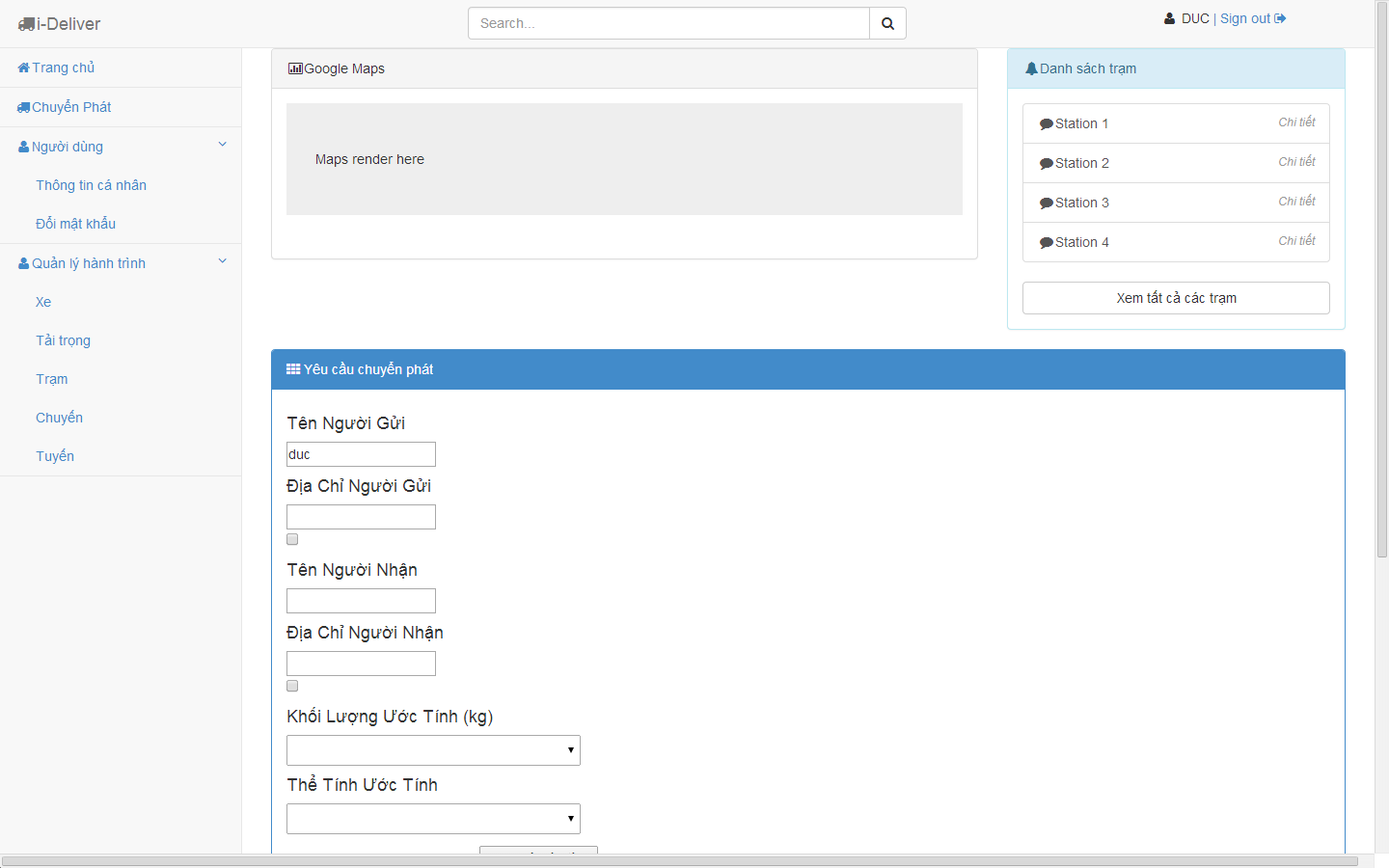
2

4

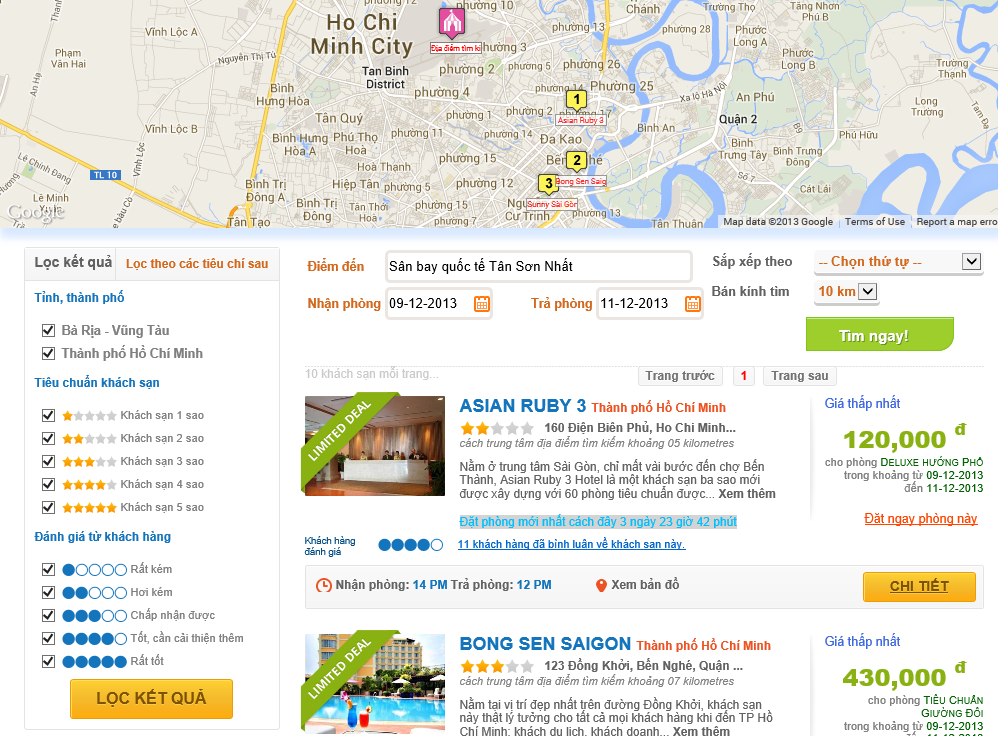
Figure 2 Home page

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | “Trang chủ” link | A link to the home page of the hotel information portal | N/A | N/A | Link | N/A |
| 2. | “Khách sạn” link | A link to the hotel list page | N/A | N/A | Link | N/A |
| 3. | “Khuyến mãi” | A link to the promotion page for all hotels | N/A | N/A | Link | N/A |
| 4. | “Dành cho chủ khách sạn”link | A link to the page for hotel owners | N/A | N/A | Link | N/A |
| 5. | “Đăng nhập” button | A button that displays the log in popup | N/A | N/A | Button | N/A |
| 6. | “Điểm đến” textbox | The place where the traveler wish to go to | No | Yes | Textbox | Text |
| 7. | “Từ ngày” texbox | The date when the traveler wants to check in | No | Yes | Textbox | Text |
| 8. | “Đến ngày” textbox | The date when the traveler wants to check out | No | Yes | Textbox | Text |
| 9. | “Tìm ngay” textbox | A button that leads to the search results page | N/A | N/A | Button | N/A |
| 10. | “Tìm kiếm nâng cao” button | A link to the advanced search page | N/A | N/A | Link | N/A |

### Search Hotels Page



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | “Điểm đến” textbox | The place where the traveler wishes to go to | No | Yes | Textbox | Text |
| 2. | “Từ ngày” textbox | The date when the traveler wants to check in | No | Yes | Textbox | Text |
