

**ONLINE LEARNING WEBSITE**

**Software Design Specification**

– Hanoi, August 2022 –

Record of changeS

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

\*A - Added M - Modified D - Deleted

**Table of Contents**

[I. Overview 4](#_Toc96516286)

[1. Code Packages 4](#_Toc96516287)

[2. Database Design 4](#_Toc96516288)

[a. Database Schema 4](#_Toc96516289)

[b. Table Description 4](#_Toc96516290)

[II. Code Designs 5](#_Toc96516291)

[1. <Feature/Function Name1> 5](#_Toc96516292)

[a. Class Diagram 5](#_Toc96516293)

[b. Class Specifications 5](#_Toc96516294)

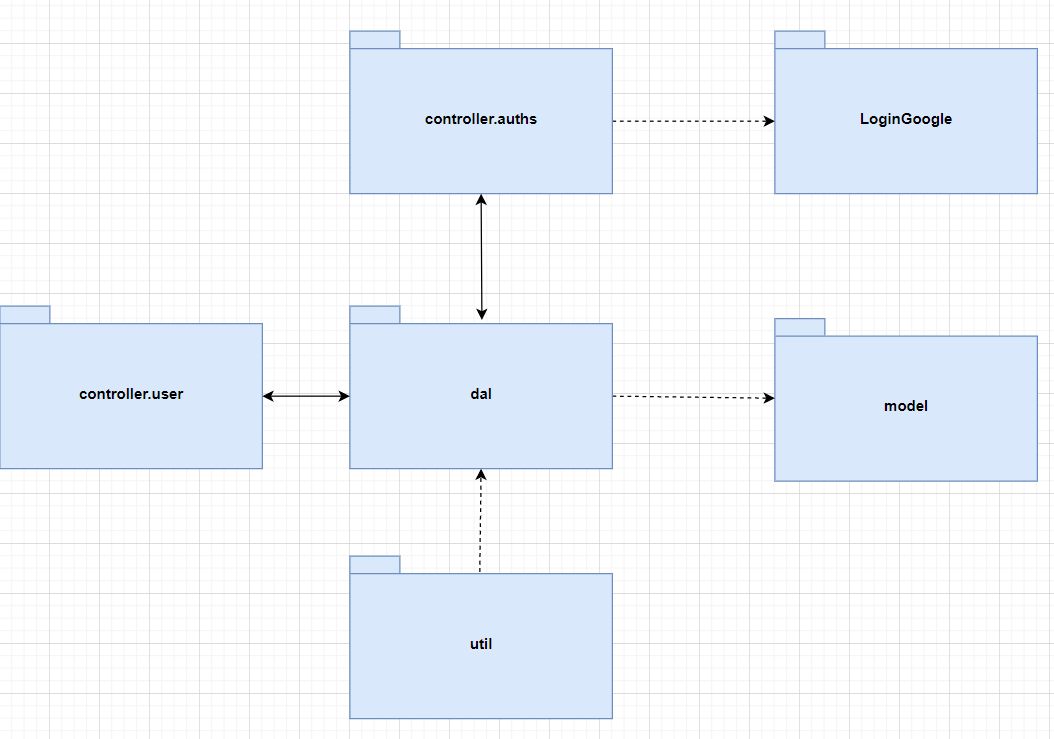
[c. Sequence Diagram(s) 5](#_Toc96516295)

[d. Database queries 6](#_Toc96516296)

[2. <Feature/Function Name2> 6](#_Toc96516297)

# I. Overview

## 1. Code Packages



***Package descriptions***

|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| *01* | *Dal* | *This component is used to get data directly FROM the database* |
| *02* | *Util* | *<Description of the package>* |
| *03* | *Model* | *This component includes interfaces used to receive data retrieved FROM the database and classes used to receive data* |
| *04* | *Controller.user* | *Acts as a go-between for the service layer and the view, passing on different input requests.* |
| *05* | *Controller.auths* | *This component for the service authorities, such as validate user name password, sign in, log out.* |
| *06* | *LoginGoogle* | *This component use to login page with Gmail.* |

