

**<<QUIZ PRACTICE>>**

**Software Design Document**

– Hanoi, 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.2s8eyo1)

[2. <Feature/Function Name2> 5](#_heading=h.17dp8vu)

[III. Database Tables 5](#_heading=h.3rdcrjn)

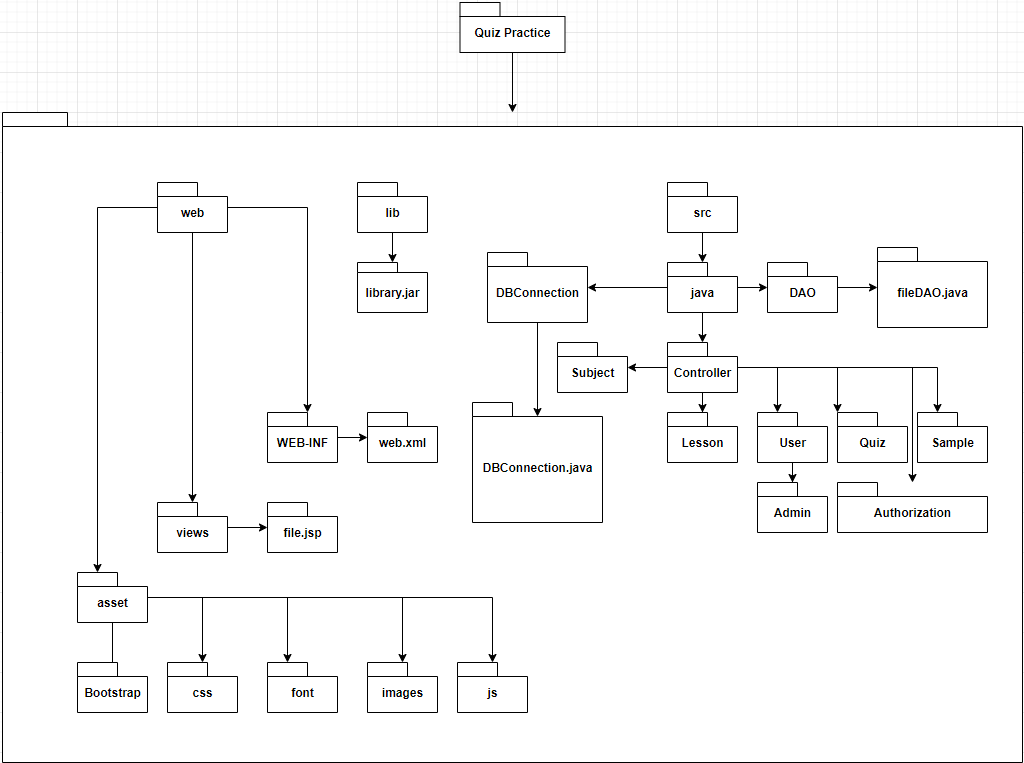
[1. <Table name 1> 5](#_heading=h.26in1rg)

[2. <Table name 2…> 5](#_heading=h.lnxbz9)

# I. Overview

## 1. Code Packages/Namespaces

*[Provide the package diagram for each sub-system. The content of this section including the overall package diagram, the explanation, package and class naming conventions in each package. Please see the sample and description table format below – following Java project naming convention]*



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

| **No** | **Package** | **Description** |
| --- | --- | --- |
| *01* | *web* | *Hold all front-end feature*  *Naming follow HTML/CSS naming style* |
| *02* | *asset* | *Hold library which support for front-end* |
| *03* | *views* | *Hold all jsp file* |
| *04* | *WEB-INF* | *Hold only web.xml file* |
| *05* | *lib* | *Hold all addition library which define as jar type* |
| *06* | *src/java* | *Hold all java file* |
| *07* | *DBConnection* | *Hold only DBConnection.java file* |
| *08* | *DAO* |  |

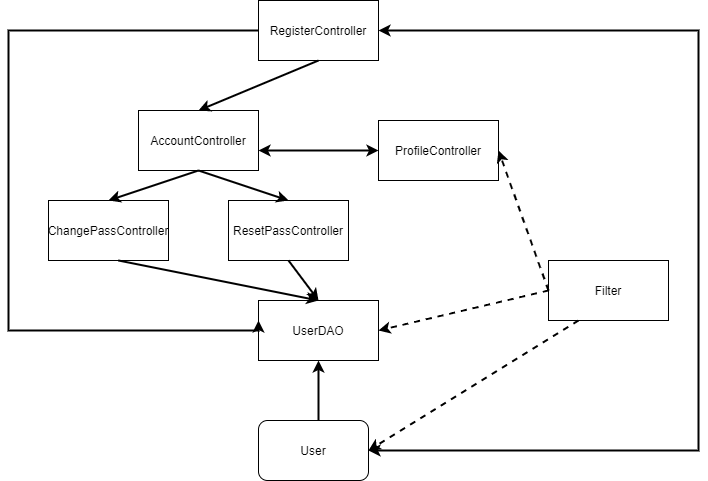
## 2. Coding Conventions

# II. Code Designs

## 1. <Common Feature>

### a. Class Diagram

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



### b. Class Specifications

#### User Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | Get User’s properties | Get data of the properties which belong to the user class |
| 2 | Set User’s properties | Set data of the properties which belong to the user class |

