

**CAPSTONE PROJECT REPORT**

**Report 4 – Software Design Document**

– HoChiMinh, September 2020 –

**Table of Contents**

[I. Project Report 4](#_heading=h.gjdgxs)

[1. Status Report 4](#_heading=h.30j0zll)

[2. Team Involvements 4](#_heading=h.1fob9te)

[3. Issues/Suggestions 4](#_heading=h.3znysh7)

[II. Software Design Document 5](#_heading=h.2et92p0)

[1. Overall Description 5](#_heading=h.tyjcwt)

[1.1 Assumptions 5](#_heading=h.3dy6vkm)

[1.2 Design Constraints 5](#_heading=h.1t3h5sf)

[2. System Architecture Design 5](#_heading=h.4d34og8)

[2.1 Overall Architecture 5](#_heading=h.2s8eyo1)

[2.2 System Architecture 6](#_heading=h.17dp8vu)

[2.3 Package Diagram 6](#_heading=h.3rdcrjn)

[3. System Detailed Design 7](#_heading=h.26in1rg)

[3.1 Feature: Photographer booking 7](#_heading=h.lnxbz9)

[*a.*](#_heading=h.35nkun2) *Class diagram* 7

[*b.*](#_heading=h.1ksv4uv) *Activity diagram* 7

[3.2 Feature: Photographer’s information management 7](#_heading=h.44sinio)

[*a.*](#_heading=h.2jxsxqh) *Class diagram* 7

[*b.*](#_heading=h.z337ya) *Sequence diagram (s)* 7

[3.3 Feature: Account Management 7](#_heading=h.3j2qqm3)

[*a.*](#_heading=h.1y810tw) *Class diagram* 7

[*b.*](#_heading=h.4i7ojhp) *Sequence diagram (s)* 7

[3.4 Account Management 7](#_heading=h.2xcytpi)

[*a.*](#_heading=h.1ci93xb) *Class diagram* 7

[*b.*](#_heading=h.3whwml4) *Sequence diagram (s)* 7

[3. All 7](#_heading=h.2bn6wsx)

[4. Class Specifications 8](#_heading=h.qsh70q)

[4.1 XYZ Class 8](#_heading=h.3as4poj)

[4.2 ABC Class 8](#_heading=h.1pxezwc)

[4.3 … 9](#_heading=h.49x2ik5)

[5. Data & Database Design 9](#_heading=h.2p2csry)

[5.1 Database Design 9](#_heading=h.147n2zr)

[5.2 Data File Design 13](#_heading=h.3o7alnk)

# I. Project Report

## 1. Status Report

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Work Item** | **Status** | **Notes (Work Item in Details)** |
| 1 |  | Pending |  |
| 2 |  | In Progress |  |
| 3 |  | Completed |  |

## 2. Team Involvements

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task** | **Member** | **Notes (Task Details, etc.)** |
| 1 |  | Thaott |  |
| 2 |  | Datbc |  |
| 3 |  | Kiendst |  |
| 4 |  | Mytn |  |

## 3. Issues/Suggestions

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Issue** | **Status** | **Notes (Solution, Suggestion, etc.)** |
| 1 |  | Pending |  |
| 2 |  | In Progress |  |
| 3 |  | Completed |  |

# II. Software Design Document

## 1. Overall Description

### 1.1 Assumptions

Assuming all the users for this system have basic mobile knowledge and knows how to operate the system smoothly.

The process and algorithm for calculating available schedules are assumed to be given prior.

### 1.2 Design Constraints

Hard drive space: The system will need at least 8GB storage space and minimum 1GB Storage space in order to run the booking system.

## 2. System Architecture Design

### 2.1 Overall Architecture

***Figure 1 – Overall Architecture***

PBS system is consisting of three components, the spring boot web server provides APIs, two mobile applications for customers and photographers, and the web application for admin role.

We also use several third-party services such as Aws S3 for image storage, Firebase cloud messaging for push notification function and Amazon RDS for MySQL instance.

### 2.2 System Architecture

**2.2.1 Web server architecture**

We build our web server following n-tier architecture. We decided to use this pattern for following reasons:

* It has been proven to be a scalable architecture model.
* Provide reusability, maintainability and fault tolerance.
* Helps our member to create reusable and flexible functionality.

