

**ONLINE LEARNING**

**Software Design Specification**

– Hanoi, June 2023 –

**Record of changeS**

| **Date** | **A\* M, D** | **In charge** | **Change Description** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

\*A - Added M - Modified D - Deleted

**Table of Contents**

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

[1. Code Packages 4](#_heading=h.30j0zll)

[2. Database Design 4](#_heading=h.1fob9te)

[a. Database Schema 4](#_heading=h.3znysh7)

[b. Table Description 4](#_heading=h.2et92p0)

[II. Code Designs 5](#_heading=h.tyjcwt)

[1. <Feature/Function Name1> 5](#_heading=h.3dy6vkm)

[a. Class Diagram 5](#_heading=h.1t3h5sf)

[b. Class Specifications 5](#_heading=h.4d34og8)

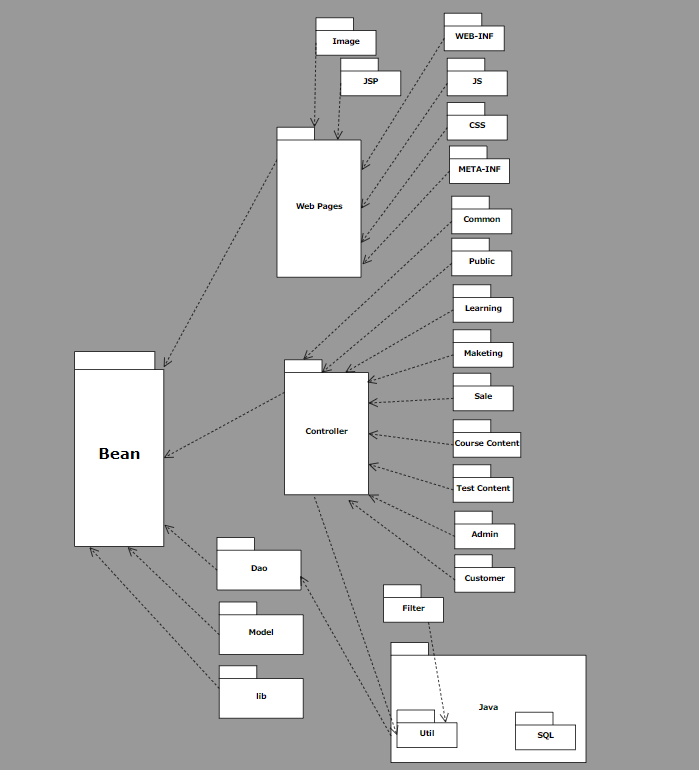
[c. Sequence Diagram(s) 5](#_heading=h.2s8eyo1)

[d. Database queries 6](#_heading=h.17dp8vu)

[2. <Feature/Function Name2> 6](#_heading=h.3rdcrjn)