#### UserDAO Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | register(User user) | set data of the user in the Database |
| 2 | updateProfile(User user) | Update profile of the user in the Database |
| 3 | changePass(String pass,int userID ) | Change user password and update in the Database |
| 4 | isExisted(User u) | Check whether the user account or email existed or not |
| 5 | getUser(User u) | Get the user name and password and status |
| 6 | GetUsers() | Get all the user from database |

#### AccountController Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | Get account and compare the input and the data in the data base |
| 2 | Request.setAtribute | Set the message for the alert of the login |
| 3 | Cookies.setcookies | Set the cookies of the username and password |
| 4 | Request.getRequestDispacher | Redirect the user to the screen after login |

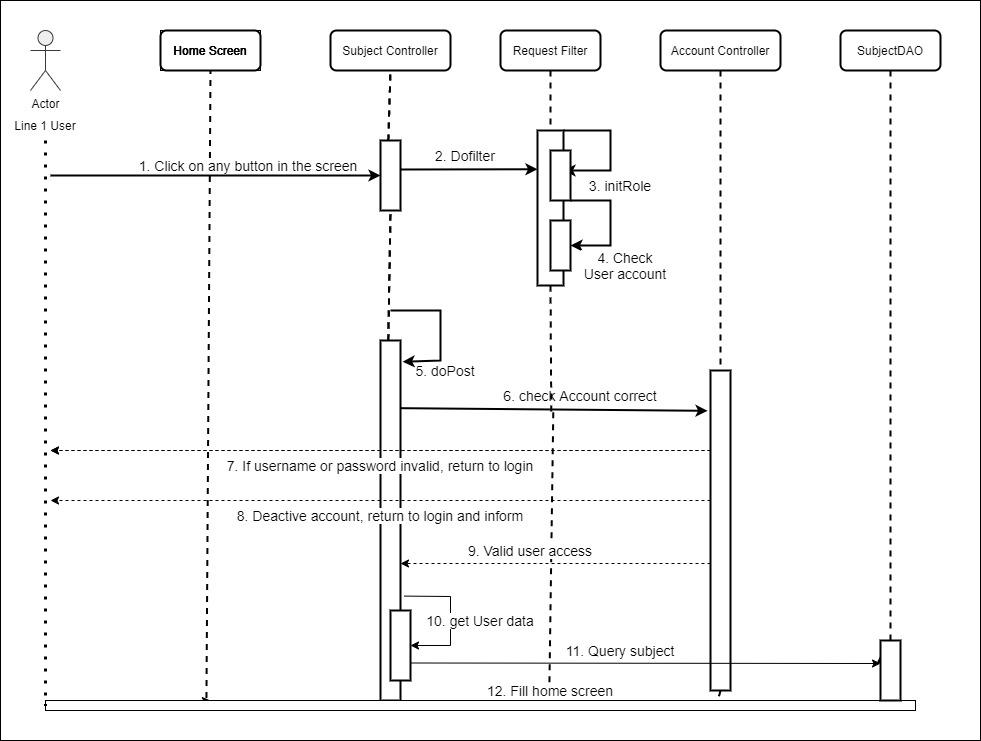
#### EditProfileController Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | Get account and compare the input and the data in the data base |
| 2 | Request.setAtribute | Set the message for the alert of the login |
| 3 | Cookies.setcookies | Set the cookies of the username and password |
| 4 | Request.getRequestDispacher | Redirect the user to the screen after login |
| 5 | updateProfile(newUser) | Update the user to the database |

#### RegisterController Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | Get account and compare the input and the data in the data base |
| 2 | Request.setAtribute | Set the message for the alert of the login |
| 3 | Cookies.setcookies | Set the cookies of the username and password |
| 4 | Request.getRequestDispacher | Redirect the user to the screen after login |
| 5 | isExisted(User u) | Check whether the user account or email existed or not |
| 6 | register(u) | Insert new user profile to the Database |