| 3. | “Đến ngày” textbox | The date when the traveler wants to check out | No | Yes | Textbox | Text |
| 4. | “Chọn thứ tự” dropdown list | The order of the results | No | No | Dropdown list | N/A |
| 5. | “Bán kính” dropdown list | The radius from the center of the destination | No | No | Button | N/A |
| 6. | “Tìm kiếm” button | A button that searches and displays hotels | N/A | N/A | Button | N/A |
| 7. | “Tỉnh, thành phố” checkboxes | A groups of provinces to filter the results | No | No | Check-boxes | Text |
| 8. | “Tiêu chuẩn khách sạn” checkboxes | A groups of star qualities to filter the results | No | No | Check-boxes | Text |
| 9. | “Đánh giá” checkboxes | A group of user rating averages to filter the results | No | No | Check-boxes | Text |
| 10. | “Lọc kết quả” button | A button that filters the results found | N/A | N/A | Link | N/A |
| 11. | Details of a hotel section | Detailed information of a hotel | N/A | N/A | Div | N/A |
| 12. | Price of a hotel section | Lowest price for a hotel. | N/A | N/A | Div | N/A |
| 13. | Google map section | A Google map that displays all the hotels found | N/A | N/A | Map | N/A |
| 14. | Marker for a destination | A Google marker for the destination the user searches | N/A | N/A | Marker | N/A |
| 15. | Marker for a hotel | A Google marker for a hotel in the result | N/A | N/A | Marker | N/A |