# I. Overview

## 1. Code Packages



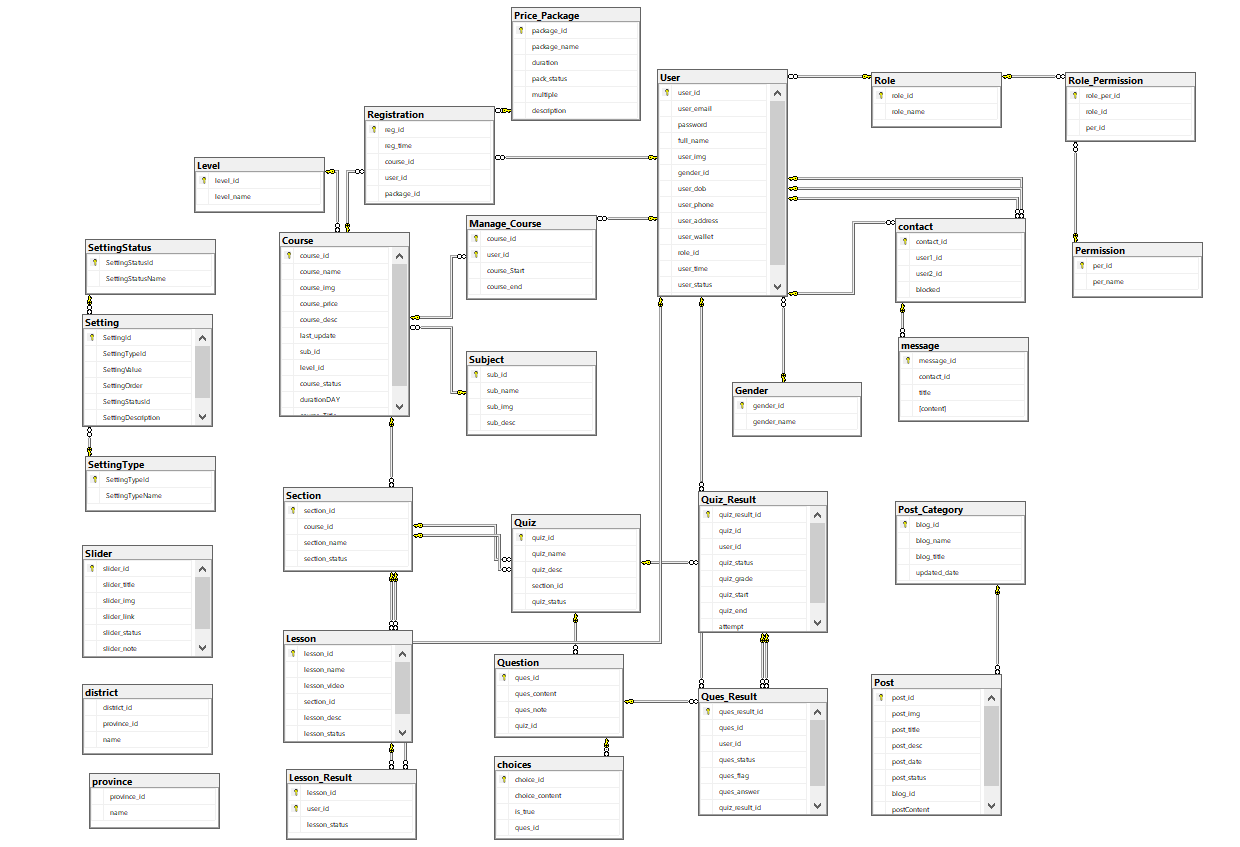
***Package descriptions***

| **No** | **Package** | **Description** |
| --- | --- | --- |
| *01* | *bean* | *bean is a package containing the main modules of a program.* |
| *02* | *model* | *This is the data layer that contains the business logic of the system, and also*  *represents the state of the application. It’s independent of the presentation*  *layer, the controller fetches the data from the model layer and sends it to the*  *view layer.* |
| *03* | *Web Pages* | *The view layer represents the output of the application, usually some form*  *of UI. The presentation layer is used to display the Model data fetched by the*  *Controller.* |
| *04* | *controller* | *The controller layer acts as an interface between View and Model. It receives*  *requests from the View layer and processes them, including the necessary*  *validations.* |
| *05* | *filter* | *Package filter is a firewall technique used to control network access by*  *monitoring outgoing and incoming packets and allowing them to pass or halt*  *based on the source and destination Internet Protocol (IP) addresses,*  *protocols and ports.* |
| *06* | *dao* | *The dao layer will communicate with the storage system, the database*  *management system, such as performing jobs related to data storage and*  *query (search, add, delete, edit, ...).* |
| *07* | *entity* |  |
| *08* | *css* |  |
| *09* | *img* |  |

## 2. Database Design

### a. Database Schema

*[Provide the tables relationship like example below – following MySQL database naming convention]*



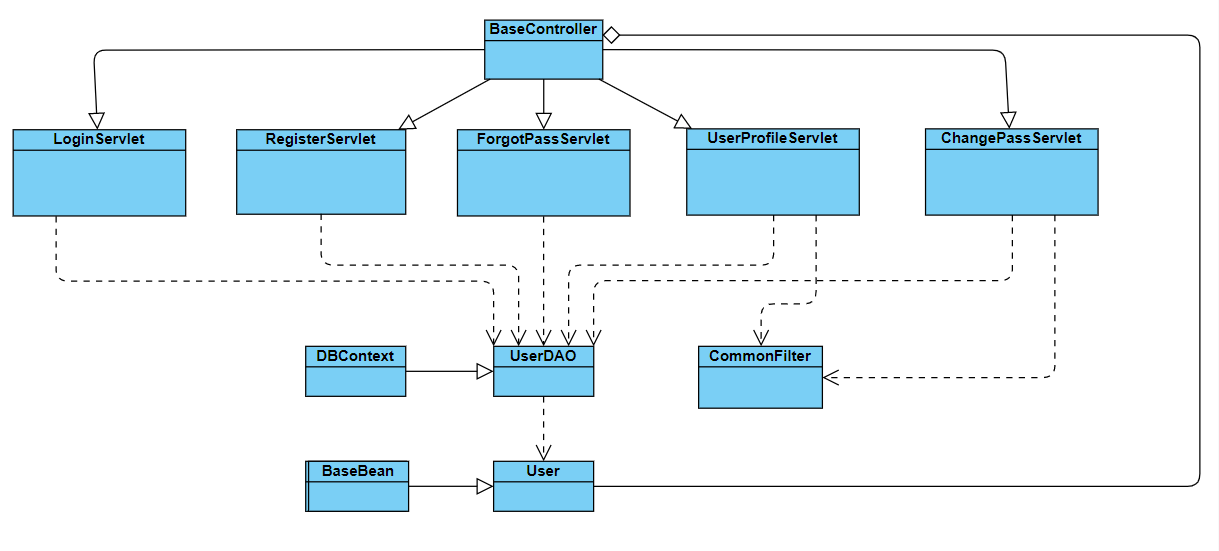
### b. Table Description

| **No** | **Table** | **Description** |
| --- | --- | --- |
| *01* | User | Contains information about the user |
| *02* | Course | Contains information about the course |
| *03* | Manage Course | Shows the relationship between users who study which course |
| *04* | Choices | Show possible choice for each question |
| *05* | Contact | Shows the relationship between users have contact with each other |
| *06* | District | Contains names of each province’s districts |
| *07* | Province | Contains names of every province in Vietnam |
| *08* | Gender | Contains genders |
| *09* | Lesson | Contains Lesson information |
| *10* | Lesson\_Result | Contains Lesson status is finish or not |
| *11* | Level | Contains level of course |
| *12* | Message | Messages of a contact |
| *13* | Permission | Possible Permission for each role |
| *14* | Post | Contains information about post |
| *15* | Post\_Category | Sort post in category |
| *16* | Price\_Pakage | Contains information about possible payment package |
| *17* | Ques\_Result | Contains result of question every time user attempt quiz |
| *18* | Question | Contains list of question for each quiz |
| *19* | Quiz | Contains quiz information |
| *20* | Quiz\_Result | Save result of each quiz attempts |
| *21* | Registration | Contains customer registration for course |
| *22* | Role | Contains role in website |
| *23* | Role\_Permission | Show connection between role and permission |
| *24* | Section | Each course have multiple section |
| *25* | Setting | Contains information about setting |
| *26* | Setting\_Status | Show setting status |
| *27* | Setting\_Type | Contains setting type |
| *28* | Slider | Contains information about slider display in homepage |
| *29* | Subject | Each subject contains many course |