## 2. Database Design

### a. Database Schema

Diagram

Description automatically generated

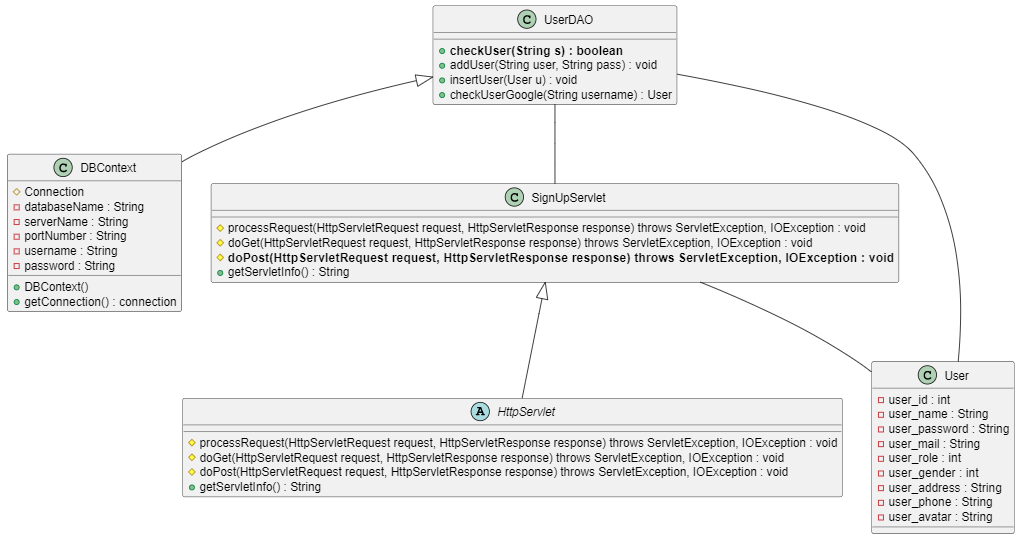
### b. Table Description

|  |  |  |
| --- | --- | --- |
| **No** | **Table** | **Description** |
| *1* | *User* | *Information of the User in Online learning system* |
| *2* | *Discussion* | *The table store rate and discussion of user in Online learning system* |
| *3* | *Record* | *Storage practice date, mark of student when they finished a multiple choice practice of course.* |
| *4* | *Enroll* | *Store user's enrollment information for courses* |
| *5* | *Course* | *Information of the course in Online learning system* |
| *6* | *CommentCourse* | *The table store rate, question, and discussion of user in Online learning system* |
| *7* | *Quizz* | *The practice library of Online learning system* |
| *8* | *Question* | *The question library of Online learning system* |
| *9* | *Percentage* | *Store percentage of student when they finish a lesson* |
| *10* | *CommentLesson* | *The table store rate, question, and discussion of user in Online learning system* |
| *11* | *Lesson* | *Information of the lesson in Online learning system* |
| *12* | *Chapter* | *Information of the chapter in Online learning system* |
| *13* | *Image* | *The image library of Online learning system* |

# II. Code Designs

## 1. Register

### a. Class Diagram



### b. Class Specifications

*SignUpServlet*

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doPost(HttpServletRequest request, HttpServletResponse response)* | *Get email, password and fullname from resgister form, check it and add to database* |
|  |  |  |

#### DBContext

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***User***

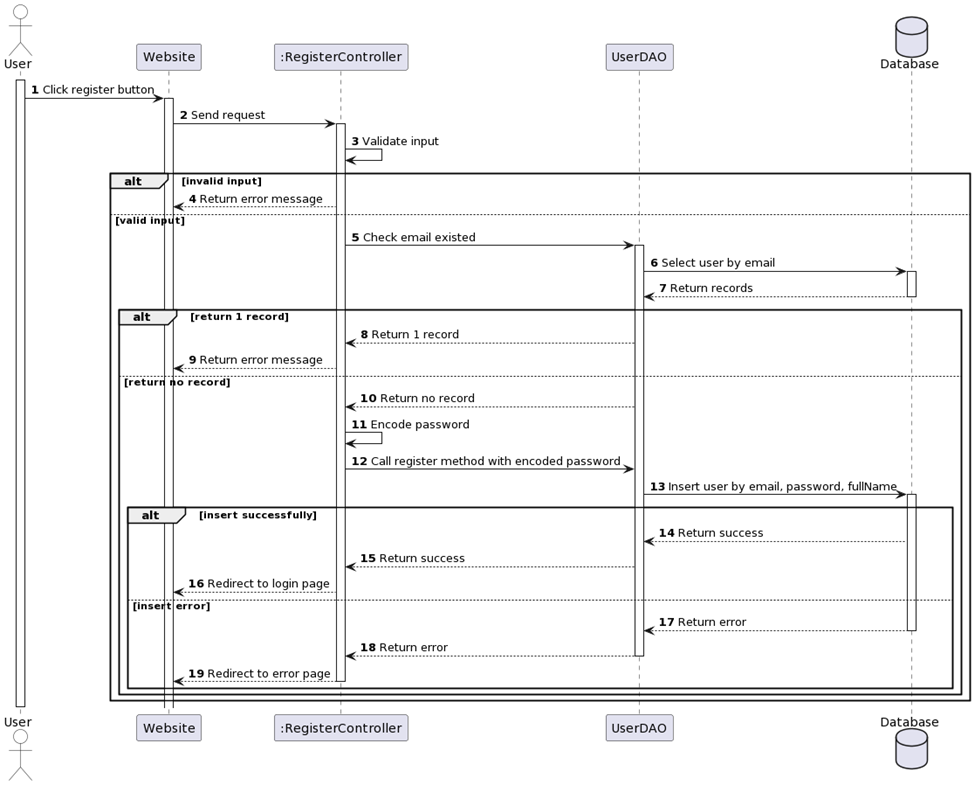
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***UserDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *ChechUser(String s)* | *Check if user name is exist or not* |
| *02* | *CheckUserGoogle*  *(String username )* | *Check google email is extisted or not in the system* |