**2.2.2 Mobile app architecture**

responses

states

data

bloc

requests

events

**Figure 2 - Mobile App Architecture**

Bloc pattern allow us to separate our mobile app into 3 layers:

* Presentation: UI
* Business Logic Component: bloc (where we can manage events – click, tap, etc. - send a request to data layer, then retrieve data from data layer and response to input from UI through states).
* Data Layer: Repository (where we can retrieve data from sources).

## 3. System Detailed Design

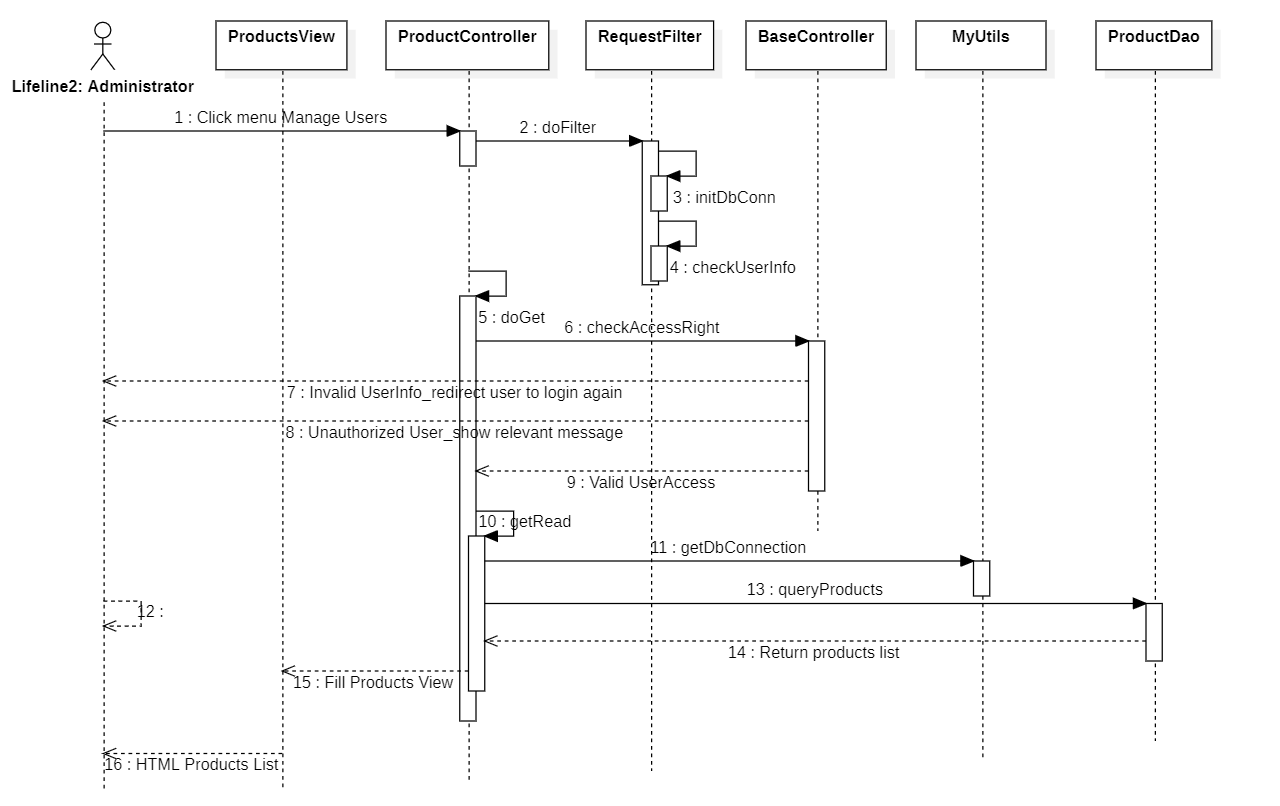
### 3.1 Overall

#### a. Class Diagram

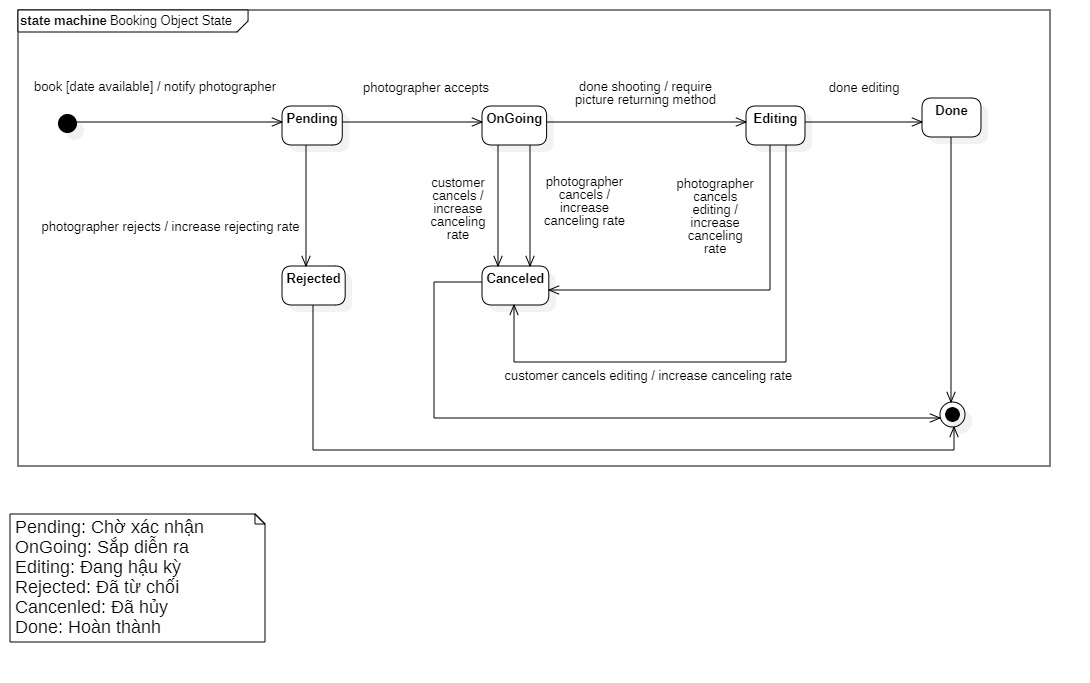
***Figure 2- Class diagram***

#### b. Sequence Diagram(s)

*[Provide the sequence diagram(s) for the feature, see the sample below]*



#### c. State Machine Diagram(s)



## 4. Class Specifications

### 4.1 User

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of a user |
| username | String | private | The attribute represents username of a user |
| password | String | private | The attribute represents password of a user |
| fullname | String | private | The attribute represents full name of a user |
| description | String | private | The attribute represents description of a user |
| avatar | String | private | The attribute represents avatar link of a user |
| phone | String | private | The attribute represents phone of a user |
| email | String | ptivate | The attribute represents email of a user |
| ratingCount | Float | private | The attribute represents the rating count of a user whose role is photographer |
| isBlocked | Boolean | private | The attribute represents if user has been blocked |
| createdAt | Date | private | The attribute represents created date of a user object |
| updatedAt | Date | private | The attribute represents updated date of a user object |
| deviceToken | String | private | The attribute represents device a token of a user object |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.2 Role

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of a role |
| role | ERole | private | The attribute represents role name |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.3 ERole

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ROLE\_CUSTOMER |  |  | Enum attribute |
| ROLE\_PHOTOGRAPHER |  |  | Enum attribute |