# II. Code Designs

## 1. Common feature

### a. Class Diagram



### b. Class Specifications

#### UserDAO Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *getAll()* | *Get all user’s information* |
|  | *checkUser(String email)* | *Check user (exists in the database or not) by email* |
|  | *checkUser(String email, String*  *pass)* | *Check user (exists in the database or not) by email and*  *password* |
|  | *getRole(User u)* | *Get the user's role* |
|  | *getUserByUid(int uId)* | *Get user’s information by user id* |
|  | *updateUser(User u, int uid)* | *Update user’s information by user id* |
|  | *updateNewPass(String email,*  *String newpass)* | *Update new password* |
|  | *updateUserPassword(User u)* | *Update new password* |
|  | *Encryption(String password)* | *Encrypting password to save in database* |
|  | *insertUser(User ca)* | *Register new account* |

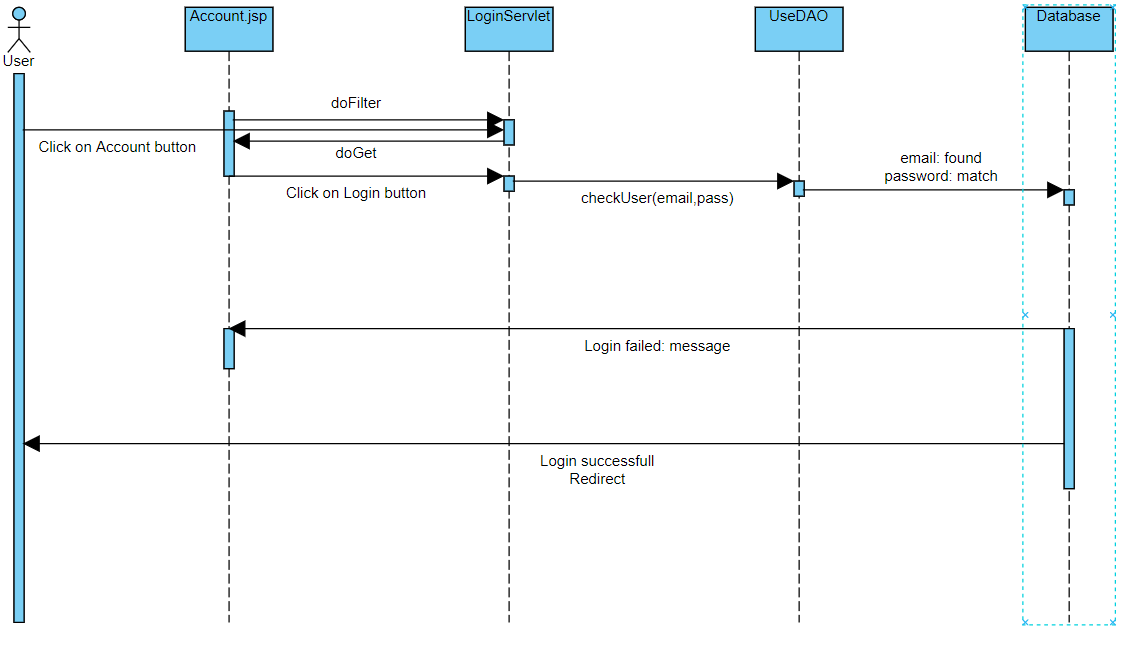
#### ABC Class

***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)



### d. Database Queries

*select \* from [User] where user\_email=?*

*select \* from [User] where user\_email=? and password=?*

*select r.role\_name from [User] u join [Role] r on u.role\_id = r.role\_id\n" where u.user\_id = ?*

*select \* from [User] where user\_id=?*

*update [User] set user\_email = ?, full\_name = ?, user\_img = ?, gender\_id = ? , user\_dob = ?,*

*user\_phone = ?, user\_address = ?, user\_wallet= ? where user\_id = ?*

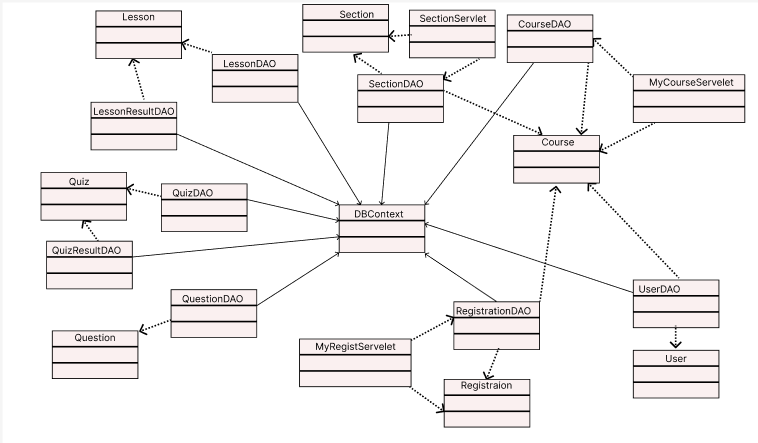
*update [User] set password = ? where user\_email = ?*

