

**<<QUIZ PRACTICE>>**

**Software Design Document**

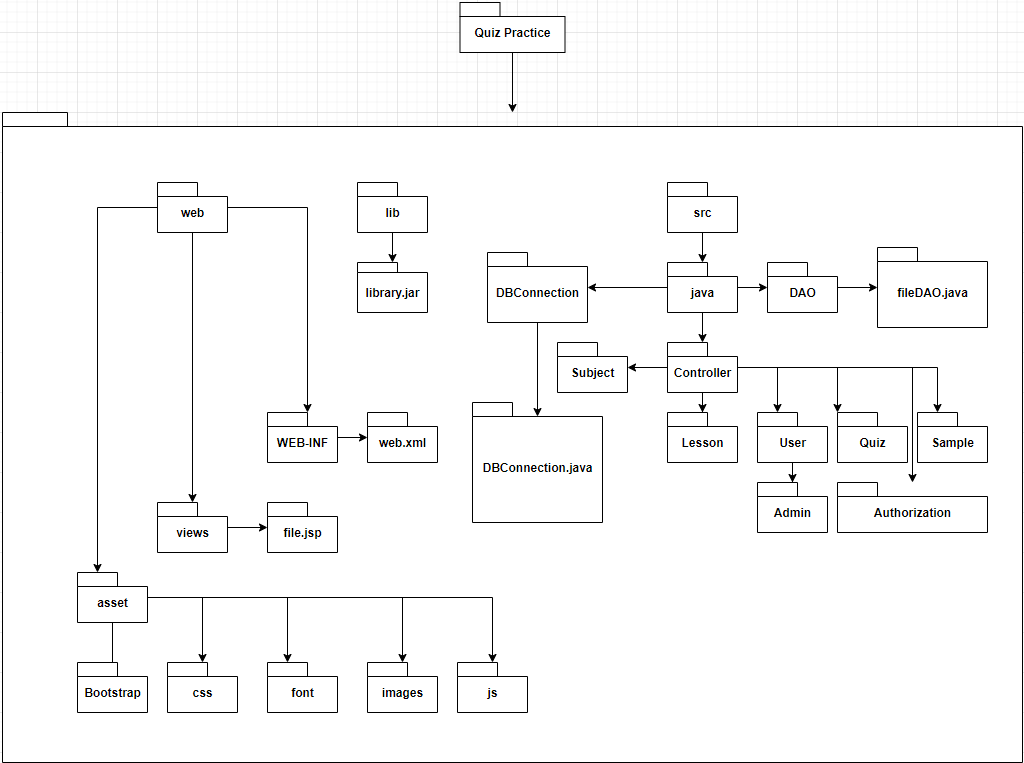
– Hanoi, August 2019 –

**Table of Contents**

# 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* |  |

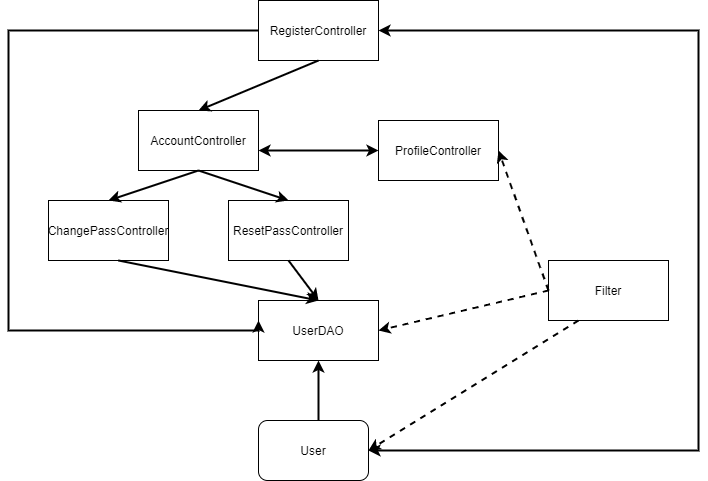
## 2. Coding Conventions

# 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)

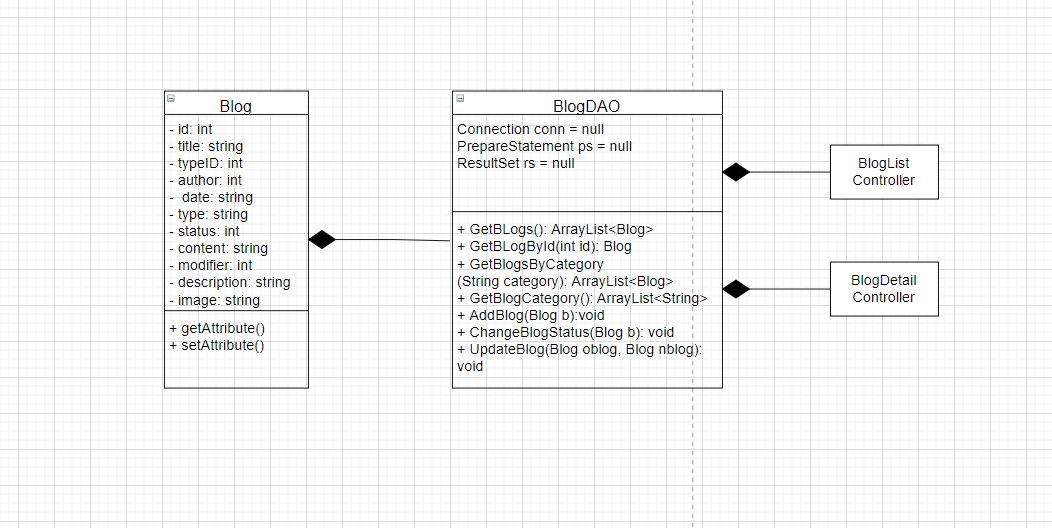


### d. Database queries

|  |  |  |  |
| --- | --- | --- | --- |
| 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>

### a. Class Diagram



Link diagram: https://drive.google.com/file/d/1zqEc9jPA2Tmr7IhZUnc4Gv8CyphLhprc/view?usp=sharing

### b. Class Specifications

#### Subject Class

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

#### Blog Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 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 |

#### Topic Class

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

#### SubjectDAO Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | getSubjects(String condition) | Get data of the subject base on condition in the Database |
| 2 | getSubject(String sjtId) | Get data of one subject |
| 3 | getTopicBySubejectId(int s\_id) | Get data of the topic by subject id in the Database |
| 4 | getCompleteSubjects(String condition) | Get data of the subject (add price, expert name,…) base on condition in the Database |

