

**DENTIST BOOKING SYSTEM**

**Software Design Document**

– Tp Ho Chi Minh, August 2019 –

**Table of Contents**

[I. Overview 3](#_heading=h.gjdgxs)

[1. Code Packages 3](#_heading=h.30j0zll)

[2. Database Schema 3](#_heading=h.1fob9te)

[II. Code Designs 4](#_heading=h.3znysh7)

[1. <Feature/Function Name1> 4](#_heading=h.2et92p0)

[a. Class Diagram 4](#_heading=h.tyjcwt)

[b. Class Specifications 4](#_heading=h.3dy6vkm)

[c. Sequence Diagram(s) 4](#_heading=h.1t3h5sf)

[d. Database queries 5](#_heading=h.4d34og8)

[2. <Feature/Function Name2> 5](#_heading=h.2s8eyo1)

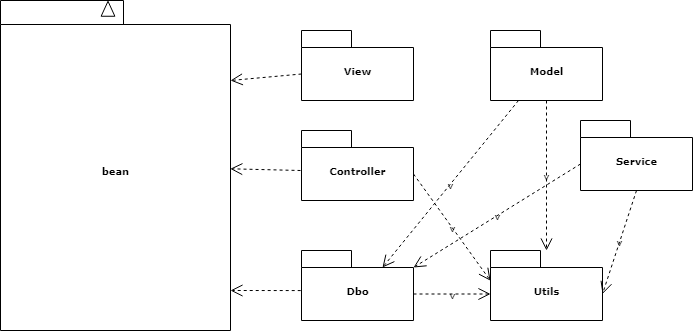
[III. Database Tables 5](#_heading=h.17dp8vu)

[1. <Table name 1> 5](#_heading=h.3rdcrjn)

[2. <Table name 2…> 5](#_heading=h.26in1rg)

# I. Overview

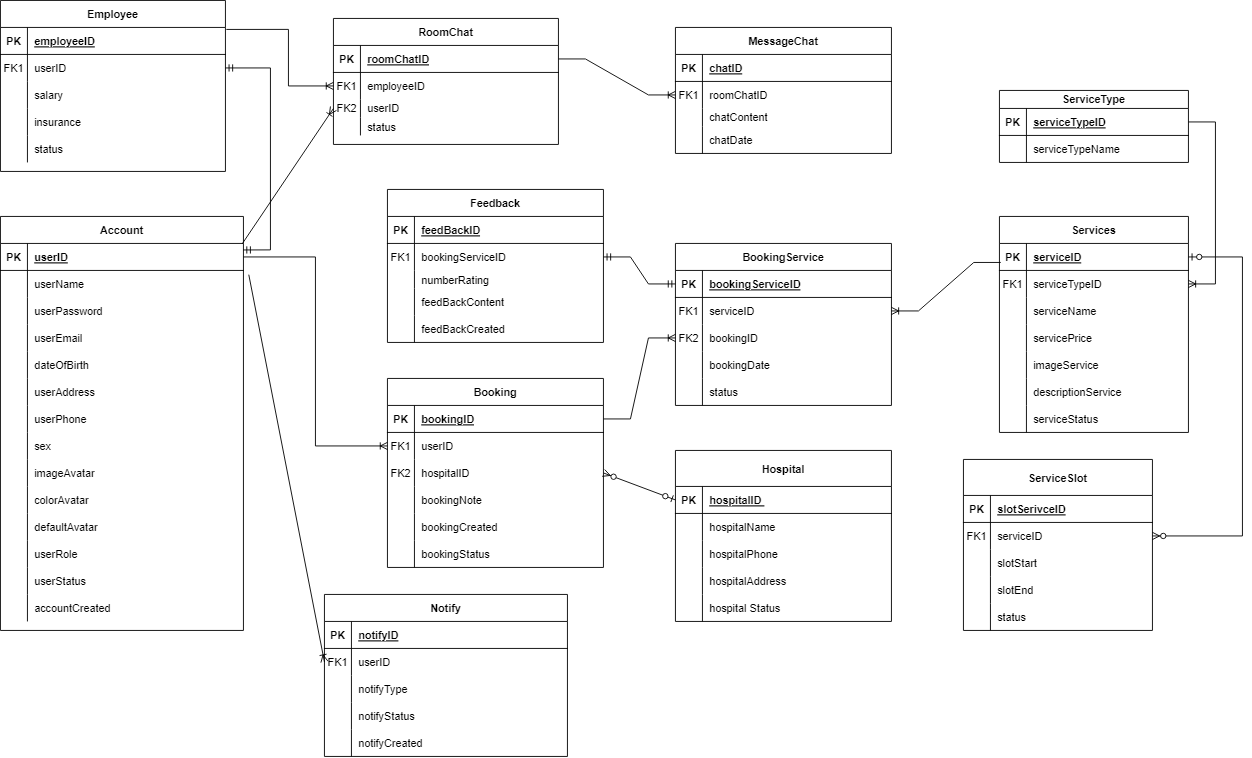
## 1. Code Packages/Namespaces



***Package descriptions & package class naming conventions***

|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| *01* | *View* | *Where to store jsp files about the user interface*  *<Class naming convention>* |
| *02* | *Model* | *Where to store all model file* |
| *03* | *Controller* | *Where to store all controller servlet*  *<Class naming convention>* |
| *04* | *Dbo* | *Where to store all dbo of the service and user.* |
| *05* | *Utils* | *Where to store the connection which database.*  *<Class naming convention>* |
| *06* | *Service* | *Where to store all service for the service and user.* |