15

14

13

12

11

10

1

4

5

6

9

8

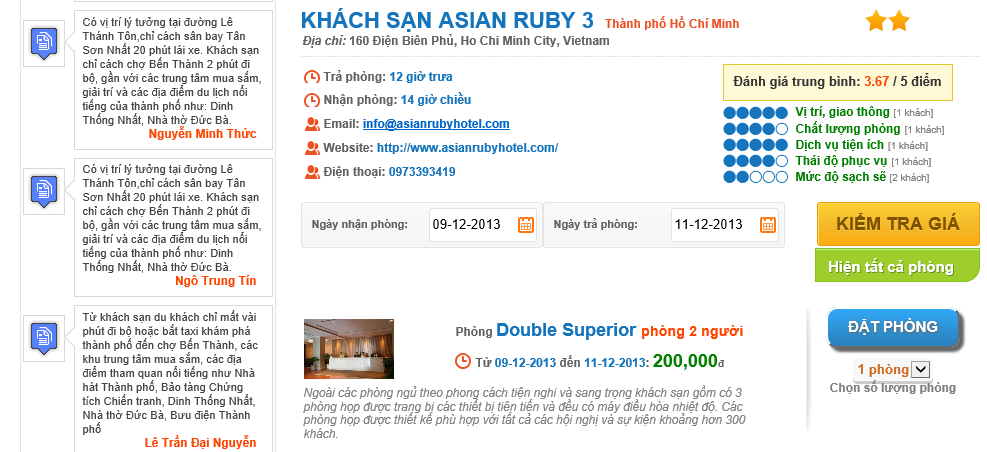
7

3

2

Figure 3 Search Hotel page

### Hotel Details Page



2

11

10

6

5

9

71

8

4

3

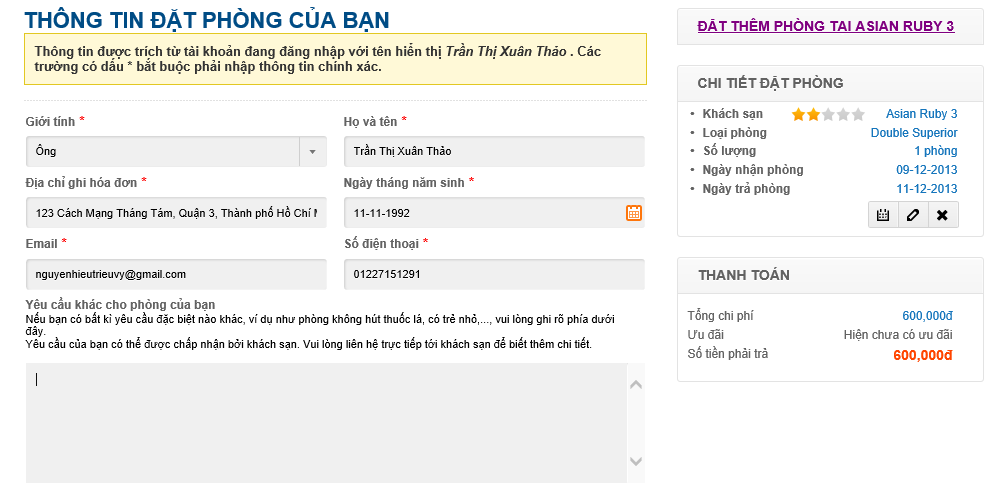
1

Figure 4 Hotel Details page

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | Details of a hotel section | Detailed information of a hotel | N/A | N/A | Div | N/A |
| 2. | Rating average points section | Rating average points rated by traveler. | N/A | N/A | Div | N/A |
| 3. | “Ngày nhận phòng” textbox | The date when the traveler wants to check in | No | Yes | Textbox | Text |
| 4. | “Ngày trả phòng” textbox | The date when the traveler wants to check out | No | Yes | Textbox | Text |
| 5. | “Kiểm tra giá” button | A button that searches and displays room types available | N/A | N/A | Button | N/A |
| 6. | “Hiện tất cả phòng” button | A button that displays all room types of a hotel | N/A | N/A | Button | N/A |
| 7. | Details of a room type section | Detailed information of a room type | N/A | N/A | Div | N/A |
| 8. | Price section | Price for a room in the searching period. If there are many prices for this period, all of them are displayed. | N/A | N/A | Div | N/A |
| 9. | “Đặt phòng” button | A button that leads to the Booking Details page | N/A | N/A | Button | N/A |
| 10. | “Số lượng phòng” dropdown list | The number of rooms the traveler wants to book | No | No | Dropdown list | Text |
| 11. | Review section | A review for the hotel | N/A | N/A | Div | N/A |

### Booking Details Page

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | “Giới tính” dropdown list | Gender of the traveler | No | Yes | Textbox | Text |
| 2. | “Họ và tên” textbox | Full name of the traveler | No | Yes | Textbox | Text |
| 3. | “Địa chỉ” textbox | Address line of the traveler | No | Yes | Textbox | Text |
| 4. | “Ngày tháng năm sinh” textbox | Date of birth of the traveler |  | Yes | Textbox | Text |
| 5. | “Email” textbox | Email address of the traveler | No | Yes | Textbox | Text |
| 6. | “Số điện thoại” textbox | Telephone number of the traveler | No | Yes | Textbox | Text |
| 7. | “Yêu cầu” text area | Any custom request the traveler want to send to the hotel | No | No | Text area | Text |
| 8. | “Đặt thêm phòng” link | A link to go back to the hotel to book more rooms | N/A | N/A | Link | N/A |
| 9. | “Chi tiết đặt phòng” section | The booking detail of the rooms booked | N/A | N/A | Div | N/A |
| 10. | Details button | A button that displays the price details of a booking when it is hovered | N/A | N/A | Button | N/A |
| 11. | Edit booking button | A button to edit the current booking | N/A | N/A | Button | N/A |
| 12. | Delete booking button | A button to remove the current booking | N/A | N/A | Button | N/A |
| 13. | “Thanh toán” section | The detail of total amount | N/A | N/A | Div | N/A |
| 14. | “Thanh toán” button | A button that leads to the payment page | N/A | N/A | Button | N/A |