### c. Sequence Diagram(s)



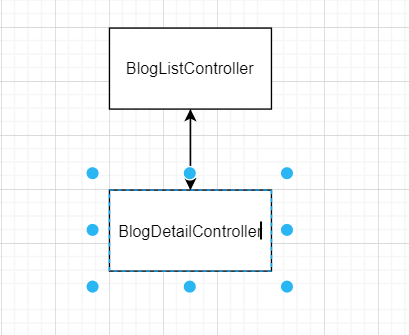
### d. Database queries

| No | Name | Query | Action |
| --- | --- | --- | --- |
| 1 | register | insert into [User]([role], username,[password],[status], first\_name,last\_name,birth,gender,phone,avatar,email,[description],[address], visa,paymentMethod,position,[sign],company,school)  values(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?) | Insert new user profile into the database |
| 2 | updateProfile | update [User] set first\_name=?,  last\_name=?,birth=?,gender=?,phone=?,avatar=?,email=?,  [description]=?,[address]=?,[visa]=?,[paymentMethod]=?,[position]=?,  [sign]=?,company=?,school=? where u\_id = ?" | Update profile of the user into the database |
| 3 | changePass | update [User] set [password]=? where u\_id = ? | Change user password into the database |
| 4 | GetUsers | Select \* from [QuizPractice].[dbo].[User] | Get all user profile from the database |

## 2. <Public feature>

2.2 Blog Function

### a. Class Diagram



### b. Class Specifications

#### Blog Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | Blog constructor | Declare and initialize a blog |
| 1 | Get Blog’s properties | Get data of the properties which belong to the blog class |
| 2 | Set Blog’s properties | Set data of the properties which belong to the blog class |

#### BlogDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | GetBlogs(int mode) | get a list of blog objects based on user authority |
| 2 | GetBlogById(int id) | get a blog object by its id. |
| 3 | GetBlogsByCategory  (String category) | get a list of blog objects based on category |
| 4 | GetBlogCategory() | get a list of blog category |
| 5 | AddBlog(Blog b) | Add a new blog to database |
| 6 | ChangeBlogStatus  (Blog b) | Change status from show to customer to hide from customer and vice versa. |
| 7 | UpdateBlog  (Blog oldBlog,  Blog newBlog) | Update information of a existing blog |

#### BlogListController class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | get a list of blog showing to customer and a list of blog category from database, send data  to blog list screen |
| 2 | doGet | default |
| 3 | doPost | default |

#### BlogDetailController class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | default |
| 2 | doGet | Receive a get HTML request, get the blog id parameter, get that blog objects by received blog id, send data to blog detail screen |
| 3 | doPost | default |

### c. Sequence Diagram(s)

### d. Database queries

### 

| No | Name | Query | Action |
| --- | --- | --- | --- |
| 1 | GetBlogs | SELECT \* FROM [QuizPractice].[dbo].[Blog] order by time desc | get a list of all blog  from database |
| 2 | GetBlogs  ByCategory | SELECT \* FROM [QuizPractice].[dbo].[Blog] where type like " + "'" + category + "'" + "order by time desc | get a list of blog according to category  from database |
| 3 | GetBlogCategory | update [User] set [password]=? where u\_id = ? | get a list of blog category |

## 5. <Marketing feature>

5.2 Blog Function

### a. Class Diagram

#### Blog Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | Blog constructor | Declare and initialize a blog |
| 1 | Get Blog’s properties | Get data of the properties which belong to the blog class |
| 2 | Set Blog’s properties | Set data of the properties which belong to the blog class |

#### BlogDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | GetBlogs(int mode) | get a list of blog objects based on user authority |
| 2 | GetBlogById(int id) | get a blog object by its id. |
| 3 | GetBlogsByCategory  (String category) | get a list of blog objects based on category |
| 4 | GetBlogCategory() | get a list of blog category |
| 5 | AddBlog(Blog b) | Add a new blog to database |
| 6 | ChangeBlogStatus  (Blog b) | Change status from show to customer to hide from customer and vice versa. |
| 7 | UpdateBlog  (Blog oldBlog,  Blog newBlog) | Update information of a existing blog |

#### BlogManageController class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | get a list of all blog and a list of blog category from database, send data to blog manage screen |
| 2 | doGet | default |
| 3 | doPost | default |

#### BlogStatusController class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | Receive a get HTML request, get the blog id parameter,change status of the blog, get a list of all blog and a list of blog category from database, send data to blog manage screen |
| 2 | doGet | default |
| 3 | doPost | default |

#### BlogAddController class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | Receive a post HTML request, get information and initialize a new blog ,add that blog to database, get a list of all blog and a list of blog category from database, send data to blog manage screen |
| 2 | doGet | default |
| 3 | doPost | default |

#### BlogEditController class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | processRequest | Receive a post HTML request, get information and initialize a new blog ,change the information of an existing blog based on blog id, get a list of all blog and a list of blog category from database, send data to blog manage screen |
| 2 | doGet | default |
| 3 | doPost | default |