*Update [User] set [password] = ? where user\_email like ?*

*insert into [User](user\_email, password, full\_name, gender\_id, role\_id) values(?,?,?,?,?)*

## 3. Learning Feature

### a. Class Diagram



### b. Class Specifications

#### UserDAO Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *getAll()* | *Get all user’s information* |
|  | *checkUser(String email)* | *Check user (exists in the database or not) by email* |
|  | *checkUser(String email, String*  *pass)* | *Check user (exists in the database or not) by email and*  *password* |
|  | *getRole(User u)* | *Get the user's role* |
|  | *getUserByUid(int uId)* | *Get user’s information by user id* |
|  | *updateUser(User u, int uid)* | *Update user’s information by user id* |
|  | *updateNewPass(String email,*  *String newpass)* | *Update new password* |
|  | *updateUserPassword(User u)* | *Update new password* |
|  | *Encryption(String password)* | *Encrypting password to save in database* |
|  | *insertUser(User ca)* | *Register new account* |

#### PostDAO Class

#### 

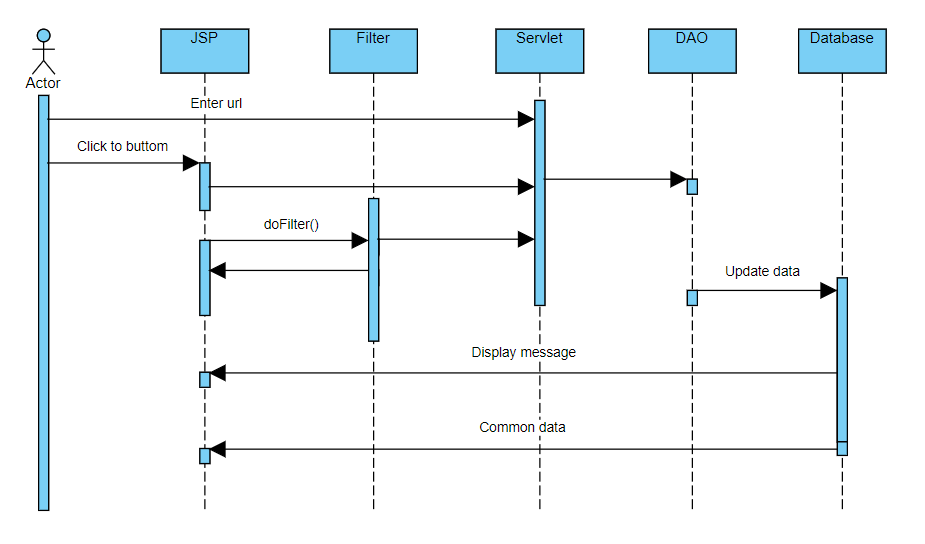
| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *getAll()* | *get all posts* |
| *02* | *getTop3Post ()* | *specify top 3 posts* |
| *03* | *getPostByBlogID()* | *get all posts by blog Id* |

#### BlogDAO Class

#### 

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *getAll()* | *get all posts* |
|  | *getBlogById* | *get blog by blog id* |
|  | *getPostByBlogID()* | *get all posts by blog Id* |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

### c. Sequence Diagram(s)



### d. Database Queries

select \* from Course

select \* from Course where course\_id like ?

select \* from Course where sub\_id = ?

select post\_id,post\_img,post\_title,post\_desc,post\_date,post\_status,p.blog\_id from Post p inner

join Blog b on p.blog\_id=b.blog\_id order by post\_date desc

select top 3 post\_id,post\_img,post\_title,post\_desc,post\_date,post\_status,p.blog\_id from Post p

inner join Blog b on p.blog\_id=b.blog\_id order by post\_date desc

select \* from Post where post\_title like ?

select \* from Blog where blog\_id = ?

select \* from Blog

select \* from Lecturer

select \* from Lecturer where lecturer\_id = ?

select \* from Level

select \* from Subject

select \* from Course

select \* from Course where course\_id like ?

select \* from Course where sub\_id = ?

select post\_id,post\_img,post\_title,post\_desc,post\_date,post\_status,p.blog\_id from Post p inner

join Blog b on p.blog\_id=b.blog\_id order by post\_date desc

select top 3 post\_id,post\_img,post\_title,post\_desc,post\_date,post\_status,p.blog\_id from Post p

inner join Blog b on p.blog\_id=b.blog\_id order by post\_date desc

select \* from Post where post\_title like ?

select \* from Post\_Category where blog\_id = ?