### c. Sequence Diagram(s)

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



### d. Database Queries

INSERT INTO [dbo].[User]\n"

+ " ([user\_name]\n"

+ " ,[user\_mail]\n"

+ " ,[user\_password]\n"

+ " ,[user\_role]\n"

+ " ,[user\_gender]\n"

+ " ,[user\_address]\n"

+ " ,[user\_phone]\n"

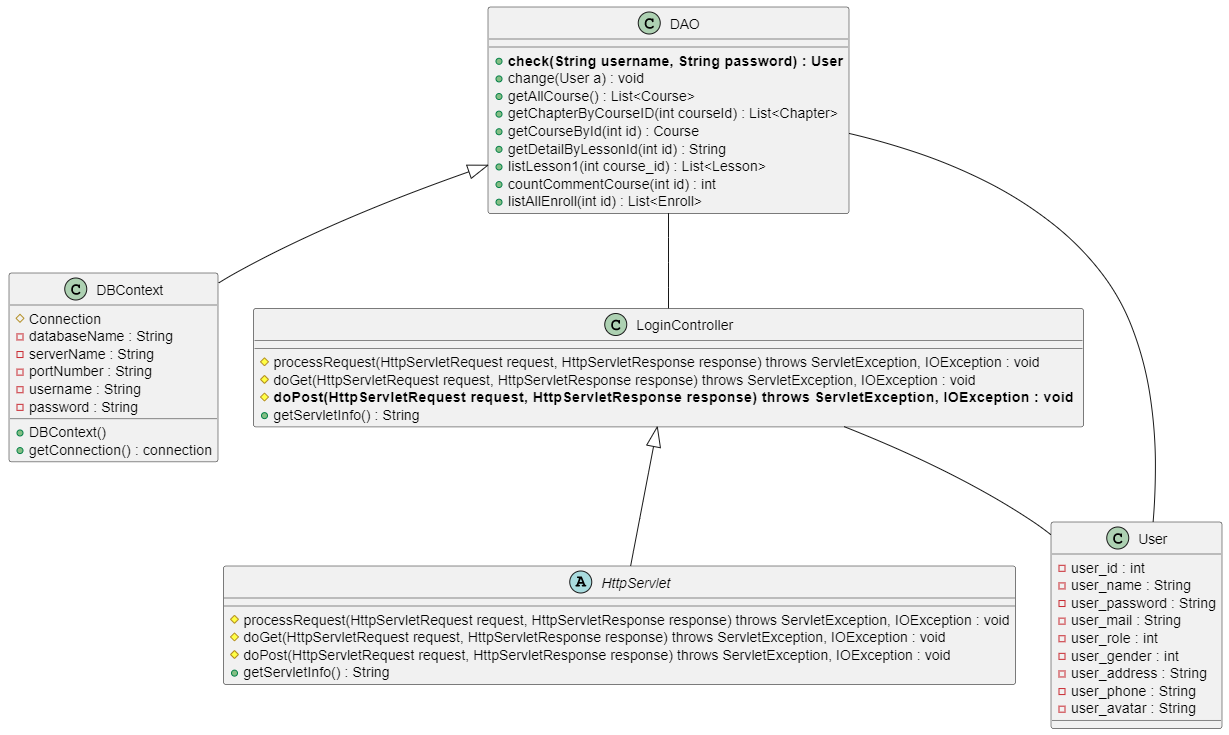
+ " ,[user\_avatar])\n"

+ " VALUES\n"

+ " (?,?,?,?,?,?,?,?)

## 2. Login

### a. Class Diagram



### b. Class Specifications

#### LoginController

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doPost(HttpServletRequest request, HttpServletResponse response)* | *Get email, password and fullname from resgister form, check it and add to database* |