### c. Sequence Diagram(s)

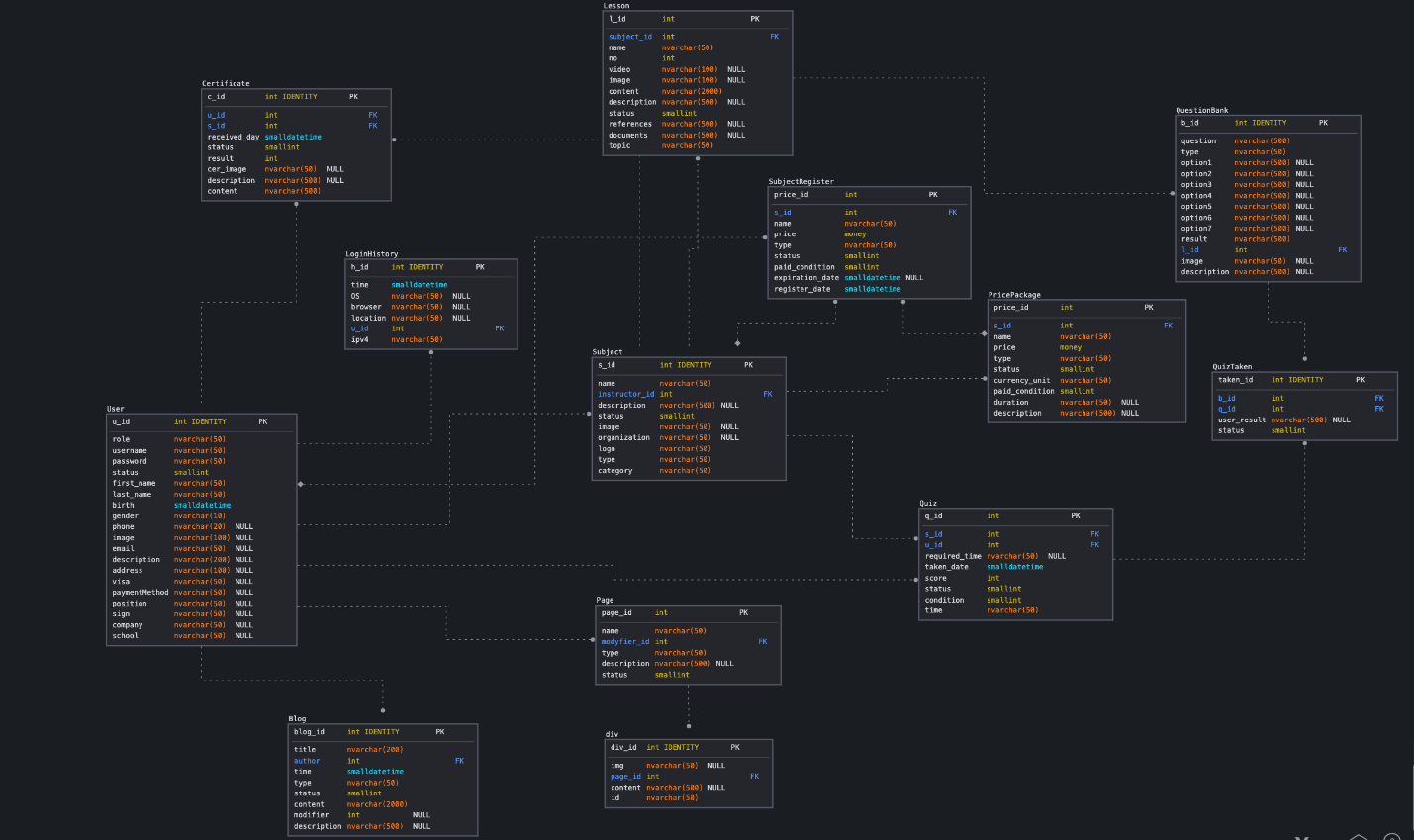
### d. Database queries

### 

| No | Name | Query | Action |
| --- | --- | --- | --- |
| 1 | AddBlog | insert into Blog (title,author,time,type,status,content,description) values(?,?,?,?,?,?,?) | add a new blog to database |
| 2 | Change  BlogStatus | update Blog set status = ? where blog\_id = ? | change status of a blog on database |
| 3 | Update  Blog | update Blog set title = ?,time = ?,type = ?,content = ?,description = ?,modifier = ? where blog\_id = ? | update information of a blog |

# III. Database Design

## 1. Database Schema



## 2. Table Description

| **No** | **Table** | **Description** |
| --- | --- | --- |
| *01* | *<Table name>* | *<Description of the table>*  *- Primary keys: <<list of primary key fields>>*  *- Foreign keys: <<list of foreign key fields>>* |
| *02* |  |  |