select \* from Post\_Category

select \* from Lecturer

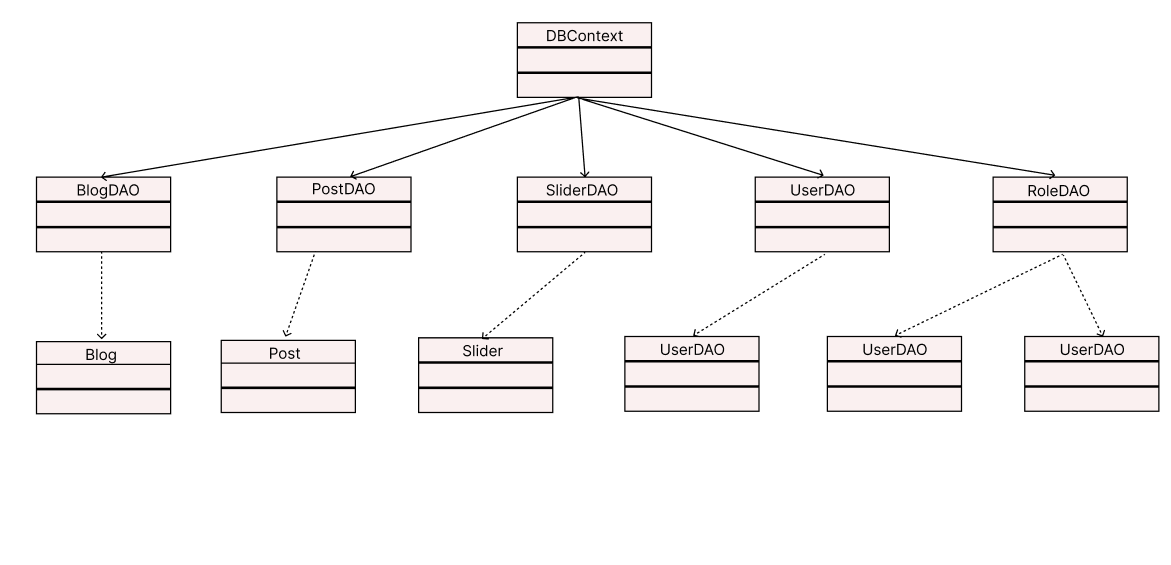
select \* from Lecturer where lecturer\_id = ?

Select \* from Level

select \* from Subject

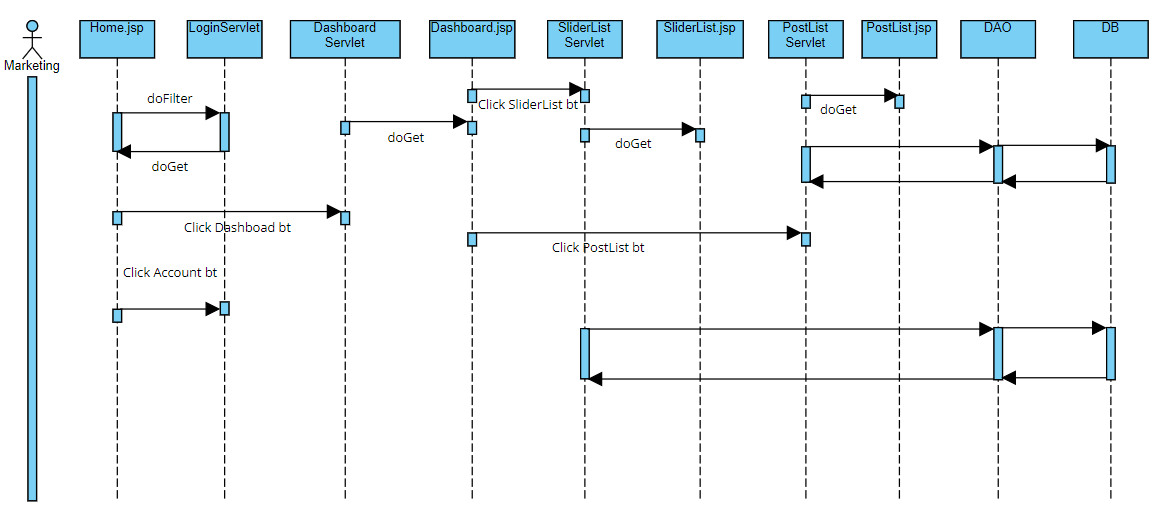
## 4. Marketing Feature

### a. Class Diagram



### b. Class Specifications

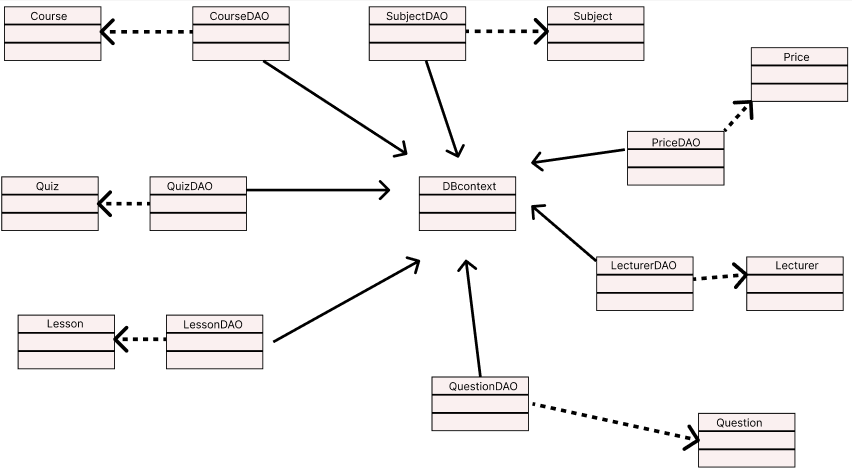
### c. Sequence Diagram(s)