| ROLE\_ADMIN |  |  | Enum attribute |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.4 Package

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | private | Unique identifier of a package |
| name | String | private | The attribute represents the package name |
| price | Integer | private | The attribute represents the package price |
| supportMultiDays |  |  |  |
| description | String | private | The attribute represents the package description |
| isAvailable | Boolean | private | The attribute represents if the package still available |
| createdAt | Date | private | The attribute represents created date of a package object |
| updatedAt | Date | private | The attribute represents updated date of a package object |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.5 Service

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of a package |
| name | String | private | The attribute represents the name of a service |
| description | String | private | The attribute represents the service’s description |
| isActive | Boolean | private | The attribute represents if the service is still available |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.6 Booking

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of a booking |
| bookingStatus | EBookingStatus | private | The attribute represents the status of a booking |
| serviceName | String | private | The attribute represents the service of a booking |
| price | Integer | private | The attribute represents the service price of a booking |
| customerCancelledReason | String | private | The attribute represents the reason if customer decided to cancel a booking |
| photographerCancelledReason | String | private | The attribute represents the reason if photographer decided to cancel a booking |
| rejectedReason | String | private | The attribute represents the reason if photographer decided to reject a booking |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.7 EBookingStatus

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| Pending |  |  | Enum attribute |
| OnGoing |  |  | Enum attribute |
| Editing |  |  | Enum attribute |
| Rejected |  |  | Enum attribute |
| Cancelled |  |  | Enum attribute |
| Done |  |  | Enum attribute |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.8 Album