#### BlogDAO Class

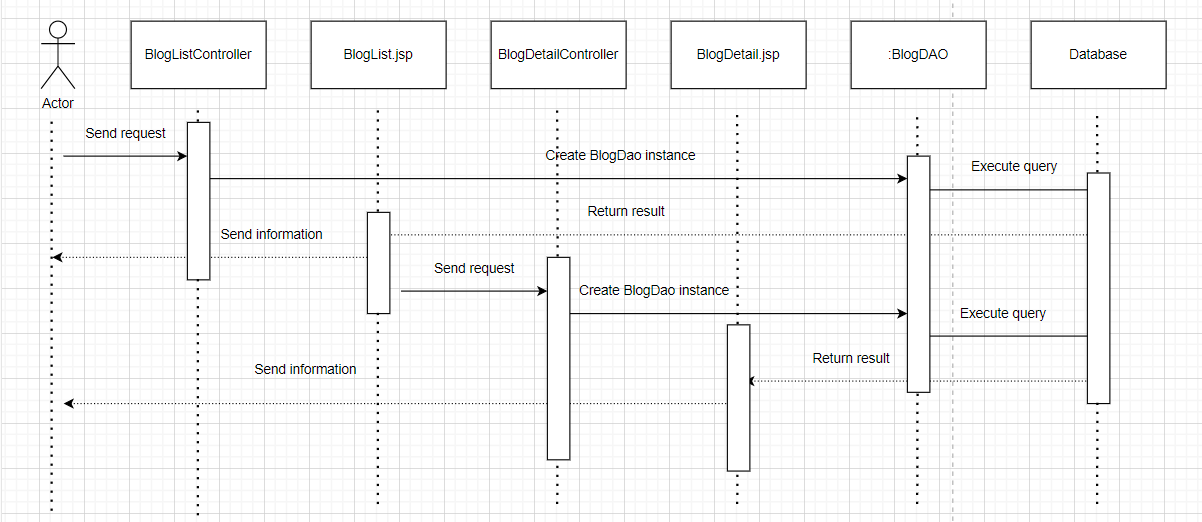
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | getBlogs(int mode) | Get a list of blogs based on mode |
| 2 | GetBlogById(int id) | Get a blog according to its id |
| 3 | GetBlogsByCategory(string category) | Get a list of blogs based on category |
| 4 | GetBlogCategory() | Get a list of blog categories |
| 5 | AddBlog(Blog b) | Insert a new blog into database. |
| 6 | ChangeBlogStatus(Blog b) | Change status of an existing blog |
| 7 | UpdateBlog  (Blog oldblog,Blog newblog) | Change information of an existing blog |

#### Home Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | processRequest | Get blog, subject, expert from database |
| 2 | getRandomEntity(List totalList, List displayList, int n) | Generate random entity to get random expert, blog, subject |
| 3 | getExpert | Get data of random expert from the database |
| 4 | getBlogTags | Get data of random blog from the database |
| 5 | getRecommenedSubjectTags | Get data of recommened subject from the database |
| 6 | getLatestSubjectTags | Get data of new updated subject from the database |

### c. Sequence Diagram(s)

Blog List:



### d. Database queries

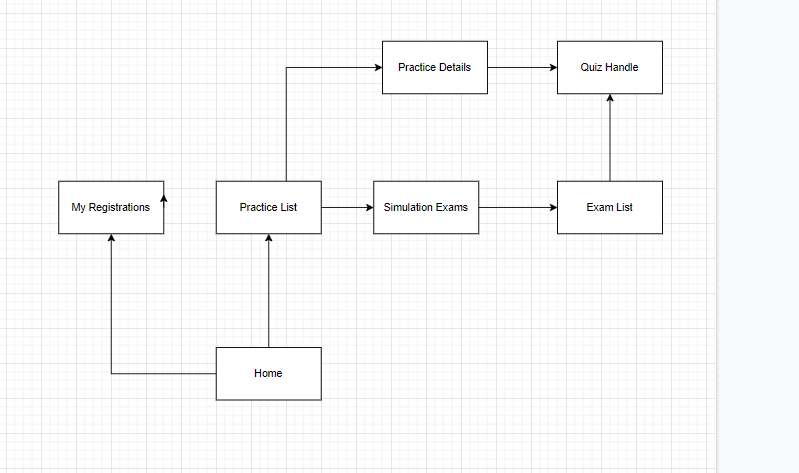
|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Query | Action |
| 1 | getCompleteSubjects | Select [Subject].\*, [PricePackage].price, [PricePackage].currency\_unit, [PricePackage].discount, [User].first\_name, [User].last\_name from [Subject] join [PricePackage] on [Subject].s\_id=[PricePackage].s\_id join [User] on [Subject].instructor\_id=[User].u\_id where [PricePackage].[status]=1 and [Subject].[status]=1 | Get Complete Subject |
| 2 | getSubjects | select \* from [QuizPractice].[dbo].[Subject] | Get All Subject base on condition |
| 3 | getTopicBySubejectId | select \* from [LessonTopic] where s\_id =? | Change user password into the database |

Blog queries

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Query | Action |
| 1 | getBlogs | SELECT \* FROM [QuizPractice].[dbo].[Blog] where status = 1 order by time desc | Get a list of Blogs |
| 2 | getBlogsByCategory | SELECT \* FROM [QuizPractice].[dbo].[Blog] where type like ' category ' order by time desc | Get a list of Blogs having the same category |
| 3 | getBlogCategory | SELECT distinct type from blog | Get the list of blog categories |
| 4 | AddBlog | insert into Blog(title,author,time,type,status,content,description) values(?,?,?,?,?,?,?) | Add a new blog |
| 5 | ChangeBlogStatus | update Blog set status = ? where blog\_id = ? | Change status of an existing blog |
| 6 | UpdateBlog | update Blog set title = ?,time = ?,type = ?,content = ?,description = ?,modifier = ? where blog\_id = ? | Update an existing blog |

## 3. <Customer Feature>

### a. Class Diagram



### b. Class Specifications

#### Quizz Class

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

#### Exam Class

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

#### ExamDAO Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | getExamsByLessonId(int s\_id, String condition) | Get data of the exam base on subject id and condition in the Database |

#### PracticeListController Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | processRequest | Get previous practices of user from database |
| 2 | newPracticeSelection | Get subject and topic base on subject from database |