### d. Database Queries

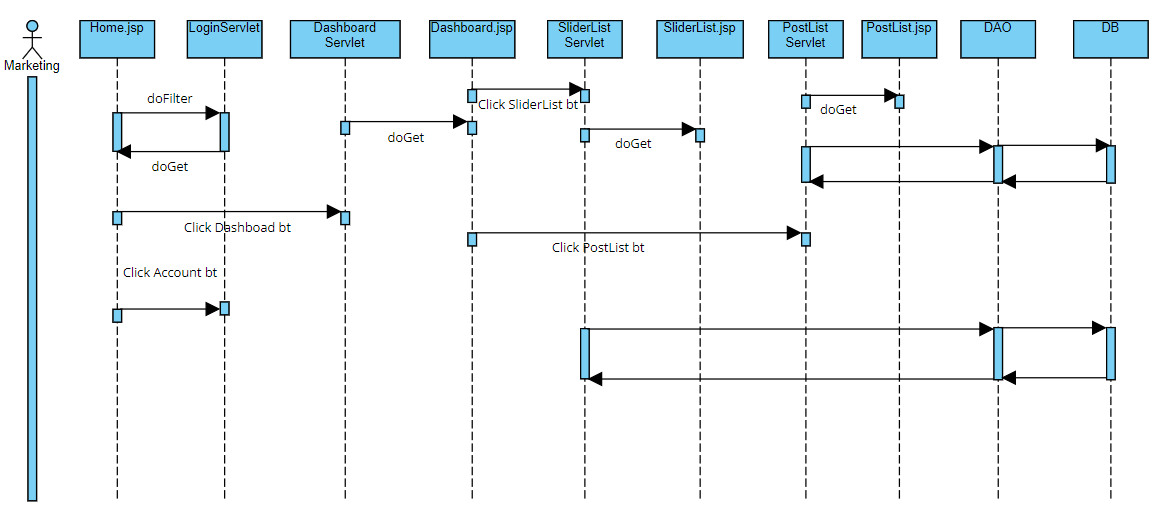
## 5. Course content

### a. Class Diagram



### b. Class Specifications

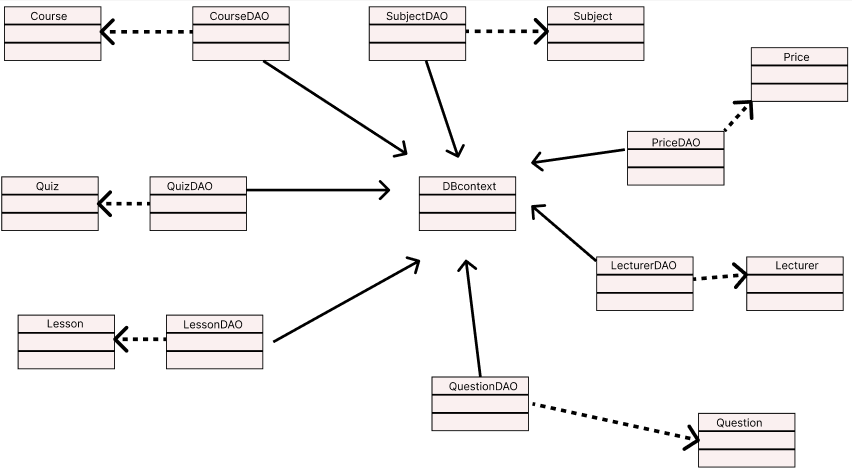
### c. Sequence Diagram(s)