## 2. Database Schema



***Table descriptions & package class naming conventions are as below***

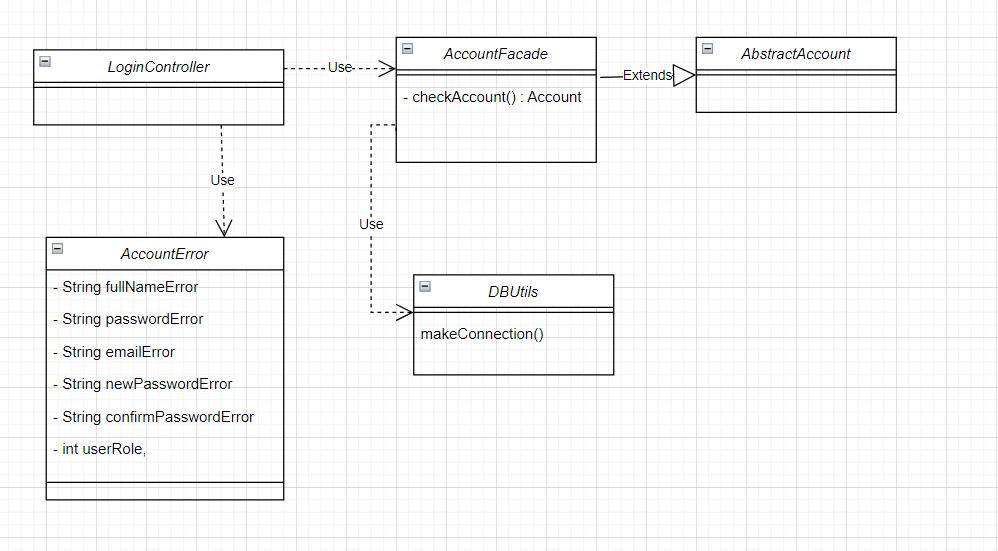
|  |  |  |
| --- | --- | --- |
| **No** | **Table** | **Description** |
| *01* | *Account* | *Save account as well as customer information it also saves employee information.*  *- Primary keys: UserID* |
| *02* | *Notify* | *Where to store notices for customers*  *- Primary keys: NotifyID*  *- Foreign keys: UserID* |
| *03* | *Booking* | *Store information about the booking.*  *- Primary keys: BookingID*  *- Foreign keys: UserID*  *- Foreign keys: HospitalID* |
| *04* | *Hospital* | *Store information about the hospital.*  *- Primary keys: HospitalID* |
| *05* | *BookingService* | *Store information about the booking details.*  *- Primary keys: BookingServiceID*  *- Foreign keys: ServiceID*  *- Foreign keys: BookingID*  *- Foreign keys: SlotServiceID* |
| *06* | *Services* | *Store information about the services of the system.*  *- Primary keys: ServiceID*  *- Foreign keys: ServiceTypeID* |
| *07* | *SerivcesTypes* | *Store information about the type of service.*  *- Primary keys: ServiceTypeID* |
| *08* | *ServiceSlot* | *Store information about the slot of each service.*  *- Primary keys: SlotServiceID*  *- Foreign keys: ServiceID*  *- Foreign keys: SlotID* |
| *09* | *Slot* | *Store information about the slot that available in the clinic.*  *- Primary keys: SlotID* |
| *10* | *Feedback* | *Store the feedback of the customer about our service.*  *-Primary keys: FeedBackID*  *-Foreign keys:BookingService* |
| *11* | *Employee* | *Store information about the employee*  *-Primary keys: EmployeeID*  *-Foreign keys : UserID* |
| *12* | *RoomChat* | *The room chat between Employee and Customer*  *-Primary keys: RoomChatID*  *-Foreign keys: EmployeeID*  *-Foreign keys: UserID* |
| *13* | *Message* | *Store all the message chat between Employee and Customer*  *-Primary keys :ChatID*  *-Foreign keys :RoomChatID* |