8

14

13

12

11

10

9

7

6

5

4

3

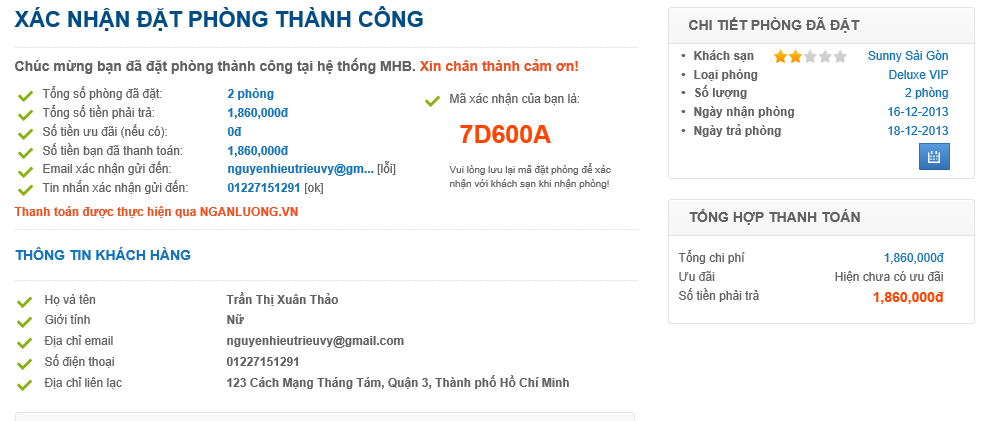
1

2

Figure 5 Booking Details page

### Booking Success Page

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | Payment information section | The details of the payment made by the customer | N/A | N/A | Div | N/A |
| 2. | Booking code | The booking code generated for the booking made | N/A | N/A | Div | N/A |
| 3. | Customer information | The detailed information of the customer who made the booking | N/A | N/A | Div | N/A |
| 4. | Details of bookings | The details of the booking made by the customer | N/A | N/A | Div | N/A |