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of album |
| name | String | private | The attribute represents the name album |
| thumbnail | String | private | The attribute represents the album’s thumbnail link |
| description | String | private | The attribute represents the album’s description |
| createdAt | Date | private | The attribute represents the created date of album object |
| isActive | Boolean | private | The attribute represents if album still active |
| likes | Integer | private | The attribute represents the album like count |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.9 Image

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of image |
| description | String | private | The attribute represents the description of album |
| imageLink | String | private | The attribute represents the album’s image link |
| comment | String | private | The attribute represents the album’s comment |
| createdAt | Date | private | The attribute represents the created date of image object |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.10 Category

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of category |
| category | String | private | The attribute represents the category name |
| iconLink | String | private | The attribute represents the category icon link |
| createdAt | Date | private | The attribute represents the created date of category object |
| updatedAt | Date | private | The attribute represents the updated date of category object |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.11 DayOfWeek

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of day of week |
| day | Int | private | The attribute represents the day of photographer’s working time |
| startTime | Date | private | The attribute represents the start time of photographer’s working time |
| endTime | Date | private | The attribute represents the end time of photographer’s working time |
| isWorkingDay | Boolean | private | The attribute represents the working day type of a calendar |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.12 Location

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of location |
| formattedAddress | String | private | The attribute represents the address name |
| longitude | String | private | The attribute represents the longitude of a location |
| latitude | Double | private | The attribute represents the latitude of a location |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.13 BookingComment

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of booking comment |
| comment | String | private | The attribute represents the comment of a user |
| rating | Float | private | The attribute represents the rating of a user |
| commentedAt | Date | private |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.14 BookingDetail

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of booking detail |
| serviceName | String | private | The attribute represents the service name of a user |
| serviceDescription | String | private | The attribute represents the service description of a user |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.15 BusyDay

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of busy day |
| startDate | Date | private | The attribute represents the start date of photographer’s working time |
| endDate | Date | private | The attribute represents the end date of photographer’s working time |
| title | String | private | The attribute represents the title of photographer’s working time |
| description | String | private | The attribute represents the description of a calendar |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.16 Thread

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of thread |
| title | String | private | The attribute represents the title of thread |
| content | String | private | The attribute represents the content of thread |
| createdAt | Date | private | The attribute represents the create of thread |
| updateAt | Date | private | The attribute represents the update of thread |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.17 ThreadTopic

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of thread topic |
| topic | String | private | The attribute represents the topic name of thread |
| createdBy | String | private | The attribute represents the create by of thread |
| createdAt | Date | private | The attribute represents the create of thread |
| updateAt | Date | private | The attribute represents the update of thread |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.18 ThreadComment

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of thread comment |
| comment | String | private | The attribute represents the comment of a user |
| createdAt | Date | private | The attribute represents the create of a user |
| updateAt | Date | private | The attribute represents the update of a user |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.19 ReturningType

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of returning type |
| type | EReturningType | private | The attribute represents the type of a type |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

### 4.20 EReturningType

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| LINK |  |  | Enum attribute |
| OFFLINE |  |  | Enum attribute |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