# 

# II. Code Designs

## 1. Authentication /Login

### a. Class Diagram

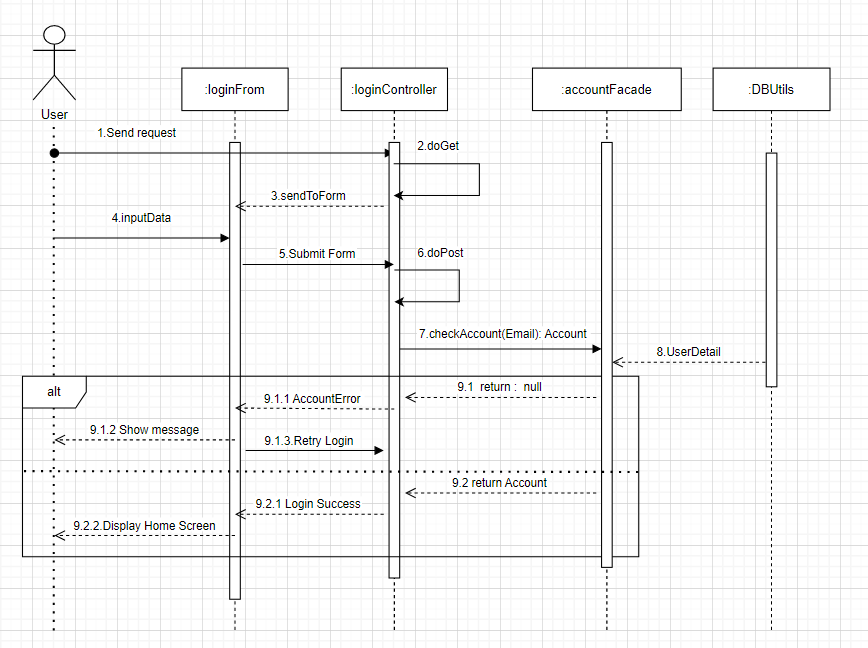


### b. Class Specifications

#### AccountFacade

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getInfoAccountFromSQL* | *get information of account, input ResultSet resultSet, output Account* |
| *02* | *registerAccount* | *create account, input (Connection connection, Account account), output String* |
| *03* | *updateAccount* | *update account, input (Connection connection, Account account, Object object),*  *output (true or false)* |
| *04* | *getAccount* | *get account, input (Connection connection, Object object, Object action, Object status),*  *output (List<Account>)* |
| *05* | *checkAccount* | *check account, input (Connection connection, Object object),*  *output (Account)* |
| *06* | *countAccount* | *count account, input (Connection connection),*  *output (int)* |

### c. Sequence Diagram(s)



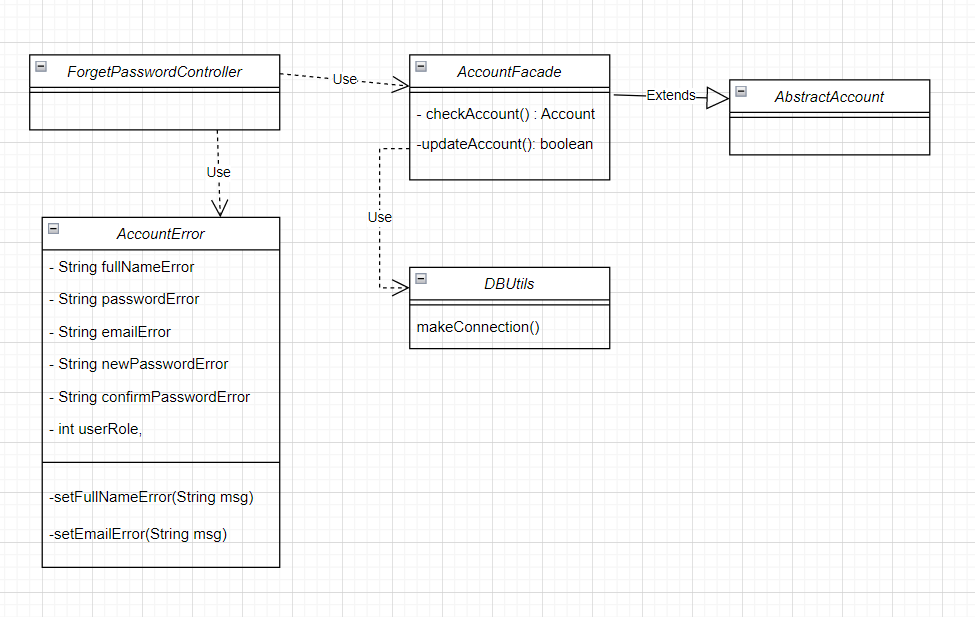
### d. Database queries

checkAccount(Email) : Account

+ "SELECT \* FROM Account WHERE UserEmail = ?"

## 2. Authentication / Forget Password

### a. Class Diagram

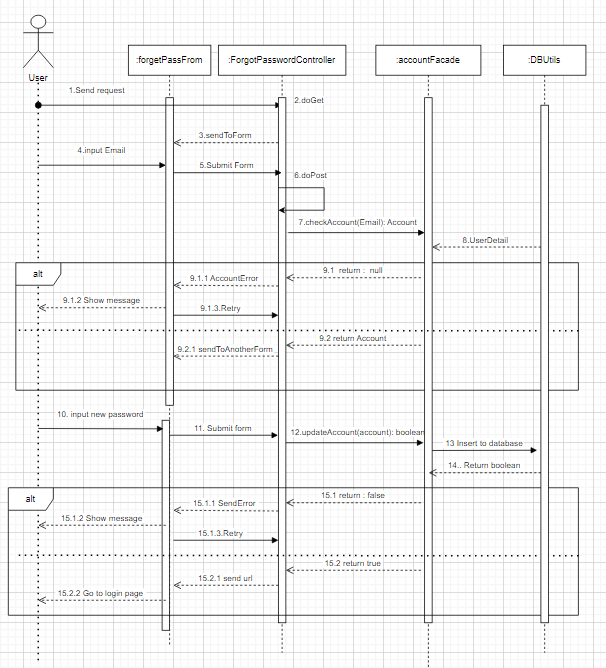


### b. Class Specifications

#### AccountFacade

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getInfoAccountFromSQL* | *get information of account, input ResultSet resultSet, output Account* |
| *02* | *registerAccount* | *create account, input (Connection connection, Account account), output String* |
| *03* | *updateAccount* | *update account, input (Connection connection, Account account, Object object),*  *output (true or false)* |
| *04* | *getAccount* | *get account, input (Connection connection, Object object, Object action, Object status),*  *output (List<Account>)* |
| *05* | *checkAccount* | *check account, input (Connection connection, Object object),*  *output (Account)* |
| *06* | *countAccount* | *count account, input (Connection connection),*  *output (int)* |