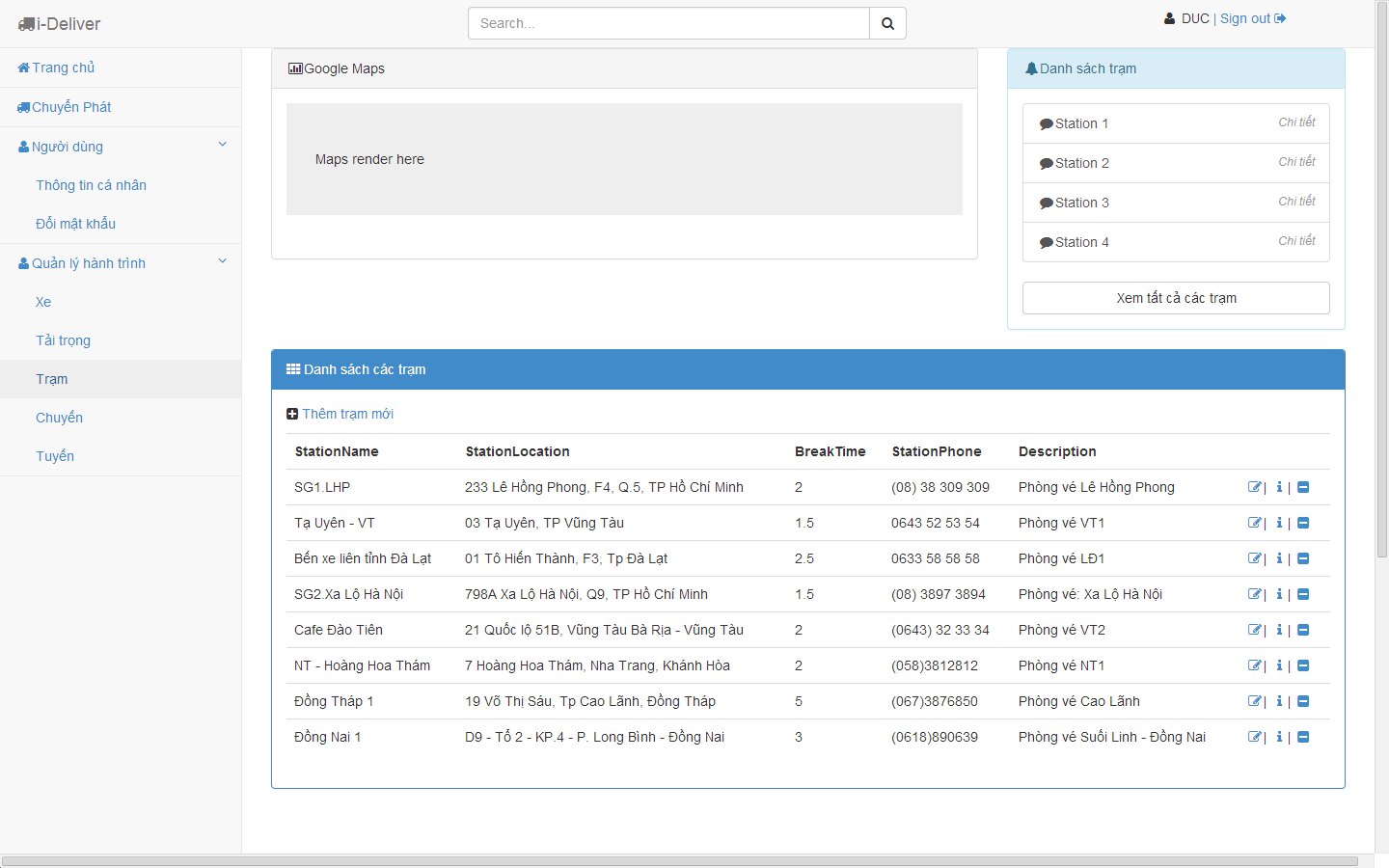
4

3

2

1

### Dashboard Page



17

16

15

14

13

12

11

10

9

8

7

6

5

4

3

2

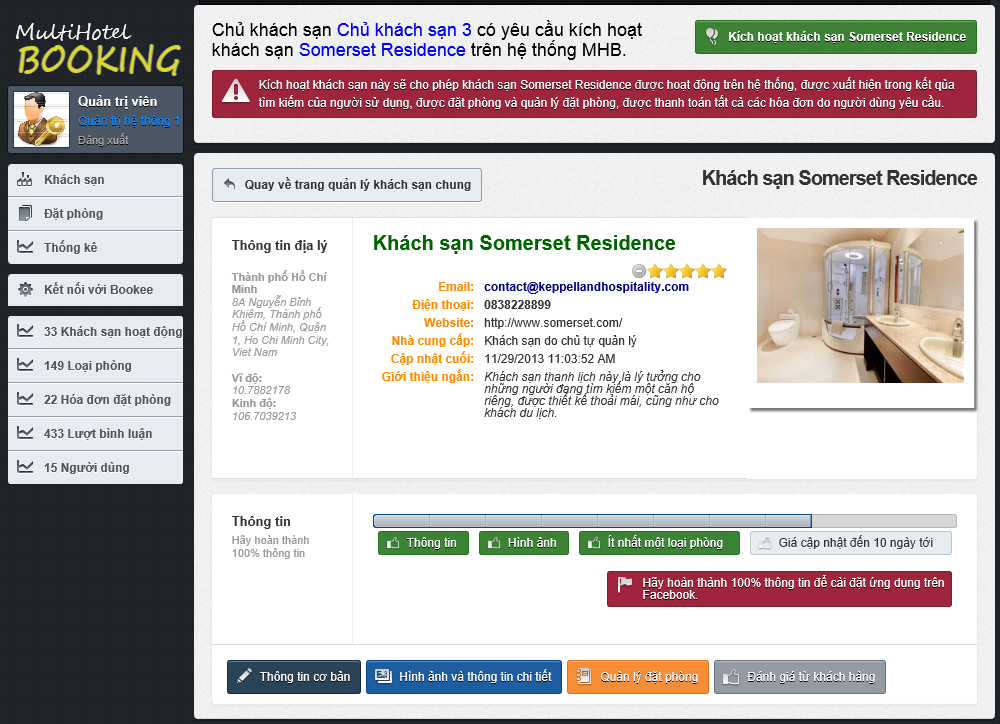
1

Figure 6 Dashboard page

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | User information section | Information of the logged user | N/A | N/A | Div | N/A |
| 2. | “Khách sạn” link | A link to the dashboard page | N/A | N/A | Link | N/A |
| 3. | “Đặt phòng” link | A link to the booking manangement page | N/A | N/A | Link | N/A |
| 4. | “Thống kê” link | A link to the statistics page | N/A | N/A | Link | N/A |
| 5. | “Kết nối tới Bookee” link | A link to the data synchronization page | N/A | N/A | Link | N/A |
| 6. | New hotel opening requests notification | A notification for new hotel opening request issued by hotel owner | N/A | N/A | Div | N/A |
| 7. | New bookings notification | A notification for new bookings in all the hotels | N/A | N/A | Div | N/A |
| 8. | New bookings notification for a hotel | A notification for new bookings in a specific hotel | N/A | N/A | Div | N/A |
| 9. | “Tất cả” button | A button to list all the hotels | N/A | N/A | Button | N/A |
| 10. | “Tự quản lý” button | A button to filter on local hotels | N/A | N/A | Button | N/A |
| 11. | “MHB quản lý” button | A button to filter on external hotels | N/A | N/A | Button | N/A |
| 12. | “Tên” button | A button to sort the hotels by their names | N/A | N/A | Button | N/A |
| 13. | “Mới cập nhật” button | A button to sort the hotels by their last modified time | N/A | N/A | Button | N/A |
| 14. | “Mới đặt phòng” button | A button to sort the hotels by their last booked time | N/A | N/A | Button | N/A |
| 15. | Hotel list section | A list of all the hotels | N/A | N/A | Div | N/A |
| 16. | “Tạo khách sạn mới” button | A button that redirects to the hotel creating page | N/A | N/A | Button | N/A |
| 17. | Newly booked hotels section | A list of 10 hotel that have the most new bookings. | N/A | N/A | Div | N/A |