#### ExamListController Class

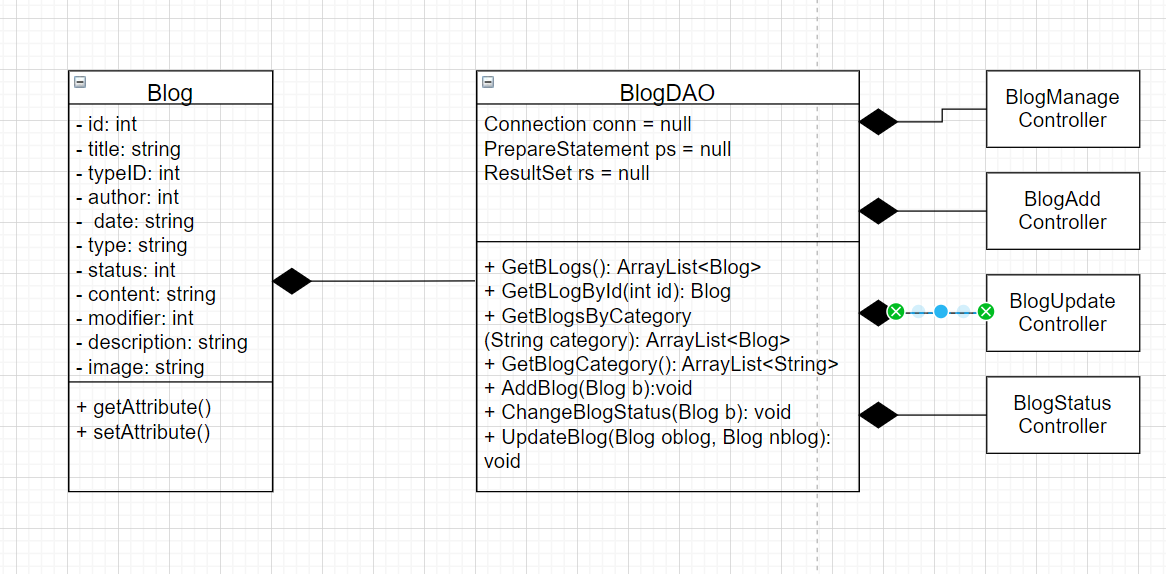
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | processRequest | Get exam list base on customer id from database |

### c. Sequence Diagram(s)

### d. Database queries

## 4. <Marketing Feature>

### a. Class Diagram



### b. Class Specifications

#### Blog Class

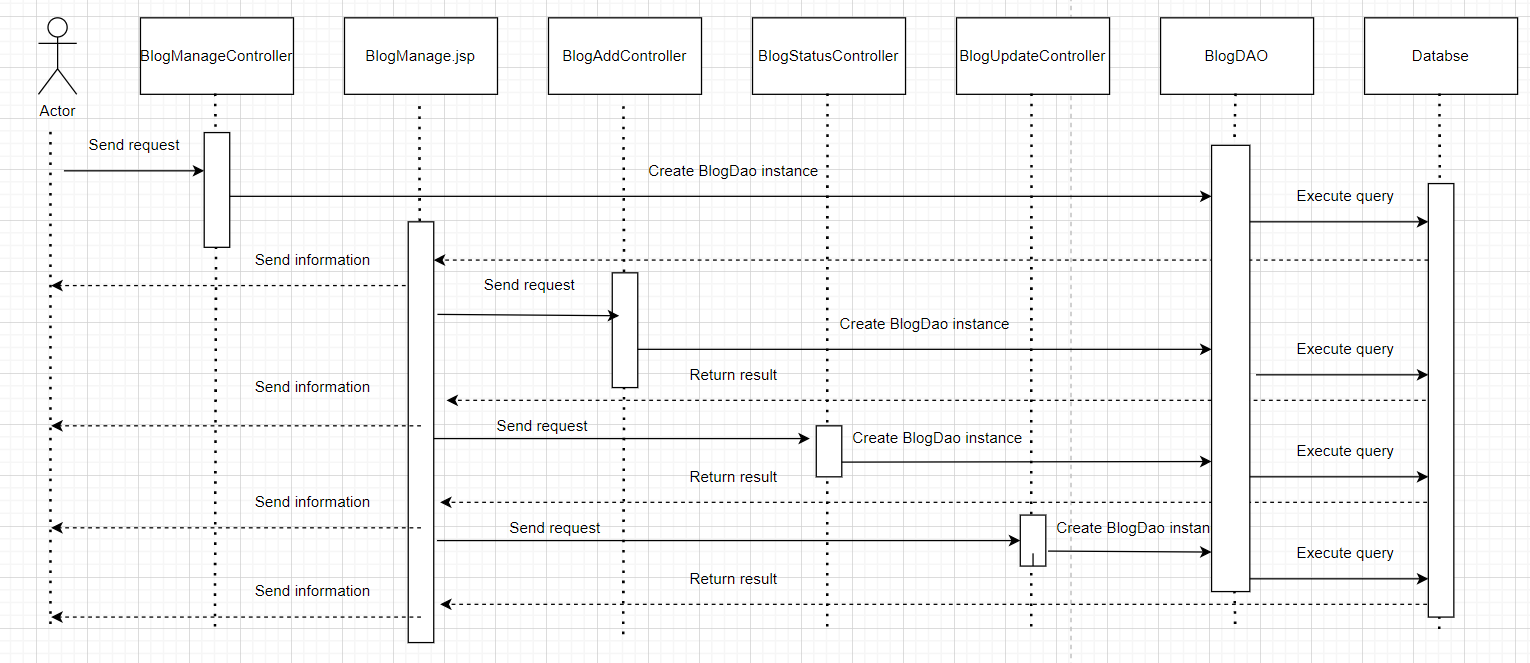
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | Get Blog’s properties | Get data of the properties which belong to the blog class |
| 2 | Set Blog’s properties | Set data of the properties which belong to the blog class |
|  |  |  |

#### BlogDAO Class

#### Home Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | processRequest | Get blog, subject, expert from database |
| 2 | getRandomEntity(List totalList, List displayList, int n) | Generate random entity to get random expert, blog, subject |
| 3 | getExpert | Get data of random expert from the database |
| 4 | getBlogTags | Get data of random blog from the database |
| 5 | getRecommenedSubjectTags | Get data of recommened subject from the database |
| 6 | getLatestSubjectTags | Get data of new updated subject from the database |