### c. Sequence Diagram(s)



### d. Database queries

checkAccount(Email) : Account

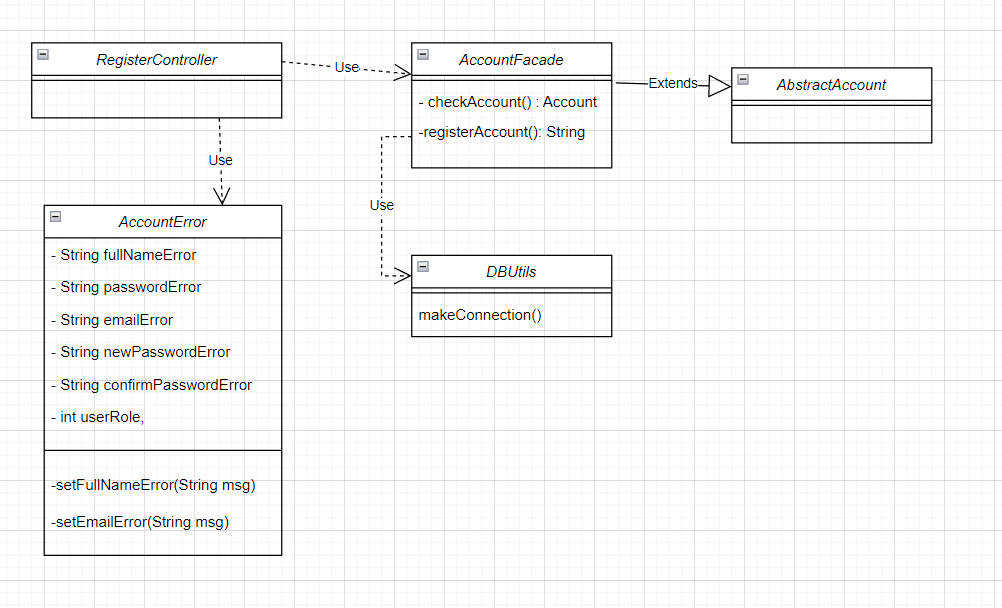
+ "SELECT \* FROM Account WHERE UserEmail = ?"

UpdateAccount(Account): Boolean

+ "UPDATE Account SET UserPassword = ? WHERE UserEmail = ?";

## 3. Authentication /Register

### a. Class Diagram

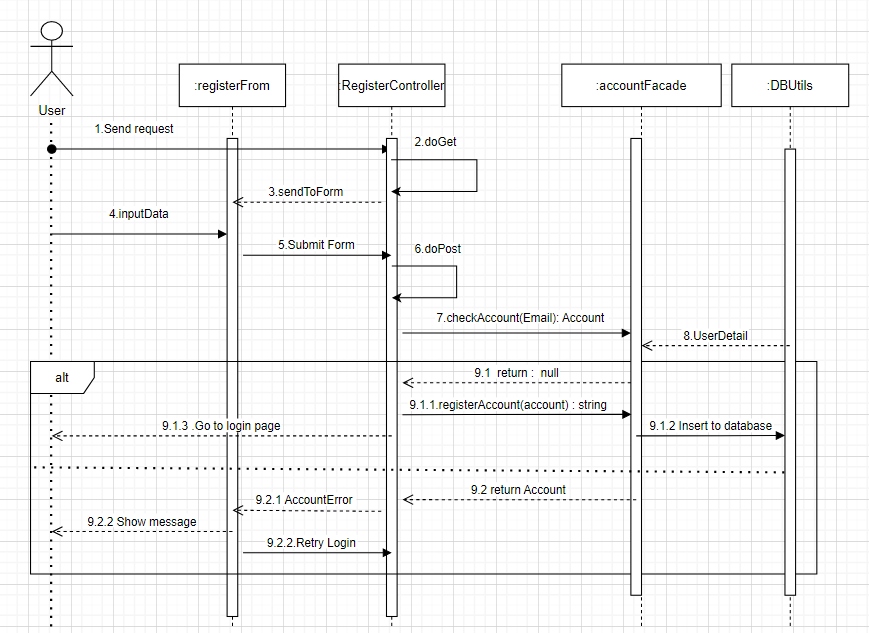


### b. Class Specifications

#### AccountFacade

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getInfoAccountFromSQL* | *get information of account, input ResultSet resultSet, output Account* |
| *02* | *registerAccount* | *create account, input (Connection connection, Account account), output String* |
| *03* | *updateAccount* | *update account, input (Connection connection, Account account, Object object),*  *output (true or false)* |
| *04* | *getAccount* | *get account, input (Connection connection, Object object, Object action, Object status),*  *output (List<Account>)* |
| *05* | *checkAccount* | *check account, input (Connection connection, Object object),*  *output (Account)* |
| *06* | *countAccount* | *count account, input (Connection connection),*  *output (int)* |

### c. Sequence Diagram(s)



### d. Database queries

checkAccount(Email) : Account

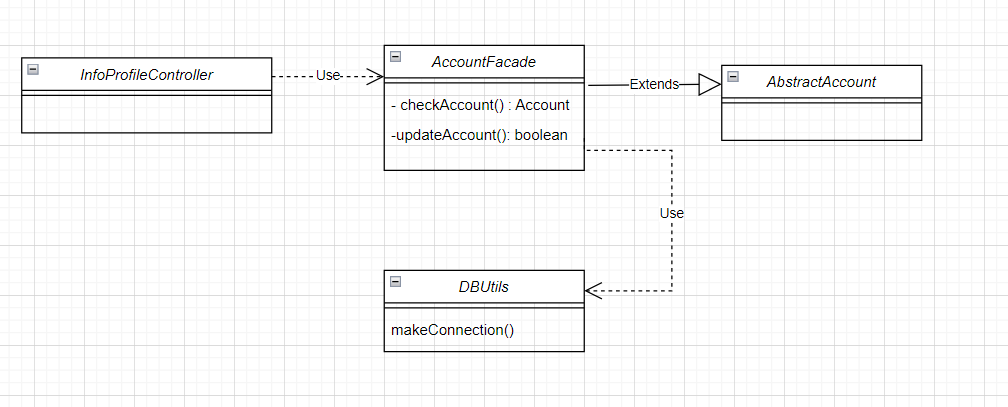
+ "SELECT \* FROM Account WHERE UserEmail = ?";