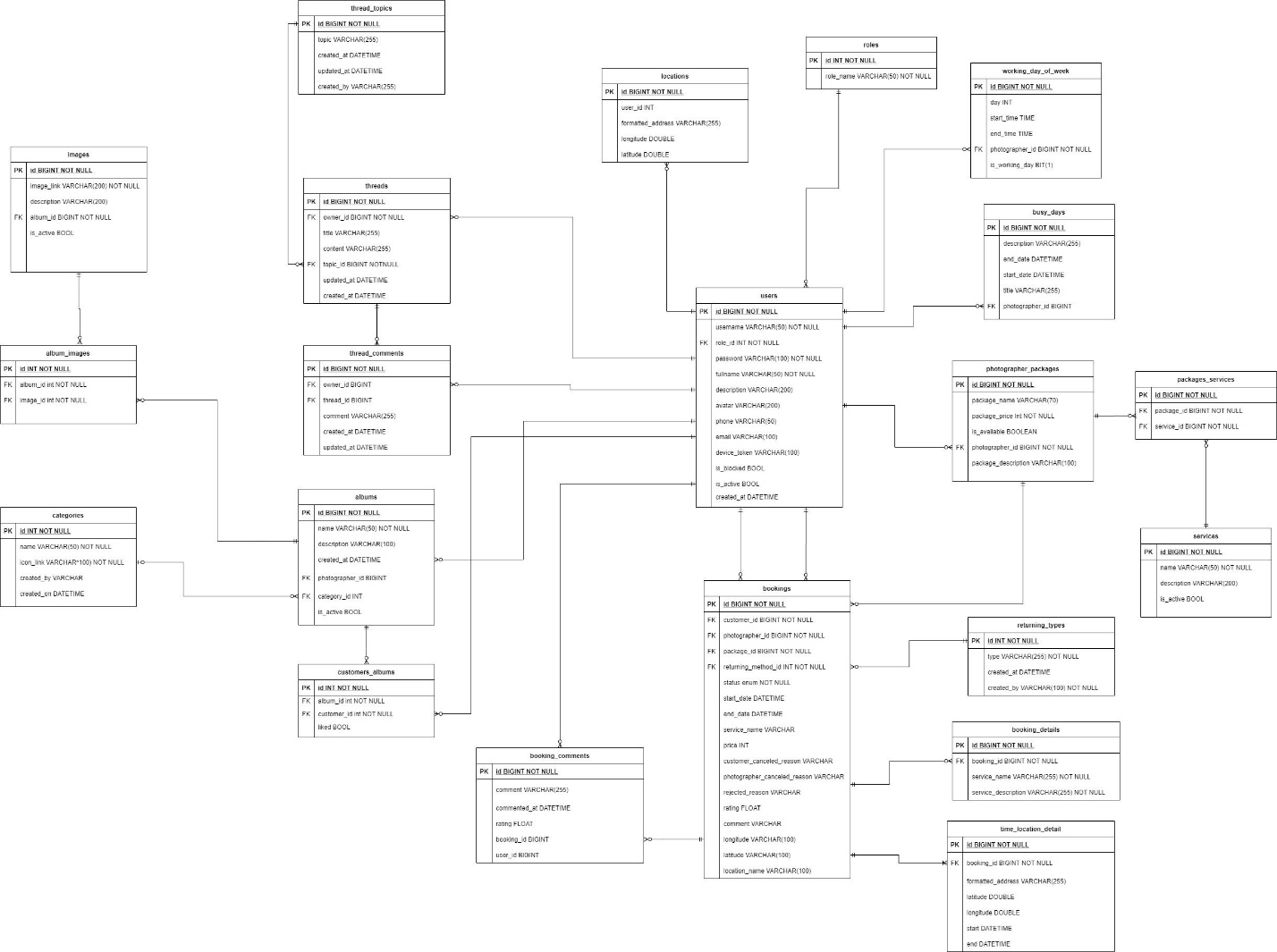
### 4.21 TimeLocationDetail

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Long | private | Unique identifier of time location detail |
| lat | Double | private | The attribute represents the latitude of a user |
| lon | Double | private | The attribute represents the longitude of a user |
| formattedAddress | String | private | The attribute represents the formate address of a user |
| start | Date | private | The attribute represents the start of a location |
| end | Date | private | The attribute represents the end of a location |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Method** | **Visibility** | **Return Type** | **Description** |
| 1 | getter | public | Attribute Type | get attribute value |
| 2 | setter | public | void | set attribute value |

## 5. Data & Database Design

### 5.1 Database Design



***Figure 3 – Database design***

#### User

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | username | varchar | 50 | X | X |  |  |
| 3 | password | varchar | 100 |  | X |  |  |
| 4 | fullname | varchar | 50 |  | X |  |  |
| 5 | description | varchar | 200 |  |  |  |  |
| 6 | avatar | varchar | 200 |  |  |  | Avatar image link of user |
| 7 | phone | varchar | 50 |  |  |  |  |
| 8 | email | varchar | 100 |  |  |  |  |
| 9 | device\_token | varchar | 100 |  |  |  | The firebase token of device, using for firebase messaging. |
| 9 | is\_blocked | boolean |  |  |  |  |  |
| 10 | is\_active | boolean |  |  |  |  |  |
| 11 | created\_at | datetime |  |  |  |  |  |
| 12 | role\_id | int |  |  | X | X |  |

#### Role

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | int |  | X | X | X |  |
| 2 | role\_name | varchar | 50 |  | X |  |  |