### c. Sequence Diagram(s)



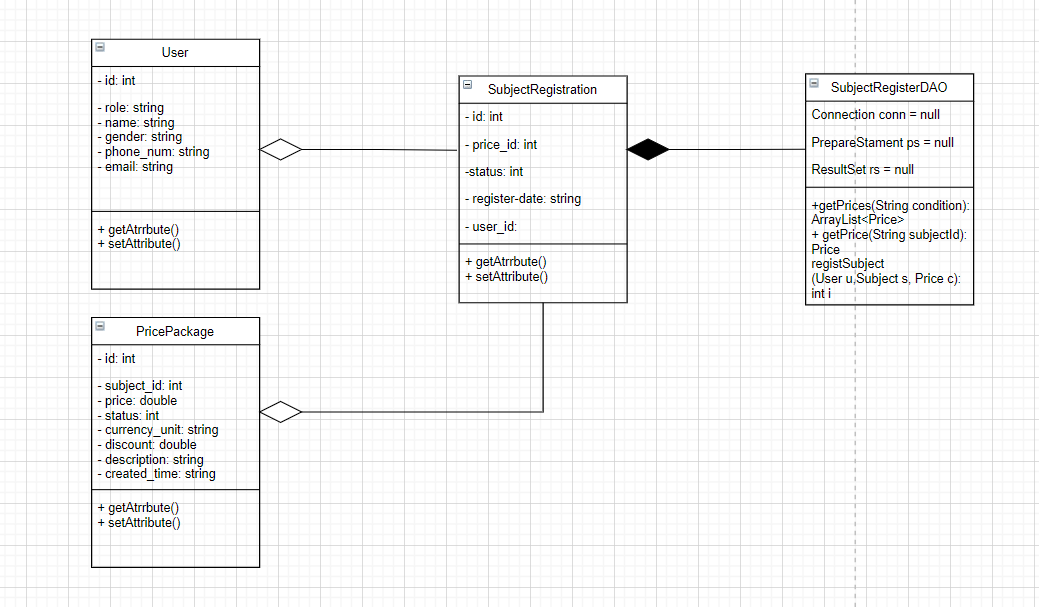
### d. Database queries

Blog queries

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Query | Action |
| 1 | getBlogs | SELECT \* FROM [QuizPractice].[dbo].[Blog] where status = 1 order by time desc | Get a list of Blogs |
| 2 | getBlogsByCategory | SELECT \* FROM [QuizPractice].[dbo].[Blog] where type like ' category ' order by time desc | Get a list of Blogs having the same category |
| 3 | getBlogCategory | SELECT distinct type from blog | Get the list of blog categories |
| 4 | AddBlog | insert into Blog(title,author,time,type,status,content,description) values(?,?,?,?,?,?,?) | Add a new blog |
| 5 | ChangeBlogStatus | update Blog set status = ? where blog\_id = ? | Change status of an existing blog |
| 6 | UpdateBlog | update Blog set title = ?,time = ?,type = ?,content = ?,description = ?,modifier = ? where blog\_id = ? | Update an existing blog |

## 7. <Sale Feature>

#### a. Class Diagram



#### b. Class Specifications

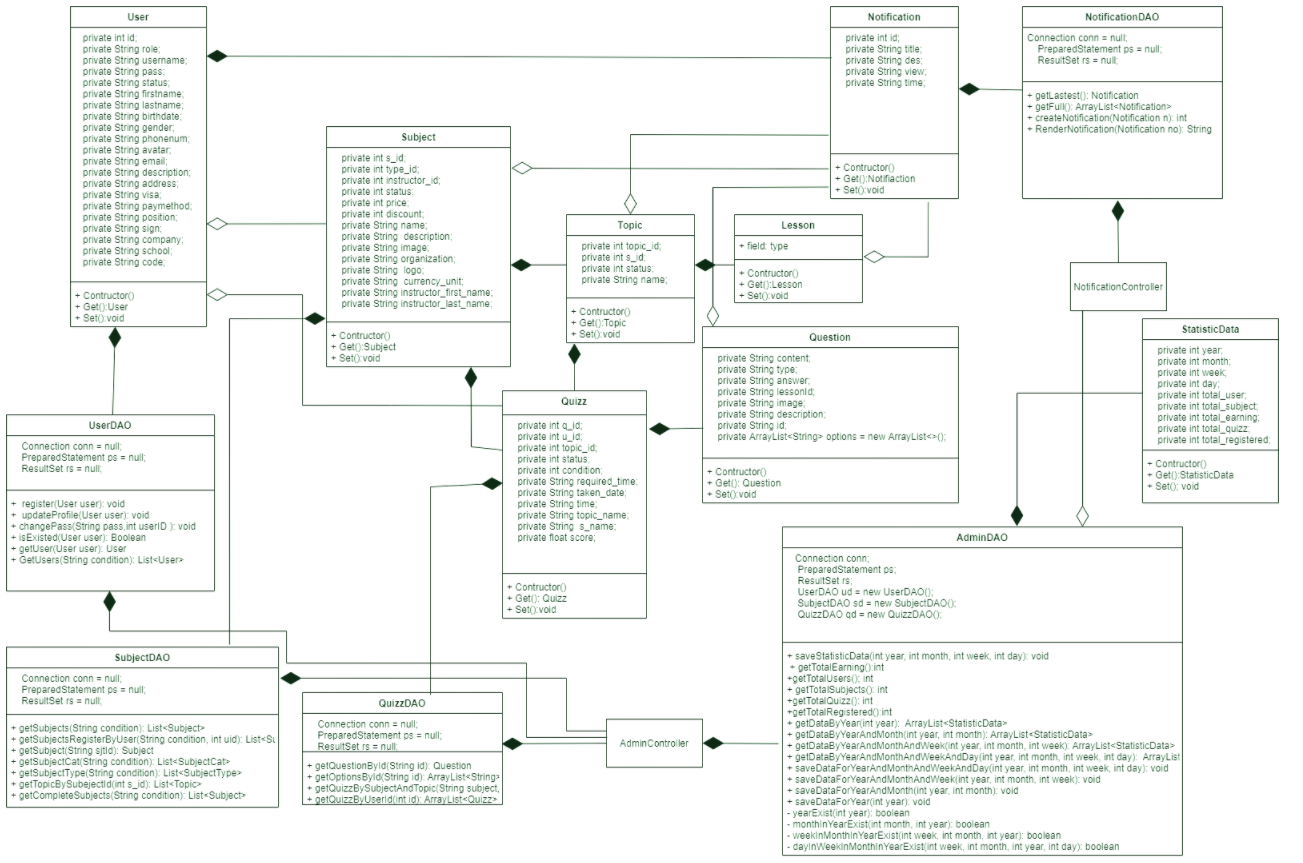
Các class ở feature 5 sẽ cập nhật sau

### c. Sequence Diagram(s)

### d. Database queries

## 8. <Admin Feature>

#### a. Class Diagram



#### 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 |

#### Quizz Class

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

#### Subject Class

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

#### StatisticData Class

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

#### SubjectDAO Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | getSubjects(String condition) | Get data of the subject base on condition in the Database |
| 2 | getSubject(String sjtId) | Get data of one subject |
| 3 | getTopicBySubejectId(int s\_id) | Get data of the topic by subject id in the Database |
| 4 | getCompleteSubjects(String condition) | Get data of the subject (add price, expert name,…) base on condition in the Database |