registerAccount(Account) : String

+"INSERT INTO Account(UserID, FullName, UserPassword, UserEmail, DefaultAvatar, ColorAvatar) VALUES(?, ?, ?, ?, ?, ?)";

## 4. User Information /View Profile

### a. Class Diagram

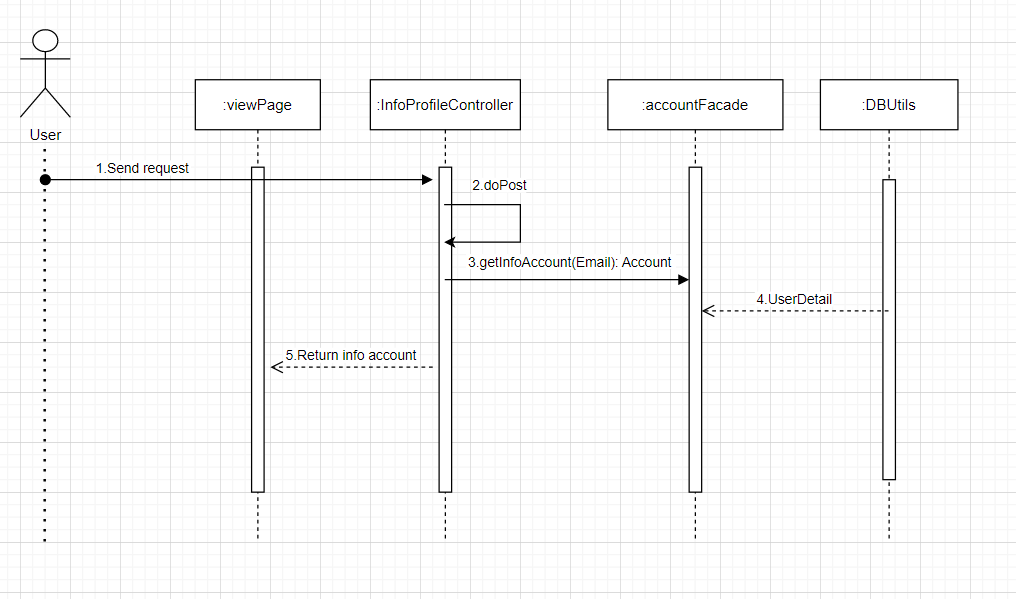


### b. Class Specifications

#### AccountFacade

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getInfoAccountFromSQL* | *get information of account, input ResultSet resultSet, output Account* |
| *02* | *registerAccount* | *create account, input (Connection connection, Account account), output String* |
| *03* | *updateAccount* | *update account, input (Connection connection, Account account, Object object),*  *output (true or false)* |
| *04* | *getAccount* | *get account, input (Connection connection, Object object, Object action, Object status),*  *output (List<Account>)* |
| *05* | *checkAccount* | *check account, input (Connection connection, Object object),*  *output (Account)* |
| *06* | *countAccount* | *count account, input (Connection connection),*  *output (int)* |

### c. Sequence Diagram(s)



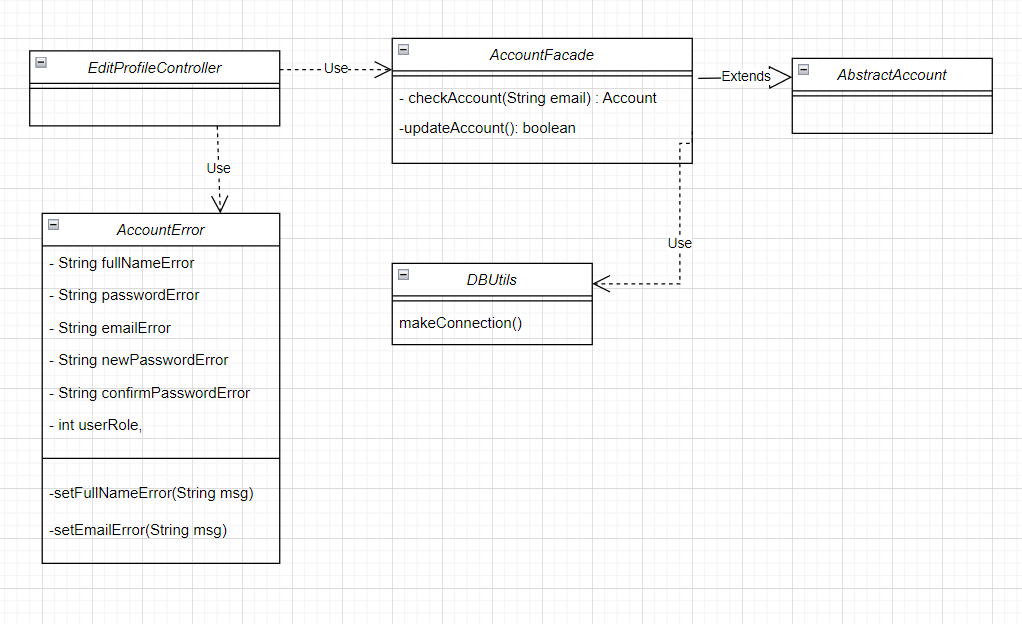
### d. Database queries

getInfoAccount(Email): Account

"SELECT \* FROM Account WHERE UserEmail = ?";