### Hotel Management Page

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | “Kích hoạt khách sạn” button | A button used to activate a hotel opening request | N/A | N/A | Button | N/A |
| 2. | “Quay về trang quản lý khách sạn chung” button | A link to the dashboard page | N/A | N/A | Button | N/A |
| 3. | Name of the hotel | Name of the hotel you are viewing | N/A | N/A | Label | N/A |
| 4. | Information section | Detailed information of the hotel being viewed | N/A | N/A | Div | N/A |
| 5. | Completeness measure | Percent of information completeness for the hotel | N/A | N/A | Div | N/A |
| 6. | “Thông tin cơ bản” button | A link to the hotel editing page | N/A | N/A | Button | N/A |
| 7. | “Hình ảnh và thông tin nâng cao” button | A link to the hotel resouces page | N/A | N/A | Button | N/A |
| 8. | “Quản lý đặt phòng” button | A link to the booking management page | N/A | N/A | Button | N/A |
| 9. | “Đánh giá của người dùng” button | A link to the review management page | N/A | N/A | Button | N/A |



9

8

7

6

5

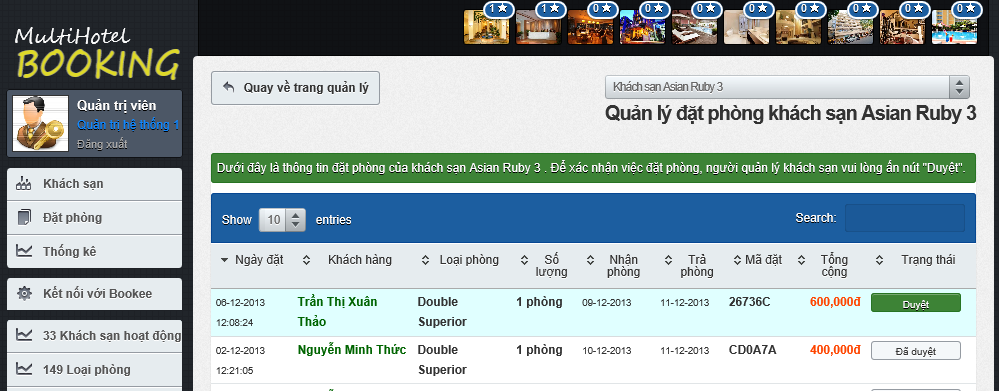
4

3

2

1

### Approve Booking Page



3

2

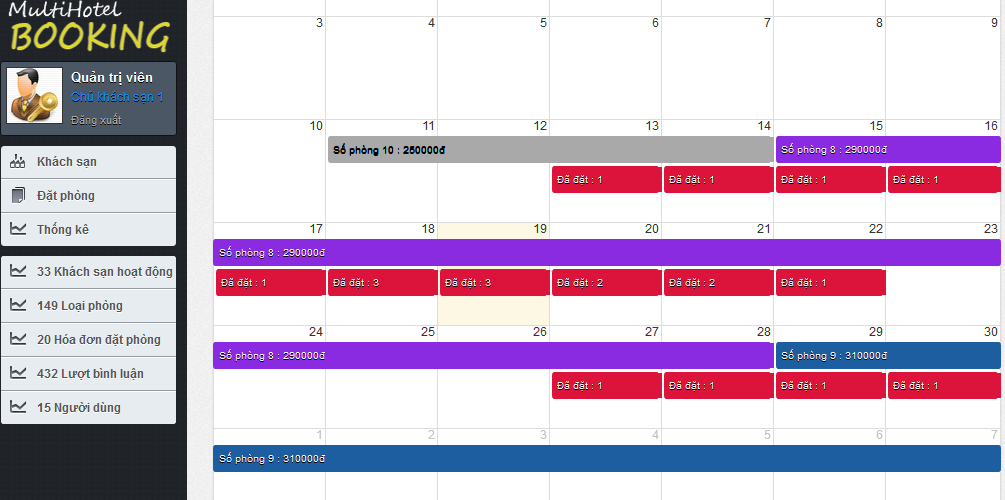
1

Figure 7 Approve Booking page

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | Booking detail row | Details of a booking | N/A | N/A | Row | N/A |
| 2. | “Duyệt” link | A button to approve of a booking made by a traveler | N/A | N/A | Button | N/A |
| 3. | “Đã duyệt” link | A disabled button that indicates that the booking was approved sucessfully | Yes | N/A | Button | N/A |