#### Topic Class

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

#### Notification Class

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

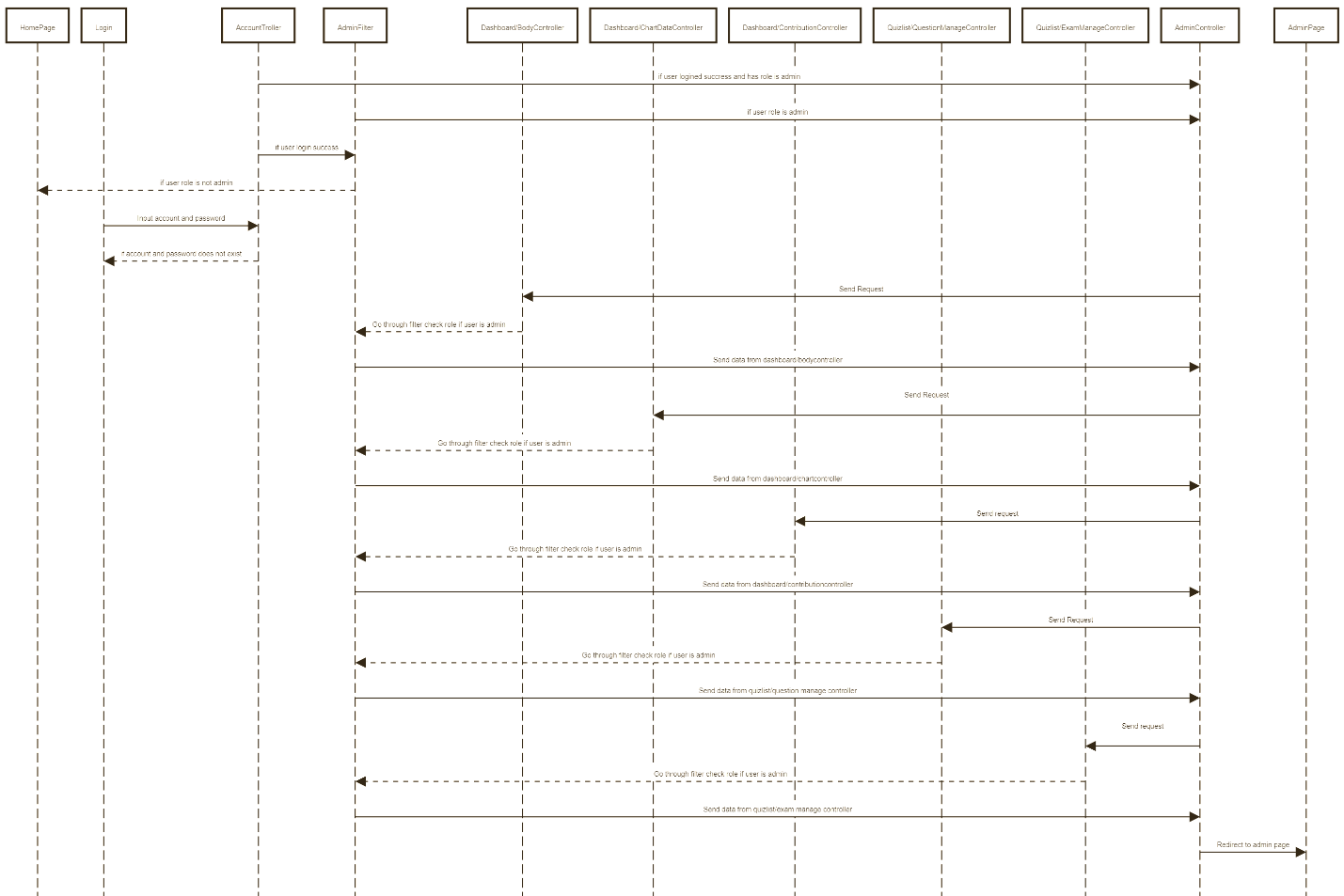
#### SubjectDAO Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | getLastest() | Get the lastest notification from the database |
| 2 | getFull() | Get all notification from the database order by Time desc |
| 3 | createNotification(Notification no) | Create new notification |
| 4 | RenderNotification(Notification no) | Rendet no as html in a string type |

#### AdminDAO Class

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| 1 | saveLoginHistory(LoginHistory input) | Save login history for user |
| 2 | getLoginHistories() | Get all login history information |
| 3 | saveStatisticData(int year, int month, int week, int day) | Save or update data by time |
| 4 | getTotalEarning() | Get newest total earning |
| 5 | getTotalUsers() | Get newest total user |
| 6 | getTotalSubjects() | Get newest total subject |
| 7 | getTotalQuizz() | Get newst total taken quiz |
| 8 | getTotalRegistered() | Get newest total user who have registered subject |
| 9 | getDataByYear(int year) | Get data for a specific year |
| 10 | getDataByYearAndMonth(int year, int month) | Get data for a specific year and month |
| 11 | getDataByYearAndMonthAndWeek(int year, int month, int week) | Get data for a specific year and month and week |
| 12 | getDataByYearAndMonthAndWeekAndDay(int year, int month, int week, int day) | Get data for a specific year and month and week and day |
| 13 | saveDataForYearAndMonthAndWeekAndDay(int year, int month, int week, int day) | Create new data or update date for for a specific year and month and week and day |
| 14 | saveDataForYearAndMonthAndWeek(int year, int month, int week) | Create new data or update date for for a specific year and month and week |
| 15 | saveDataForYearAndMonth(int year, int month) | Create new data or update date for for a specific year and month |
| 16 | saveDataForYear(int year) | Create new data or update date for for a specific year |
| 17 | yearExist(int year) | Check if a specific year has existed in database or not |
| 18 | monthInYearExist(int month, int year) | Check if a specific year and month has existed in database or not |
| 19 | weekInMonthInYearExist(int week, int month, int year) | Check if a specific year and month and week has existed in database or not |
| 20 | dayInWeekInMonthInYearExist(int week, int month, int year, int day) | Check if a specific year and month and week and day has existed in database or not |

c. Sequence Diagram(s)