## 5. User Information /Edit View Profile

### a. Class Diagram



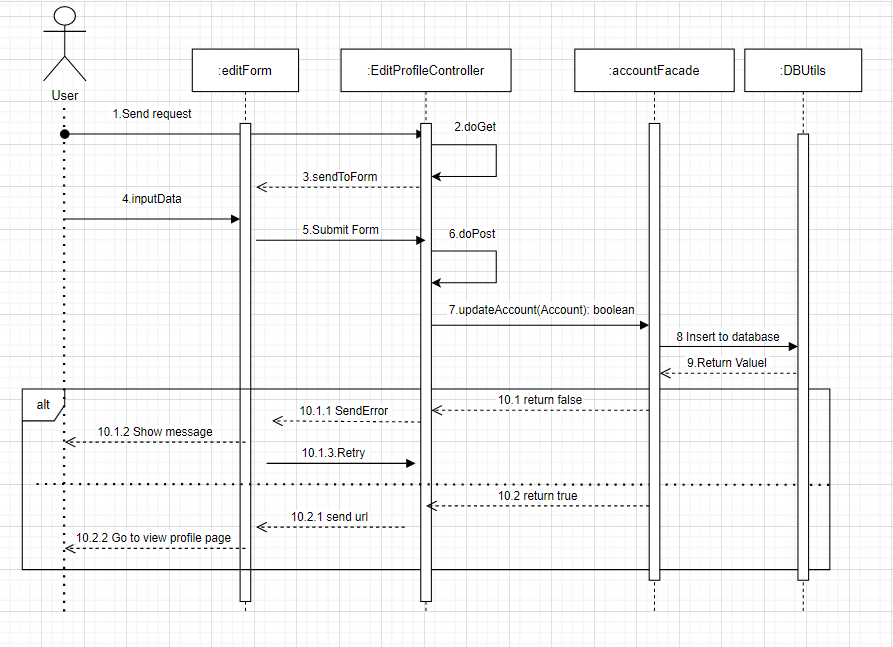
### b. Class Specifications

#### AccountFacade

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getInfoAccountFromSQL* | *get information of account, input ResultSet resultSet, output Account* |
| *02* | *registerAccount* | *create account, input (Connection connection, Account account), output String* |
| *03* | *updateAccount* | *update account, input (Connection connection, Account account, Object object),*  *output (true or false)* |
| *04* | *getAccount* | *get account, input (Connection connection, Object object, Object action, Object status),*  *output (List<Account>)* |
| *05* | *checkAccount* | *check account, input (Connection connection, Object object),*  *output (Account)* |
| *06* | *countAccount* | *count account, input (Connection connection),*  *output (int)* |

### 

### c. Sequence Diagram(s)



### 

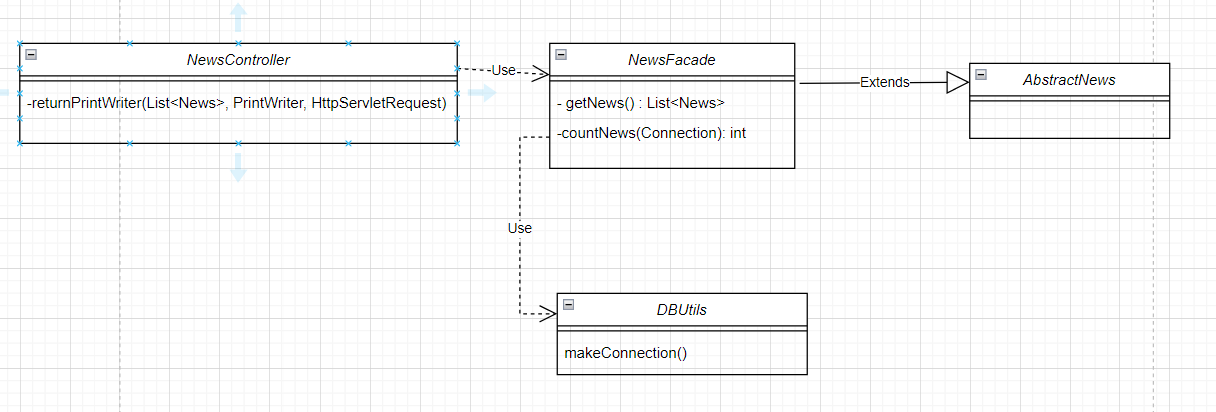
### d. Database queries

UpdateAccount(Account): Boolean

+ "UPDATE Account SET UserPassword = ? WHERE UserEmail = ?";

## 6. News Management / View News information

### a. Class Diagram



### b. Class Specifications

#### NewsDetailFacade

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getInfoNewsDetailFromSQL* | *get News Detail, input (ResultSet resultSet),*  *output (NewsDetail)* |
| *02* | *getNewsDetail* | *add new News Detail, input (Connection connection, NewsDetail newsDetail), output (boolean)* |
| *03* | *addNewsDetail* | *update News Detail, input (Connection connection, NewsDetail newsDetail), output (boolean)* |
| *04* | *updateNewsDetail* | *update News Detail, input (Connection connection, NewsDetail newsDetail), output (boolean)* |