### Prices Monitor Page

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Element | Description | Read Only | Mandatory | Control Type | Data type |
| 1. | Gray price bar | A price bar (in gray) that represents a price in the past. | N/A | N/A | Div | N/A |
| 2. | Violet price bar | A price bar (in violet) that represents a price crossing from the past to the future (including the current system date). | N/A | N/A | Div | N/A |
| 3. | Blue price bar | A price bar (in blue) that represents a price in the future. | N/A | N/A | Div | N/A |
| 4. | Red bar | The number of bookings made on a specific day for the selected room type. | N/A | N/A | Div | N/A |



1

3

2

4

## Database Design



### Logical database design

From the ERD specified in the SRS section, logical database design includes the following relations:

|  |  |  |
| --- | --- | --- |
| **Index** | **Table Name** | **Description** |
| 1 | Account | List of user that registered to the system with valid username and password. |
| 2 | UserInfo | Detail information of each user. |
| 3 | Coach | List of all coach the center has. |
| 4 | CoachType | List of type of coach base on the number of seat that the coach has. |
| 5 | Route | List of route that center can deliver to. |
| 6 | Trip | Information of trip of each coach the center have everyday. |
| 7 | Station | The list of all stations the center has. |
| 8 | Volume | The maximum volume that each type of coach can deliver on each of route. |
| 9 | Request | Information of each request that user posted to the system. |
| 10 | DeliveryStatus | Status of the request of base on the action of user and staff. |
| 11 | Invoice | Detail of invoice corresonding to the approve request. |
| 12 | ManageFee | The fee of service base on the range of volume and range of weight. |
| 13 | Comment | Content of comment of user posted for each station. |
| 14 | Rating | Rating level that user rate for each station. |
| 15 | Assigning | Table of mapping between coach and request. |



The details of each relation are specified below

* + 1. **Table description**
       1. **User**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| Username | int |  | - | PK | Username of user |
| Password | nvarchar(50) |  | - |  | Password of user |
| BackupPassword | nvarchar(50) |  | - |  | Backup password for user |
| Role | nvarchar(50) |  | - |  | Role of user in the system:  customer, admin or staff |
| Email | Nvarchar(50) |  | - |  | Email of user |
| Phone | Nvarchar(50) |  | - |  | Phone of user |
| BannedStatus | Bit |  | - |  | User is banned or not |

Unique: Username.

Foreign key: N/A.

* + - 1. **UserInfo**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| Username | nvarchar(50) |  | - | FK | Username of user registerd to the system or the staff |
| Firstname | nvarchar(50) |  | - |  | Firstname of user registerd to the system or the staff |
| Lastname | nvarchar(50) |  | - |  | Lastname of user registerd to the system or the staff |
| Address | nvarchar(500) |  | - |  | Address of user registerd to the system or the staff |

Unique: Username.

Foreign key: N/A.

* + - 1. **Coach**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CoachID | int |  | Auto generated | PK | Uniquely identifies of the coach |
| NumberPlate | nvarchar(50) |  | - |  | Number plate of coach |
| RouteID | int |  | - | FK | The route that coach travel |
| CoachTypeID | int |  | - | FK | The type of coach |

Unique: CoachID, NumberPlate.

Foreign key: RouteID (table Route), CoachTypeID (table CoachType).

* + - 1. **CoachType**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CoachTypeID | Int |  | Auto generated | PK | Uniquely identifies of the coach |
| Seats | int |  | - |  | The number of seats of the coach, that define the type of coach |

Unique: Seats.

Foreign key: N/A.

* + - 1. **Route**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| RouteID | int |  | Auto generated | PK | Uniquely identifies a route |
| RouteName | nvarchar(50) |  | - |  | The name of route |
| StartPoint | int |  | - | FK | The start position (station) of this route |
| EndPoint | int |  | - | FK | The end position (station) of this route |
| AvgRateLevel | float |  | - |  | The average rate level that customer rate for the route |
| Duration | float |  | - |  | The estimate duration (in hours) for moving on this route |
| Distance | float |  | - |  | The estimate distance (in km) between start and end location |

Unique: RouteID.

Foreign key: StartPoint (table Station), EndPoint (table Station).

* + - 1. **Trip**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| TripID | int |  | Auto generated | PK | Uniquely identifies a route |
| EstimateDepartureTime | time |  | - |  | The estimate time for the trip departures in each day |
| RealDepartureTime | time |  | - |  | The real time that the trip departures in a specific day |
| EstimateArrivalTime | time |  | - |  | The estimate time for the trip departures in each day |
| RealArrivalTime | float |  |  |  | The real time for the trip arrival to the station in a specific day |
| AvailableVolume | float |  |  |  | The volume that the coach on this trip can deliver |
| Date | date |  |  |  | The date that trip start. |
| RouteID |  |  |  | FK | The route that the trip traveling on |
| CoachID |  |  |  | FK | The coach of this trip. |

Unique: RouteID.

Foreign key: RouteID (table Route), CoachID (table Coach).