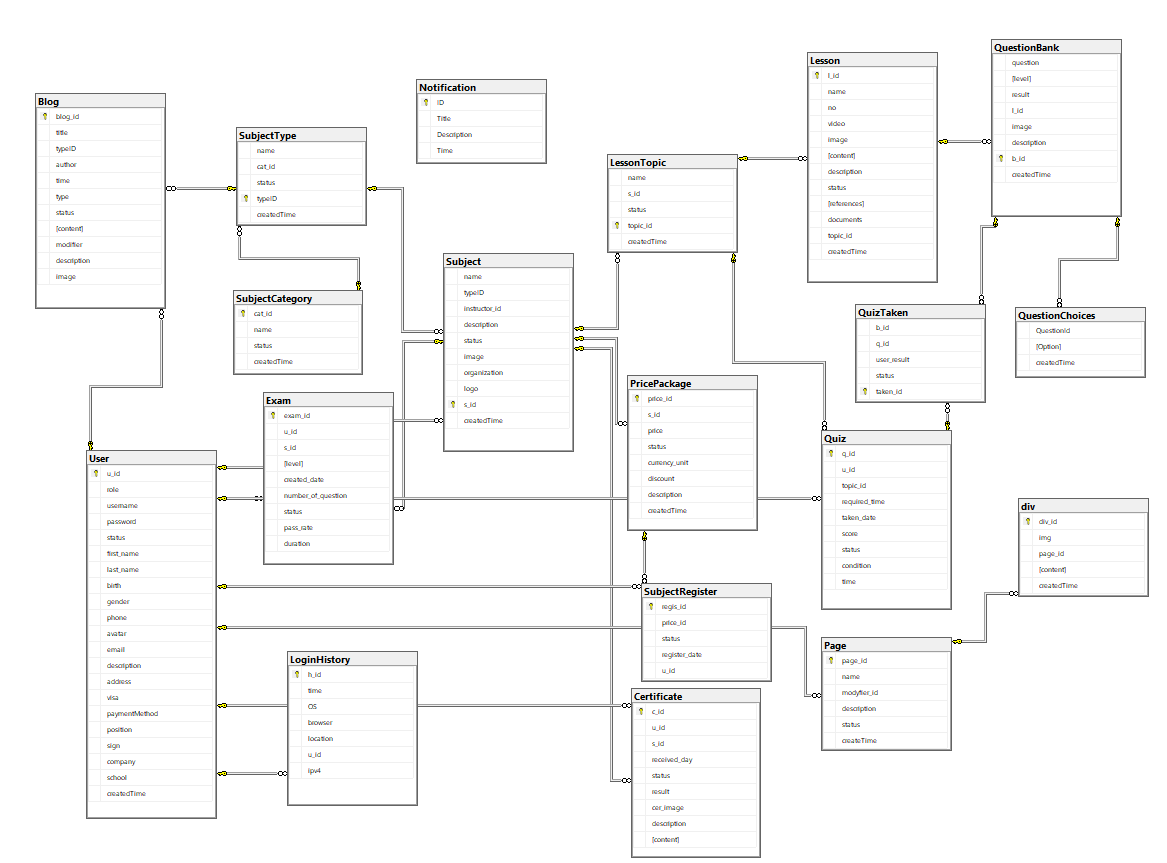


d. Database queries

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Query | Action |
| 0 | 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 |
|  | changePass | update [User] set [password]=? where u\_id = ? | Change user password into the database |
|  | GetUsers | Select \* from [QuizPractice].[dbo].[User] | Get all user profile from the database |
|  | getCompleteSubjects( | Select [Subject].\*, [PricePackage].price, [PricePackage].currency\_unit, [PricePackage].discount, [User].first\_name, [User].last\_name from [Subject] join [PricePackage] on [Subject].s\_id=[PricePackage].s\_id join [User] on [Subject].instructor\_id=[User].u\_id where [PricePackage].[status]=1 and [Subject].[status]=1 |  |
|  | getTopicBySubejectId | select \* from [LessonTopic] where s\_id =? |  |
|  | **getSubjects** | select \* from [QuizPractice].[dbo].[Subject] |  |
|  | getLastest | Select top (1) \* from [QuizPractice].[dbo].[Notification] order by [Time] desc; |  |
|  | getFull | Select \* from [QuizPractice].[dbo].[Notification] order by [Time] desc; |  |
|  | - createNotification | Select into [QuizPractice].[dbo].[Notification] values (?,?,?,getdate()); |  |
|  | getTotalEarning() | SELECT sum(p.price) as 'total' FROM [QuizPractice].[dbo].[SubjectRegister] as sr  join [QuizPractice].[dbo].[PricePackage] as p  on sr.price\_id = p.price\_id |  |
|  | getDataByYear(int year) | SELECT  [year]  ,[Total\_User]  ,[Total\_Subject]  ,[Total\_Earning]  ,[Total\_Quizz]  ,[Total\_Registered]  FROM [QuizPractice].[dbo].[DataByTime] where [Scope] = 'Overview' condition order by [Year] desc |  |
| 1. 11 | getDataByYearAndMonth(int year, int month) | SELECT  [year]  ,[Month]  ,[Total\_User]  ,[Total\_Subject]  ,[Total\_Earning]  ,[Total\_Quizz]  ,[Total\_Registered]  FROM [QuizPractice].[dbo].[DataByTime] where [Scope] = 'Monthly' condition order by [year] desc,[month] desc |  |
|  | getDataByYearAndMonthAndWeek(int year, int month, int week) | SELECT  [year]  ,[Month]  ,[week]  ,[Total\_User]  ,[Total\_Subject]  ,[Total\_Earning]  ,[Total\_Quizz]  ,[Total\_Registered]  FROM [QuizPractice].[dbo].[DataByTime] where [Scope] = 'Weekly' condition order by [year] desc,[month] desc, [Week] desc |  |
|  | getDataByYearAndMonthAndWeekAndDay(int year, int month, int week, int day) | SELECT  [Year]  ,[Month]  ,[Week]  ,[Day]  ,[Total\_User]  ,[Total\_Subject]  ,[Total\_Earning]  ,[Total\_Quizz]  ,[Total\_Registered]  FROM [QuizPractice].[dbo].[DataByTime] where [Scope] = 'Daily' condition order by [year] desc,[month] desc, [Week] desc, [day] desc |  |
|  | saveDataForYearAndMonthAndWeekAndDay(int year, int month, int week, int day) | insert into [QuizPractice].[dbo].[DataByTime](  [Scope],  [Year],  [Month] ,  [Week] ,  [day] ,  [Total\_User],  [Total\_Subject],  [Total\_Earning],  [Total\_Quizz],  [Total\_Registered])  values ('Daily', year ,  month ,  week ,  day ,  getTotalUsers() ,  getTotalSubjects() ,  getTotalEarning() ,  getTotalQuizz() ,  getTotalRegistered() )  update [QuizPractice].[dbo].[DataByTime]  set  [Total\_User] = getTotalUsers() ,  [Total\_Subject]= getTotalSubjects() ,  [Total\_Earning] = getTotalEarning() ,  [Total\_Quizz]= getTotalQuizz() ,  [Total\_Registered] = getTotalRegistered()  where [Scope] = 'Daily' and [Year] = year and [Month] = month and [week] = week and [day] = day |  |
|  | saveDataForYearAndMonthAndWeek(int year, int month, int week) | insert into [QuizPractice].[dbo].[DataByTime](  [Scope],  [Year],  [Month] ,  [Week] ,  [Total\_User],  [Total\_Subject],  [Total\_Earning],  [Total\_Quizz],  [Total\_Registered])  values ('Weekly', year ,  month ,  week ,  getTotalUsers() ,  getTotalSubjects() ,  getTotalEarning() ,  getTotalQuizz() ,  getTotalRegistered() )  update [QuizPractice].[dbo].[DataByTime]  set  [Total\_User] = getTotalUsers() ,  [Total\_Subject]= getTotalSubjects() ,  [Total\_Earning] = getTotalEarning() ,  [Total\_Quizz]= getTotalQuizz() ,  [Total\_Registered] = getTotalRegistered()  where [Scope] = 'Weekly' and [Year] = year and [Month] = month and [week] = week |  |
|  | saveDataForYearAndMonth(int year, int month) | Save data for year: year month: month);  String query = insert into [QuizPractice].[dbo].[DataByTime](  [Scope],  [Year],  [Month] ,  [Total\_User],  [Total\_Subject],  [Total\_Earning],  [Total\_Quizz],  [Total\_Registered])  values (  'Monthly',  year ,  month ,  getTotalUsers() ,  getTotalSubjects() ,  getTotalEarning() ,  getTotalQuizz() ,  getTotalRegistered() )  update [QuizPractice].[dbo].[DataByTime]  set  [Total\_User] = getTotalUsers() ,  [Total\_Subject]= getTotalSubjects() ,  [Total\_Earning] = getTotalEarning() ,  [Total\_Quizz]= getTotalQuizz() ,  [Total\_Registered] = getTotalRegistered()  where [Year] = year and [Month] = month and [Scope] = 'Monthly' |  |
|  | saveDataForYear(int year) | insert into [QuizPractice].[dbo].[DataByTime](  [Scope],  [Year] ,[Total\_User]  ,[Total\_Subject]  ,[Total\_Earning]  ,[Total\_Quizz]  ,[Total\_Registered]) values ('Overview', year ,  getTotalUsers() ,  getTotalSubjects() ,  getTotalEarning() ,  getTotalQuizz() ,  getTotalRegistered() )  update [QuizPractice].[dbo].[DataByTime]  set  [Total\_User] = getTotalUsers() ,  [Total\_Subject]= getTotalSubjects() ,  [Total\_Earning] = getTotalEarning() ,  [Total\_Quizz]= getTotalQuizz() ,  [Total\_Registered] = getTotalRegistered()  where [Year] = year and [Scope] = 'Overview' |  |