### c. Sequence Diagram(s)



### d. Database queries

getNews( int AmountInt) : List<News>

+ "SELECT TOP 8 \* FROM News WHERE PostDate < GETDATE() ORDER BY PostDate DESC";

## 2. <Feature/Function Name1>

*[Provide the detailed design for the function <Feature/Function Name1>. It include Class Diagram, Class Specifications, and Sequence Diagram(s)]*

### a. Class Diagram

*[This part presents the class diagram for the relevant feature]*



### b. Class Specifications

*[Provide the description for each class and the methods in each class, following the table format as below]*

#### XYZ Class

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *<method name>* | *<Description of the method, including the inputs, outputs & internal method processing>* |
|  |  |  |

#### ABC Class

***Class Methods***

*[Provide the detailed description for the class methods]*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *<method name>* | *<Description of the method, including the inputs, outputs & internal method processing>* |
|  |  |  |

### c. Sequence Diagram(s)

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



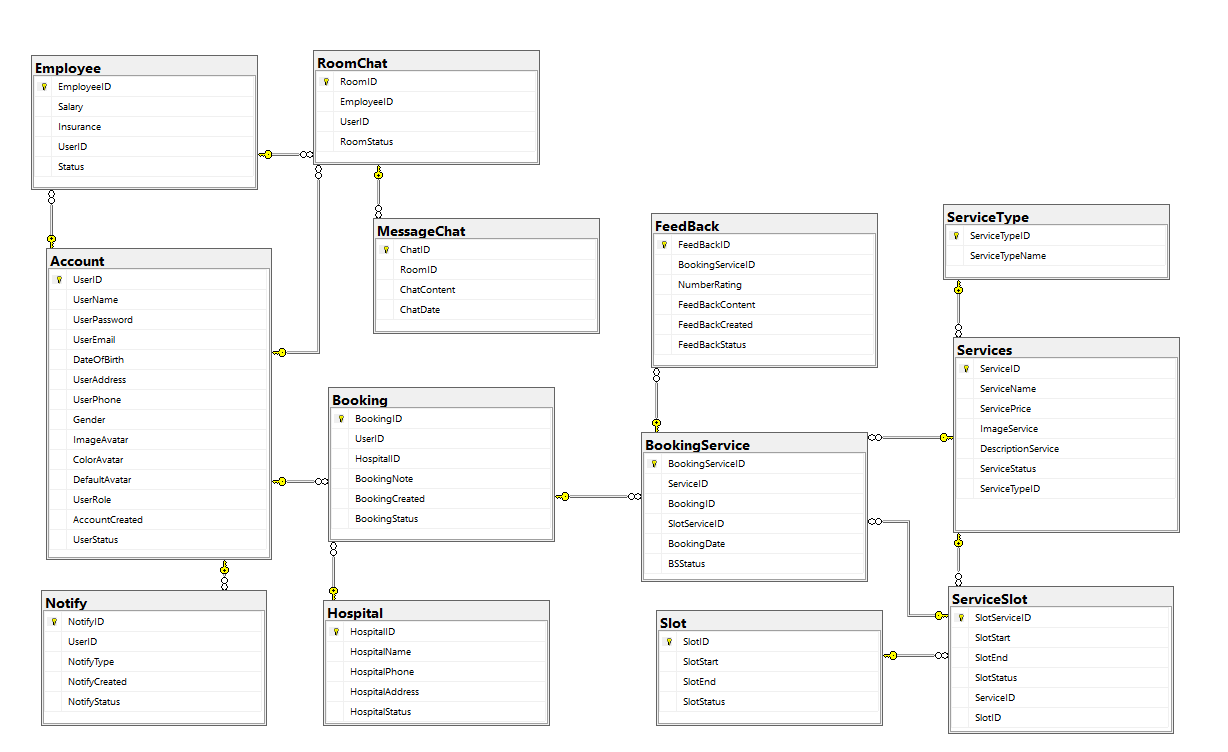
### d. Database queries

*[Provide the detailed SQL (select, insert, update...) which are used in implementing the function/screen]*

## 2. <Feature/Function Name2>

…

# III. Database Tables



## 1. Employee

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Field name | Type | Size | Unique | Not Null | PK/FK | Notes |
| 1 | EmployeeID | VARCHAR | 10 |  |  | PK |  |
| 2 | Salary | INT |  |  |  |  |  |
| 3 | Insurance | INT |  |  |  |  |  |
| 4 | UserID | VARCHAR | 10 | x |  | FK |  |

## 2. Account

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Field name | Type | Size | Unique | Not Null | PK/FK | Notes |
| 1 | UserID | VARCHAR | 10 |  |  | PK |  |
| 2 | FullName | NVARCHAR | 100 |  | x |  |  |
| 3 | UserPassword | VARCHAR | 50 |  | x |  |  |
| 4 | UserEmail | NVARCHAR | 50 |  | x |  |  |
| 5 | DateOfBirth | DATE |  |  |  |  |  |
| 6 | UserAddress | NVARCHAR | 150 |  |  |  |  |
| 7 | UserPhone | CHAR | 10 |  |  |  |  |
| 8 | Gender | CHAR | 1 |  |  |  |  |
| 9 | ImageAvatar | VARBINARY | MAX |  |  |  |  |
| 10 | ColorAvatar | VARCHAR | 10 |  |  |  |  |
| 11 | DefaultAvatar | CHAR | 1 |  |  |  |  |
| 12 | UserRole | INT |  |  |  |  | DEFAULT '0' |
| 13 | UserStatus | INT |  |  |  |  | DEFAULT '0' |
| 14 | AccountCreated | DATETIME |  |  |  |  | DEFAULT CURRENT\_TIMESTAMP |