### d. Database Queries

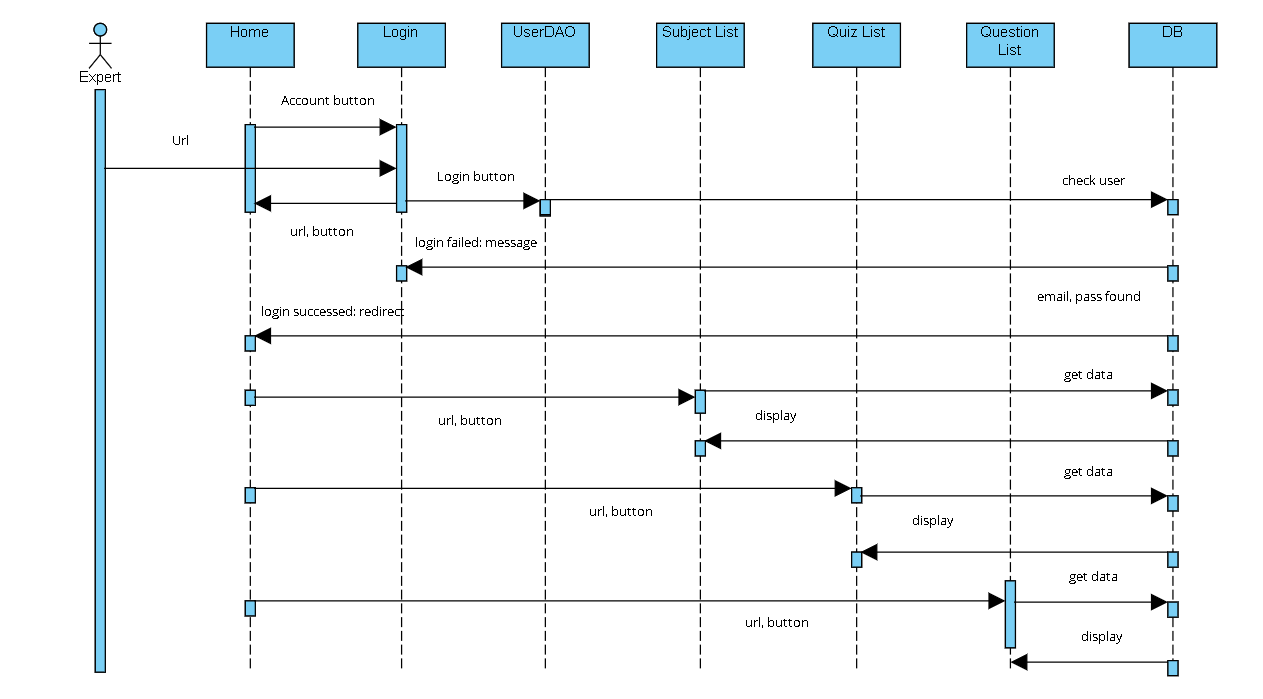
## 6. Test content

### a. Class Diagram



### b. Class Specifications

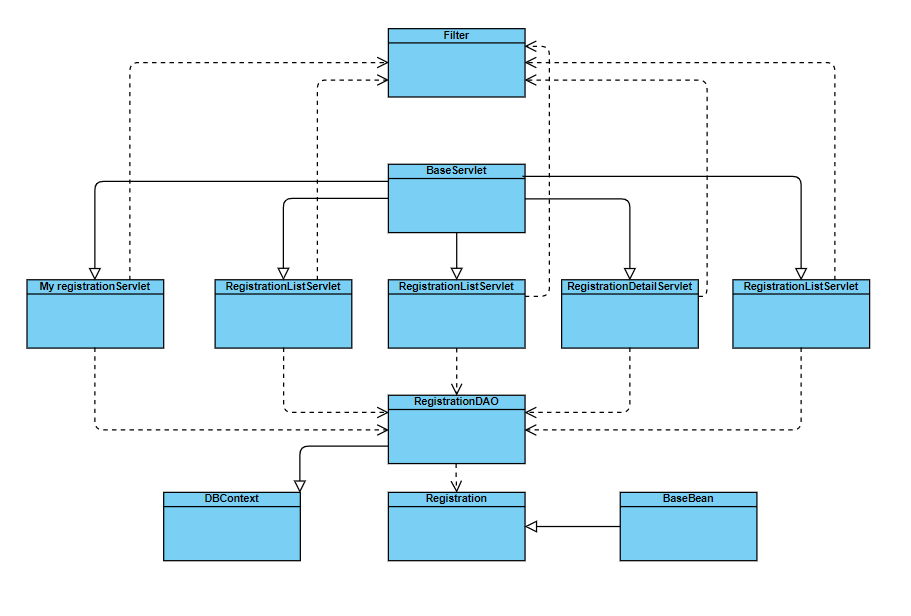
### c. Sequence Diagram(s)