# III. Database Design

## 1. Database Schema



## 2. Table Description

### 2.1 User

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | U\_id | Int |  |  | X | PK |  |
| 2 | Role | Nvarchar | 50 |  | X |  |  |
| 3 | Username | Nvarchar | 50 |  | X |  |  |
| 4 | Password | Nvarchar | 50 |  | X |  |  |
| 5 | Status | Smallint |  |  | X |  |  |
| 6 | First\_name | Nvarchar | 50 |  | X |  |  |
| 7 | Last\_name | Nvarchar | 50 |  | X |  |  |
| 8 | Birth | Smalldatetime |  |  | X |  |  |
| 9 | Gender | Nvarchar | 10 |  | X |  |  |
| 10 | Phone | Nvarchar | 20 |  |  |  |  |
| 11 | Avatar | Nvarchar | 300 |  |  |  |  |
| 12 | Email | Nvachar | 50 |  |  |  |  |
| 13 | Description | Nvarchar | 500 |  |  |  |  |
| 14 | Address | Nvarchar | 100 |  |  |  |  |
| 15 | Visa | Nvarchar | 50 |  |  |  |  |
| 16 | paymentMethod | Nvarchar | 50 |  |  |  |  |
| 17 | Position | Nvarchar | 50 |  |  |  |  |
| 18 | Sign | Nvarchar | 50 |  |  |  |  |
| 19 | Company | Nvarchar | 50 |  |  |  |  |
| 20 | School | Nvarchar | 50 |  |  |  |  |
| 21 | createdTime | Smalldatetime |  |  | X |  |  |

### 2.2 Subject

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | S\_id | int |  |  | X | PK |  |
| 2 | typeID | Int |  |  | X | FK | Reference SubjectType |
| 3 | Instructor\_id | Int |  |  | X | FK | Reference User |
| 4 | Name | Nvarchar | 50 |  | X |  |  |
| 5 | Description | Nvarchar | 50 |  |  |  |  |
| 6 | Image | Nvarchar | 300 |  |  |  |  |
| 8 | Logo | Nvarchar | 300 |  | X |  |  |
| 9 | Status | Smallint |  |  | X |  |  |
| 10 | createdTime | Smalldatetime |  |  | x |  |  |

### 2.3 SubjectType

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | typeID | int |  |  | X | PK |  |
| 2 | Cat\_id | int |  |  | X | FK | Reference SubjectCategory |
| 3 | Name | Nvarchar | 50 |  | X |  |  |
| 4 | Status | Smallint |  |  | X |  |  |
| 5 | CreatedTime | Smalldatetime |  |  | X |  |  |

### 2.4 SubjectCategory

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | Cat\_id | int |  |  | X | PK |  |
| 2 | Name | Nvarchar | 50 |  | X |  |  |
| 3 | Status | Smallint |  |  | X |  |  |
| 4 | createdTime | smalldatetime |  |  | X |  |  |

### 2.5 Lesson

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | l\_id | int |  |  | X | PK |  |
| 2 | name | Nvarchar | 500 |  | X |  |  |
| 3 | No | Int |  |  | X |  |  |
| 4 | Video | Nvarchar | 300 |  |  |  |  |
| 5 | Image | Nvarchar | 300 |  |  |  |  |
| 6 | Content | Text |  |  | X |  |  |
| 7 | Description | Text |  |  |  |  |  |
| 8 | Status | Smallint |  |  | X |  |  |
| 9 | References | Nvarchar | 500 |  |  |  |  |
| 10 | Documents | Nvarchar | 500 |  |  |  |  |
| 11 | Topic\_id | Int |  |  |  | FK | Reference LessonTopic |
| 12 | createdTime | Smalldatetime |  |  | x |  |  |