## 3. RoomChat

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Field name | Type | Size | Unique | Not Null | PK/FK | Notes |
| 1 | RoomID | VARCHAR | 10 |  |  | PK |  |
| 2 | EmployeeID | VARCHAR | 10 |  |  | FK |  |
| 3 | UserID | VARCHAR | 10 |  |  | FK |  |
| 4 | RoomStatus | INT |  |  |  |  | DEFAULT '0' |

## 4. Feedback

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Field name | Type | Size | Unique | Not Null | PK/FK | Notes |
| 1 | FeedBackID | VARCHAR | 10 |  |  | PK |  |
| 2 | BookingServiceID | VARCHAR | 10 |  |  | FK |  |
| 3 | NumberRating | INT |  |  |  |  |  |
| 4 | FeedBackContent | NVARCHAR | Max |  |  |  |  |
| 5 | FeedBackStatus | INT |  |  |  |  | INT DEFAULT '0' |
| 6 | FeedBackCreated | DATETIME |  |  |  |  | DEFAULT CURRENT\_TIMESTAMP |

## 5. Booking

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Field name | Type | Size | Unique | Not Null | PK/FK | Notes |
| 1 | BookingID | VARCHAR | 10 |  |  | PK |  |
| 2 | UserID | VARCHAR | 10 |  |  | FK |  |
| 3 | HospitalID | VARCHAR | 10 |  |  | FK |  |
| 4 | BookingNote | NVARCHAR | Max |  |  |  |  |
| 5 | BookingCreated | DATETIME |  |  |  |  | DEFAULT CURRENT\_TIMESTAMP |
| 6 | BookingStatus | INT |  |  |  |  | DEFAULT '0' |

## 6. Notify

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Field name | Type | Size | Unique | Not Null | PK/FK | Notes |
| 1 | NotifyID | VARCHAR | 10 |  |  | PK |  |
| 2 | UserID | VARCHAR | 10 |  |  | FK |  |
| 3 | NotifyType | NVARCHAR | 10 |  | x |  |  |
| 4 | NotifyStatus | INT |  |  |  |  | DEFAULT '0' |
| 5 | NotifyCreated | DATETIME |  |  |  |  | DEFAULT CURRENT\_TIMESTAMP |

## 

## 7. MessageChat

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Field name | Type | Size | Unique | Not Null | PK/FK | Notes |
| 1 | ChatID | VARCHAR | 10 |  |  | PK |  |
| 2 | RoomID | VARCHAR | 10 |  |  | FK |  |
| 3 | ChatContent | NVARCHAR | MAX |  |  |  |  |
| 4 | ChatDate | DATETIME |  |  |  |  | DEFAULT CURRENT\_TIMESTAMP |

## 8. Hospital

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Field name | Type | Size | Unique | Not Null | PK/FK | Notes |
| 1 | HospitalID | VARCHAR | 10 |  |  | PK |  |
| 2 | HospitalName | NVARCHAR | 150 |  | x |  |  |
| 3 | HospitalPhone | CHAR | 10 |  | x |  |  |
| 4 | HospitalAddress | NVARCHAR | 150 |  | x |  |  |
| 5 | HospitalStatus | INT |  |  |  |  | DEFAULT '0' |

## 9. BookingService

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | BookingServiceID | VARCHAR | 10 |  |  | PK |  |
| 2 | ServiceID | VARCHAR | 10 |  |  | FK |  |
| 3 | BookingID | VARCHAR | 10 |  |  | FK |  |
| 4 | SlotServiceID | VARCHAR | 10 |  |  | FK |  |
| 5 | BookingDate | DATE |  |  |  |  |  |
| 6 | BSStatus | INT |  |  |  |  | DEFAULT '0' |

## 10. Services

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | ServiceID | VARCHAR | 10 |  |  | PK |  |
| 2 | ServiceName | NVARCHAR | 150 |  | x |  |  |
| 3 | ServicePrice | INT |  |  |  |  | DEFAULT '0' |
| 4 | ImageService | VARBINARY | MAX |  |  |  |  |
| 5 | DescriptionService | NTEXT |  |  |  |  |  |
| 6 | ServiceStatus | INT |  |  |  |  | DEFAULT '0' |
| 7 | ServiceTypeID | VARCHAR | 10 |  |  | FK |  |

## 

## 

## 

## 11. Slot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | SlotID | VARCHAR | 10 |  |  | PK |  |
| 2 | SlotStart | TIME |  |  |  |  |  |
| 3 | SlotEnd | TIME |  |  |  |  |  |
| 4 | SlotStatus | INT |  |  |  |  | DEFAULT '0' |

## 12. ServiceType

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | ServiceTypeID | VARCHAR | 10 |  |  | PK |  |
| 2 | ServiceTypeName | NVARCHAR | 150 |  | X |  |  |

## 13. ServiceSlot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | SlotServiceID | VARCHAR | 10 |  |  | PK |  |
| 2 | SlotStart | TIME |  |  |  |  |  |
| 3 | SlotEnd | TIME |  |  |  |  |  |
| 4 | SSlotStatus | INT |  |  |  |  | DEFAULT '0' |
| 5 | ServiceID | VARCHAR | 10 |  |  | FK |  |
| 6 | SlotID | VARCHAR | 10 |  |  | FK |  |

## 