#### Photographer-Package

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | package\_name | varchar | 70 |  | X |  |  |
| 3 | package\_price | int |  |  | X |  |  |
| 4 | is\_available | boolean |  |  |  |  |  |
| 5 | package \_description | varchar | 100 |  |  |  |  |
| 6 | photographer \_id | int |  |  | X | X |  |

#### Package-Service

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | package\_id | int |  |  | X | X |  |
| 3 | service\_id | int |  |  | X | X |  |

#### Service

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | name | varchar | 50 |  | X |  |  |
| 3 | description | varchar | 200 |  |  |  |  |
| 4 | is\_active | boolean |  |  |  |  |  |

#### Booking

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | status | enum |  |  | X |  | There will be 6 status: Pending, Ongoing, Editing, Done, Rejected, Canceled |
| 3 | start\_date | datetime |  |  | X |  |  |
| 4 | end\_date | datetime |  |  | X |  |  |
| 5 | service\_name | varchar | 100 |  | X |  | Package’s name |
| 6 | price | int |  |  | X |  | Package’s price |
| 7 | customer  \_canceled  \_reason | varchar | 100 |  |  |  | In case customers cancel and provide their reason |
| 8 | photographer \_canceled \_reason | varchar | 100 |  |  |  | In case photographers cancel and provide their reason |
| 9 | rejected\_reason | varchar | 100 |  |  |  | In case photographers reject the request and provide reason |
| 10 | rating | float |  |  |  |  | After booking is done, customer can rate the booking |
| 11 | comment | varchar | 100 |  |  |  | After booking is done, customer can comment on the booking |
| 12 | longitude | varchar | 100 |  |  |  |  |
| 13 | latitude | varchar | 100 |  |  |  |  |
| 14 | location\_name | varchar | 100 |  |  |  | The shooting location picked by customer |
| 15 | customer\_id | int |  |  | X | X |  |
| 16 | photographer \_id | int |  |  | X | X |  |
| 17 | package\_id | int |  |  | X | X |  |

#### Album

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | name | varchar | 50 |  | X |  |  |
| 3 | description | varchar | 100 |  |  |  |  |
| 4 | created\_at | datetime |  |  |  |  |  |
| 5 | is\_active | boolean |  |  |  |  |  |
| 6 | photographer \_id | int |  |  | X | X |  |
| 7 | category\_id | int |  |  |  | X |  |

#### Locations

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | Formatted\_address | varchar | 255 |  | X |  |  |
| 3 | longitude | double | 100 |  | X |  |  |
| 4 | latitude | double | 100 |  | X |  |  |
| 5 | user\_id | int |  |  | X | X |  |

#### Category

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | int |  | X | X | X |  |
| 2 | name | varchar | 50 |  | X |  |  |
| 3 | icon\_link | varchar | 100 |  | X |  | Link for the icon of category |
| 4 | created\_by | varchar | 100 |  |  |  | Name of the admin who created the category |
| 5 | Created\_on | datetime |  |  |  |  |  |

#### Album-Images

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | int |  | X | X | X |  |
| 2 | album\_id | int |  |  | X | X |  |
| 3 | image\_id | int |  |  | X | X |  |

#### Images

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | image\_link | varchar | 200 |  | X |  |  |
| 3 | description | varchar | 200 |  |  |  |  |
| 4 | is\_active | boolean |  |  |  |  |  |
| 5 | album\_id | int |  |  | X | X |  |

#### Customer-Album

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | int |  | X | X | X |  |
| 2 | liked | boolean |  |  |  |  | represent if customer has liked the album |
| 3 | album\_id | int |  |  | X | X |  |
| 4 | customer\_id | int |  |  | X | X |  |

#### Working-day-of-week

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | day | int |  |  |  |  |  |
| 3 | start\_time | datetime |  |  |  |  |  |
| 4 | end\_time | datetime |  |  |  |  |  |
| 5 | is\_working\_day | bit |  |  |  |  |  |
| 6 | photographer \_id | bigint |  |  | X | X |  |

#### Busy-day

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | description | varchar | 255 |  |  |  |  |
| 3 | start\_date | datetime |  |  |  |  |  |
| 4 | end\_date | datetime |  |  |  |  |  |
| 5 | title | varchar | 255 |  |  |  |  |
| 6 | photographer \_id | bigint |  |  | X | X |  |

#### Returning-type

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | int |  | X | X | X |  |
| 2 | type | varchar | 255 |  | X |  |  |
| 3 | created\_at | datetime |  |  |  |  |  |
| 4 | created\_by | varchar | 100 |  | X |  |  |