* + - 1. **Station**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| StationID | int |  | Auto generated | PK | Uniquely identifies a route |
| StationName | nvarchar(50) |  | - |  | The name of the station |
| StationLocation | nvarchar(50 |  | - |  | The address of location |
| BreakTime | float |  | - |  | The default break time that for each trip come to this station |
| StationPhone | nvarchar(50) |  | - |  | The telephone number of this station |
| Description | nvarchar(500) |  | - |  | The desciption for this station |
| Longitude | float |  | - |  | The longitude of the station location |
| Latitude | float |  | - |  | The latitude of the station location |
| Province | nvarchar |  | - |  | The province of the statino location |

Unique: StationID.

Foreign key: N/A.

* + - 1. **Volume**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| VolumeID | int |  | Auto generated | PK | Uniquely identifies the volume |
| MaxVolume | float |  | - |  | The maximum volume the coach can be deliver on the route |
| MinVolume | float |  | - |  | The minimum volume the coach can be deliver on the route |
| RouteID | int |  | - | FK | The route in this volume define |
| CoachTypeID | int |  | - | FK | The coach type in this volume define |

Unique: StationID.

Foreign key: RouteID (table Route), CoachTypeID (table CoachType).

* + - 1. **Request**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| RequestID | Int |  | Auto generated | PK | Uniquely identifies request |
| Username | int |  | - | FK | Uniquely identifies request type |
| DeliveryStatusID | int |  | - | FK | Uniquely identifies good type |
| FeeID | int |  | - | FK | Uniquely identifies user |
| EstimateWeight | int |  | - |  | Date of request |
| EstimateVolume | int |  | - |  | Description for request |
| DateRequest | bit |  | - |  | The date that user post the request |
| Description | nvarchar(500) |  | - |  | The addition description of request |
| FromLocation | int |  | - |  | The suggest station for package send to center |
| ToLocation | int |  | - |  | The station suggest for package deliver to |
| SenderAddress | nvarchar(500) |  | - |  | The address of sender |
| ReceiverAddress | nvarchar(500) |  | - |  | The address of receiver |
| SenderPhone | nvarchar(50) |  | - |  | The phone number of the sender |
| ReceiverPhone | nvarchar(50) |  | - |  | The phonenumber of receiver |

Unique: RequestID.

Foreign key: Username (table Account), DeliveryStatusID (table DeliveryStatus), FeeID (table Fee).

* + - 1. **DeliveryStatus**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| DeliveryStatusID | Int |  | Auto generated | PK | Uniquely identifies the delivery status |
| StatusName | nvarchar(50) |  | - |  | The name of delivery status of the package |
| Note | nvarchar(500) | X | - |  | Some more description for the status |

Unique: DeliveryStatusID.

Foreign key: N/A.

* + - 1. **Invoice**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| InvoiceID | Int |  | Auto generated | PK | Uniquely identifies a invoice |
| Weight | float | X | - |  | Weight of package |
| Volume | int | X | - |  | Volume of package |
| Price | float |  | - |  | Price of delivery service |
| RequestID | int |  | - | FK | The request that the invoice point to |

Unique: InvoiceID.

Foreign key: RequestID (table Request).

* + - 1. **ManageFee**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| FeeID | int |  | Auto generated | PK | Uniquely identifies a fee rule |
| Fee | float |  | - |  | The fee of this weight and volume range |
| MinWeight | float |  | - |  | The minimum weight of the weight range |
| MaxWeight | float |  | - |  | The maximum weight of the weight range |
| MinVolume | float |  | - |  | The minimum volume of the volume range |
| MaxVolume | float |  | - |  | The maximum volume of the volume range |

Unique: FeeID.

Foreign key: N/A.

* + - 1. **Comment**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CommentID | int |  | Auto generated | PK | Uniquely identifies the comment |
| Username | nvarchar(50) |  | - | FK | The user post comment |
| CommentContent | nvarchar(4000) |  | - |  | Content of comment |
| DatePost | date |  | - |  | Date that user post the comment |
| Status | bit |  | - |  | Approve or reject the comment |
| StationID | int |  | - | FK | The station that customer comment for |

Unique: CommentID.

Foreign key: Username (table User), StationID (table Station).

* + - 1. **Rating**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| RatingID | int |  | Auto generated | PK | Uniquely identifies the rating |
| Username | nvarchar(50) |  | - | FK | The user post comment |
| RateLevel | int |  | - |  | The level of rate: from 1 to 5 |
| StationID | int |  | - | FK | The station that user rating for |

Unique: RatingID.

Foreign key: Username (table User), StationID (table Station).

* + - 1. **Assigning**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Allow Null** | **Default Value** | **P/F Key** | **Description** |
| CoachID | int |  | - | PK | The ID of coach for the specific request |
| RequestID | int |  | - | The ID of request |
| IndicateOrder | int |  | - |  | The indicated order of the assigning the package of this request |

Unique: (CoachID, RequestID).

Foreign key: N/A.

### Physical database design

Please note that in the physical database implemented using SQL Server 2008 R2, the tables wrapped in the dashed line, (namely *webpages\_Roles*, *webpages\_UserInRoles*, *webpage\_OAuthMembership*, and *webpage\_Membership*) are generated and supported by the MVC .NET 4.0 framework to help simplify the authentication and authorization processes.



Diagram 21. Physical database design