### 2.6 LesonTopic

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | Topic\_id | int |  |  | X | PK |  |
| 2 | S\_id | Int |  |  | X | FK | Reference Subject |
| 3 | Status | Smallint |  |  | X |  |  |
| 4 | Name | Nvarchar | 500 |  | X |  |  |
| 5 | createdTime | Smalldatetime |  |  | X |  |  |

### 2.7 PricePackage

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | Price\_id | int |  |  | X | PK |  |
| 2 | S\_id | Int |  |  | X | FK | Reference Subject |
| 3 | Price | Int |  |  | X |  |  |
| 4 | Status | Smallint |  |  | X |  |  |
| 5 | Currency\_unit | Nvarchar | 50 |  | X |  |  |
| 6 | Discount | Int |  |  |  |  |  |
| 7 | Description | Nvarchar | 500 |  |  |  |  |
| 8 | createdTime | Smalldatetiem |  |  |  | x |  |

### 2.8 QuestionBank

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | B\_id | int |  |  | X | PK |  |
| 2 | L\_id | Int |  |  | X | FK | Reference Lesson |
| 3 | Question | Nvarchar | 500 |  | X |  |  |
| 4 | Level | Nvarchar | 10 |  | X |  |  |
| 5 | Result | Nvarchar | 1000 |  | X |  |  |
| 6 | Image | Nvarchar | 300 |  |  |  |  |
| 7 | Description | Nvarchar | 500 |  |  |  |  |
| 8 | createdTime | Smalldatetime |  |  | X |  |  |

### 2.9 QuestionChoices

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | QuestionId | int |  |  | X | FK | Reference QuestionBank |
| 2 | Option | Nvarchar | 500 |  | X |  |  |
| 3 | createdTime | Smalldatetime |  |  | X |  |  |

### 2.9 Blog

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | blog\_id | int |  |  | X | PK |  |
| 2 | title | Nvachar | 50 |  | X |  |  |
| 3 | author | int |  |  |  | FK |  |
| 4 | Time | int |  |  | X |  |  |
| 5 | Type | Smalldatetime |  |  | X |  |  |
| 6 | Status | Nvarchar | 300 |  | X |  |  |
| 7 | Content | Small int |  |  | X |  |  |
| 8 | modifier | Int |  |  |  |  |  |
| 9 | description | Text |  |  |  |  |  |
| 10 | image | nvachar |  |  |  |  |  |

### 3.0 Certificate

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | c\_id | int |  |  | X | PK |  |
| 2 | u\_id | int |  |  | X | FK |  |
| 3 | s\_id | int |  |  | X | FK |  |
| 4 | Received\_day | Smalldatetime |  |  | X |  |  |
| 5 | status | Small int |  |  | X |  |  |
| 6 | result | float |  |  | X |  |  |
| 7 | Cer\_image | Nvarchar | 300 |  |  |  |  |
| 8 | description | Nvarchar | 500 |  |  |  |  |
| 9 | content | Nvarchar | 500 |  | X |  |  |

### 3.1 div

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | Div\_id | nvarchar | 50 |  | X | PK |  |
| 2 | img | nvarchar | 300 |  |  |  |  |
| 3 | page\_id | int |  |  | X | FK |  |
| 4 | content | nvarchar | 500 |  |  |  |  |
| 5 | CreateTime | Smalldatetime |  |  | X |  |  |

### 3.2 Exam

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | Exam\_id | int |  |  | X | PK |  |
| 2 | u\_id | int |  |  | X | FK |  |
| 3 | s\_id | int |  |  | X | FK |  |
| 4 | level | Nvarchar | 50 |  |  |  |  |
| 5 | Create\_date | Smalldatetime |  |  | X |  |  |
| 6 | Number\_of\_question | int |  |  | X |  |  |
| 7 | status | Small int |  |  | X |  |  |
| 8 | Pass rate | Small int |  |  | X |  |  |
| 9 | Duration | int |  |  | X |  |  |

### 3.3 Login History

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | h\_id | int |  |  | X | PK |  |
| 2 | time | Smalldatetime |  |  | X |  |  |
| 3 | OS | Nvarchar |  |  |  |  |  |
| 5 | browser | Nvarchar |  |  |  |  |  |
| 6 | location | Nvarchar |  |  |  |  |  |
| 7 | U\_id | int |  |  | X | FK |  |
| 8 | Ipv4 | Small int |  |  | X |  |  |

### 3.4 Notification

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | ID | int |  |  | X | PK |  |
| 2 | Title | nchar | 30 |  | X |  |  |
| 3 | Description | nchar | 225 |  | X |  |  |
| 5 | time | smalldatetime |  |  | X |  |  |

### 3.5 Page

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | page\_id | int |  |  | X | PK |  |
| 2 | name | Nvarchar | 50 |  | X |  |  |
| 3 | Modifier\_id | int |  |  | X | FK |  |
| 5 | description | Nvarchar | 500 |  |  |  |  |
| 6 | status | Small int |  |  | X |  |  |
| 7 | Create\_time | Small date time |  |  |  |  |  |

### 3.6 Price Package

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | price\_id | int |  |  | X | PK |  |
| 2 | s\_id | int |  |  | X | FK |  |
| 3 | price | int |  |  | X |  |  |
| 5 | status | Small int |  |  | X |  |  |
| 6 | Currency unit | Nvarchar | 50 |  | X |  |  |
| 7 | Discount | Int |  |  |  |  |  |
| 8 | Description | Nvarchar | 500 |  |  |  |  |
| 9 | createTime | smalldatetime |  |  |  |  |  |

### 3.7 QuizTaken

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | b\_id | int |  |  | X |  |  |
| 2 | q\_id | int |  |  | X |  |  |
| 3 | user\_result | nvarchar | 500 |  |  |  |  |
| 5 | status | Small int |  |  | X |  |  |
| 6 | Taken\_id | int |  |  | X | PK |  |
| 7 | createtime | Small date time |  |  |  |  |  |

### 3.8 Subject Register

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | regis\_id | int |  |  | X | FK |  |
| 2 | price\_id | int |  |  | X | FK |  |
| 3 | status | Small int |  |  | X |  |  |
| 5 | Register date | Smalldate time |  |  | X |  |  |
| 6 | u\_id | int |  |  | X | PK |  |