### d. Database Queries

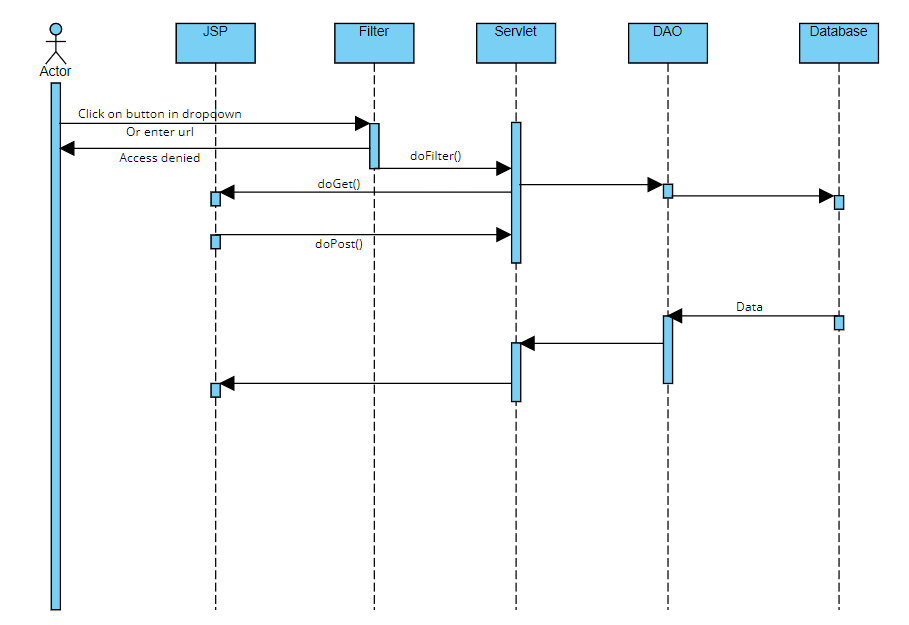
## 7. Sale feature

### a. Class Diagram



### b. Class Specifications

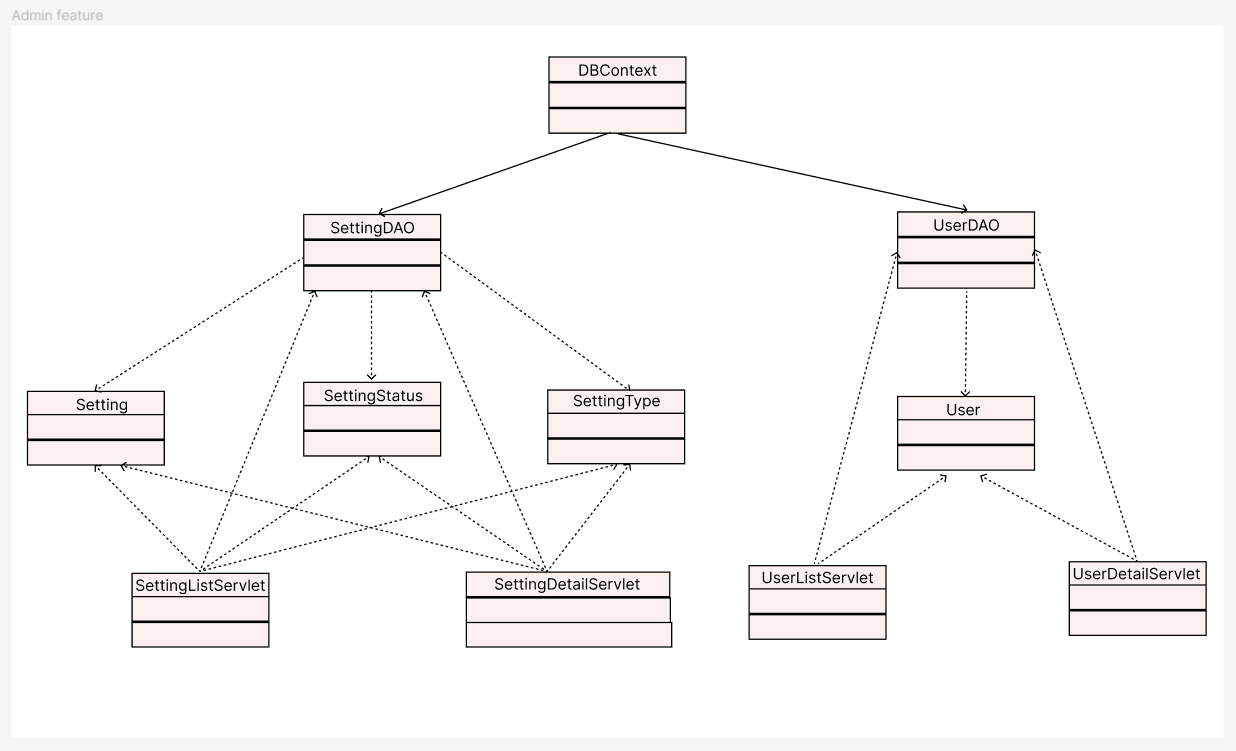
### c. Sequence Diagram(s)



### d. Database Queries

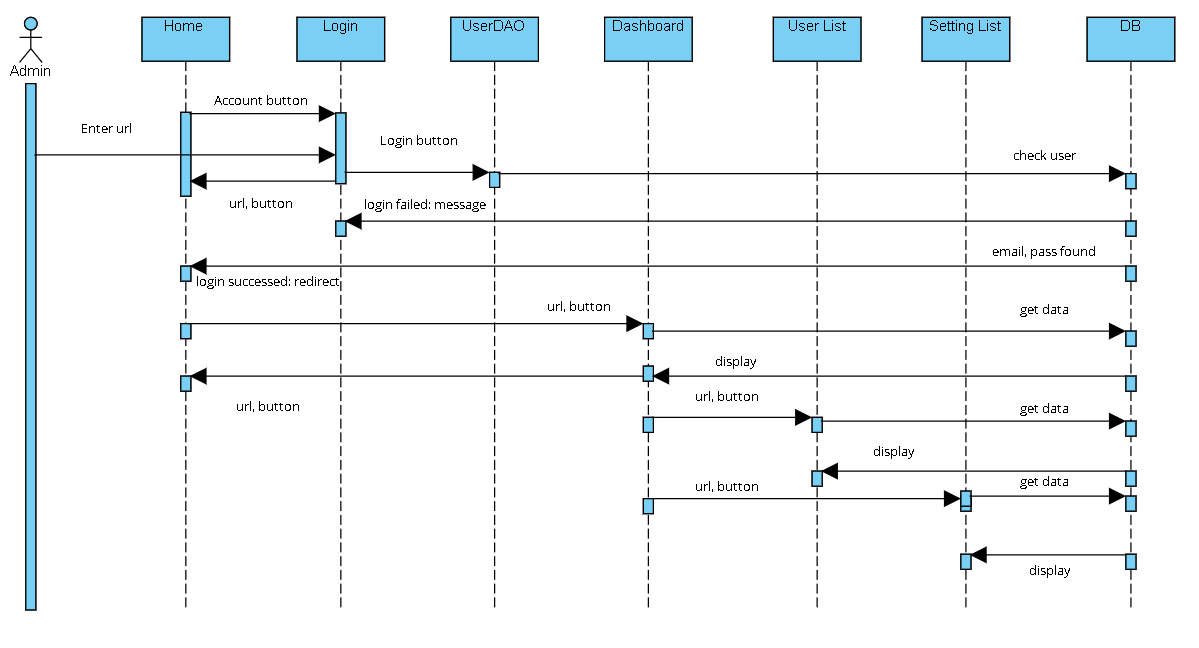
## 8. Sale feature

### a. Class Diagram



### b. Class Specifications

### c. Sequence Diagram(s)



### d. Database Queries