#### Booking-detail

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | service\_name | varchar | 255 |  | X |  |  |
| 3 | service\_description | varchar | 255 |  | X |  |  |
| 4 | booking\_id | bigint |  |  | X | X |  |

#### Time-location-detail

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | formatted\_address | varchar | 255 |  |  |  |  |
| 3 | laitude | double |  |  |  |  |  |
| 4 | longitude | double |  |  |  |  |  |
| 5 | start | datetime |  |  |  |  |  |
| 6 | end | datetime |  |  |  |  |  |
| 7 | booking\_id | bigint |  |  | X | X |  |

#### Booking-comment

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | comment | varchar | 255 |  |  |  |  |
| 3 | commented\_at | datetime |  |  |  |  |  |
| 4 | rating | float |  |  |  |  |  |
| 5 | user\_id | bigint |  |  |  |  |  |
| 6 | booking\_id | bigint | 100 |  |  |  |  |

#### Thread

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | title | varchar | 255 |  |  |  |  |
| 3 | content | varchar | 255 |  |  |  |  |
| 4 | update\_at | datetime |  |  |  |  |  |
| 5 | created\_at | datetime |  |  |  |  |  |
| 6 | topic\_id | bigint |  |  | X | X |  |
| 7 | owner\_id | bigint |  |  | X | X |  |

#### Thread-comment

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | comment | varchar | 255 |  |  |  |  |
| 3 | update\_at | datetime |  |  |  |  |  |
| 4 | created\_at | datetime |  |  |  |  |  |
| 5 | thread\_id | bigint |  |  |  | X |  |
| 6 | owner\_id | bigint |  |  |  | X |  |

#### Thread-topic

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | id | bigint |  | X | X | X |  |
| 2 | topic | varchar | 255 |  |  |  |  |
| 3 | created\_by | varchar | 255 |  |  |  |  |
| 4 | update\_at | datetime |  |  |  |  |  |
| 5 | created\_at | datetime |  |  |  |  |  |

### 5.2 Data File Design

We use Aws S3 to store files and images.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **File Name** | **Type** | **Notes** |
| 1 | User avatar | Image | path: “/{userId}/{userId}-avatar” |
| 2 | Photographer album thumbnail | Image | path: “/{ptgId}/{ptgId}/{albumid}/{albumId}-thumbnail” |
| 3 | Photographer album image | Image | path: “/{ptgId}/{ptgId}/{albumid}/{albumId}/{imageId}-image” |
| 4 | Photographer cover image | Image | path: “/{ptgId}/{ptgId}-cover” |
| 5 | Category icon image | Image | path: “/categories/{categoryId}-category” |

## 6. Algorithm

### 6.1 Multiple-factors weighted sorting

#### 6.1.1 Definition

Multiple-factors weighted ranking considered to be simple yet effective algorithm.

#### 6.1.2 Define problem

A suggestion method is always needed in every kind of commercial applications. Since it gives the customers an objective way to decide whether a photographer is trustable or not.

It also boosts the photographers’ motivation to give the best of them in order to connect to more customers.

#### 6.1.3 Solution

**Step 1: Choose suitable factors**

We will rank the photographers based on the following factors:

* Customers’ ratings.
* Photographer’s average service price.
* Distance (from photographer’s working location to the booked location).

**Step 2: Specify weight for each factor**

Admin can change the weights.

|  |  |
| --- | --- |
| **Factor** | **Weight** |
| Rating | 0.3 |
| Distance | 0.3 |
| Price | 0.4 |

**Step 3: Set a common scale for factors**

Common scale = from 0 (worst) to 100 (best)

**Step 4: Transform each object’s factors score to the common scale**

* Rating: 1 rating point on scale of 5 will equal 20 points on scale of 100
* Distance:

Calculate total distances of all photographers in the list

Calculate score of a photographer’s distance factor = (1 – distance / sum) \* max scale

(photographers with higher distance will get lower score)

* Photographer’s average service price

Calculate total prices of all photographers in the list

Calculate score of a photographer’s price factor = (1 – price / sum) \* max scale

(Photographer with higher price will get lower score)

**Step 5: Sum the scores**

Score of a photographer = 0.3 \* rating score + 0.3 \* distance score + 0.3 \* price score

**Step 6: Sort photographers based on the score.**

#### 6.1.4 Complexity

The complexity of this algorithm is O(n) for finding n photographers.

#### 6.1.5 Flowchart