#### DBContext

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

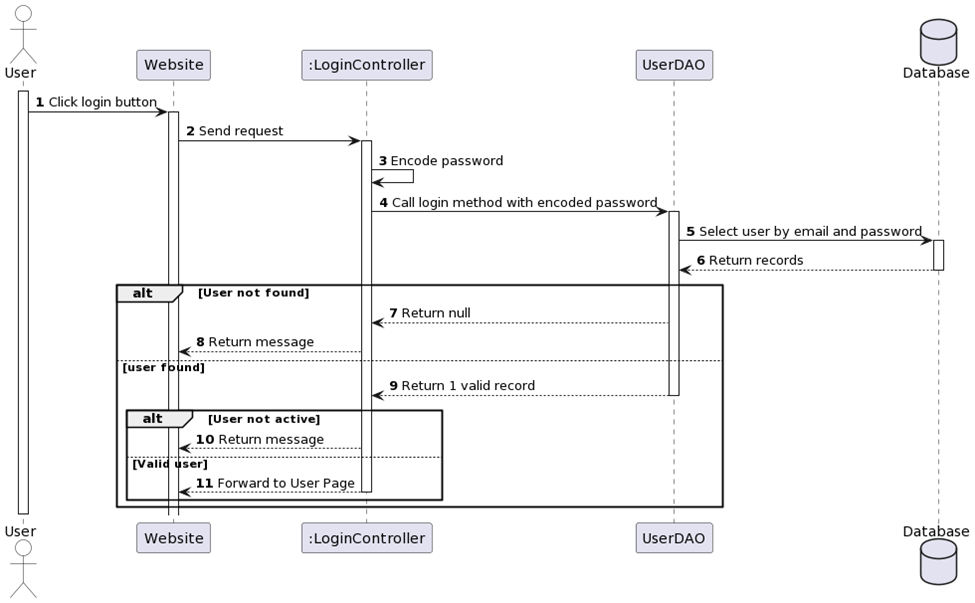
***User***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***DAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Check(String username, String password)* | *Check account in database to login* |

### c. Sequence Diagram(s)



### d. Database Queries

SELECT [User\_Id]\n"

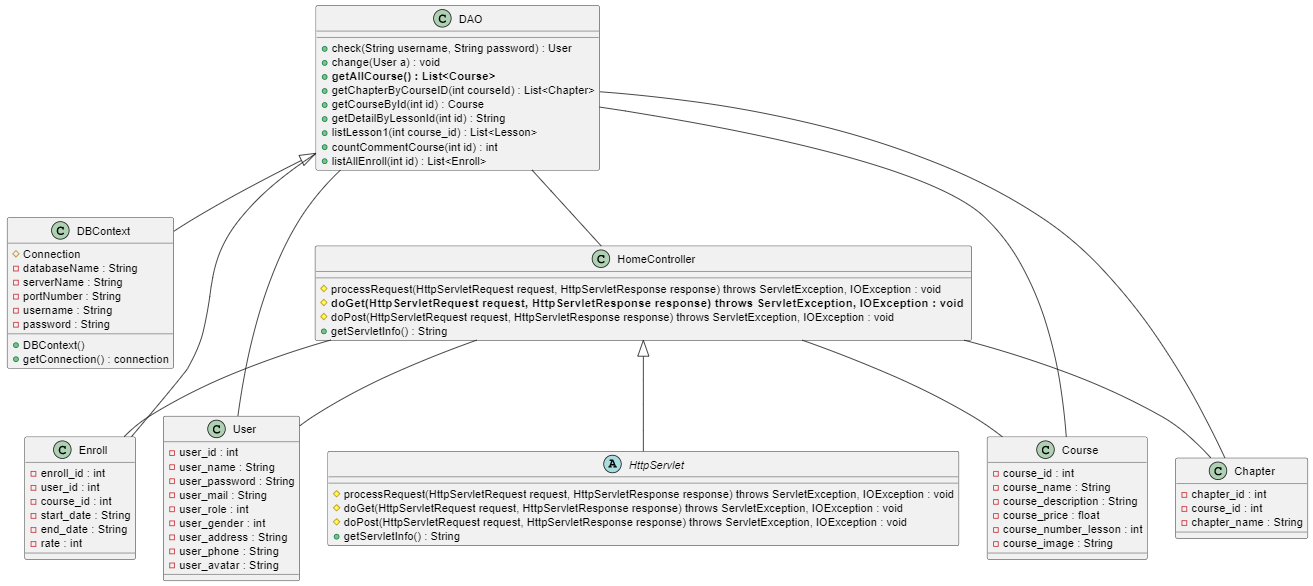
+ " ,[user\_name]\n"

+ " ,[user\_password]\n"

+ " where user\_name=? and user\_password=?

## 3. Home page

### a. Class Diagram



### b. Class Specifications

#### HomeController

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *List all courser from databse into homepage* |

#### DBContext

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***User***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Enroll***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Course***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Chapter***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***DAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *GetAllCourse(List<Course>)* | *Get all Cousre from database* |

### c. Sequence Diagram

Diagram

Description automatically generated

### d. Database Queries

SELECT TOP (1000) [Course\_id]\n"

+ " ,[course\_name]\n"

+ " ,[course\_description]\n"

+ " ,[course\_price]\n"

+ " ,[course\_number\_lesson]\n"

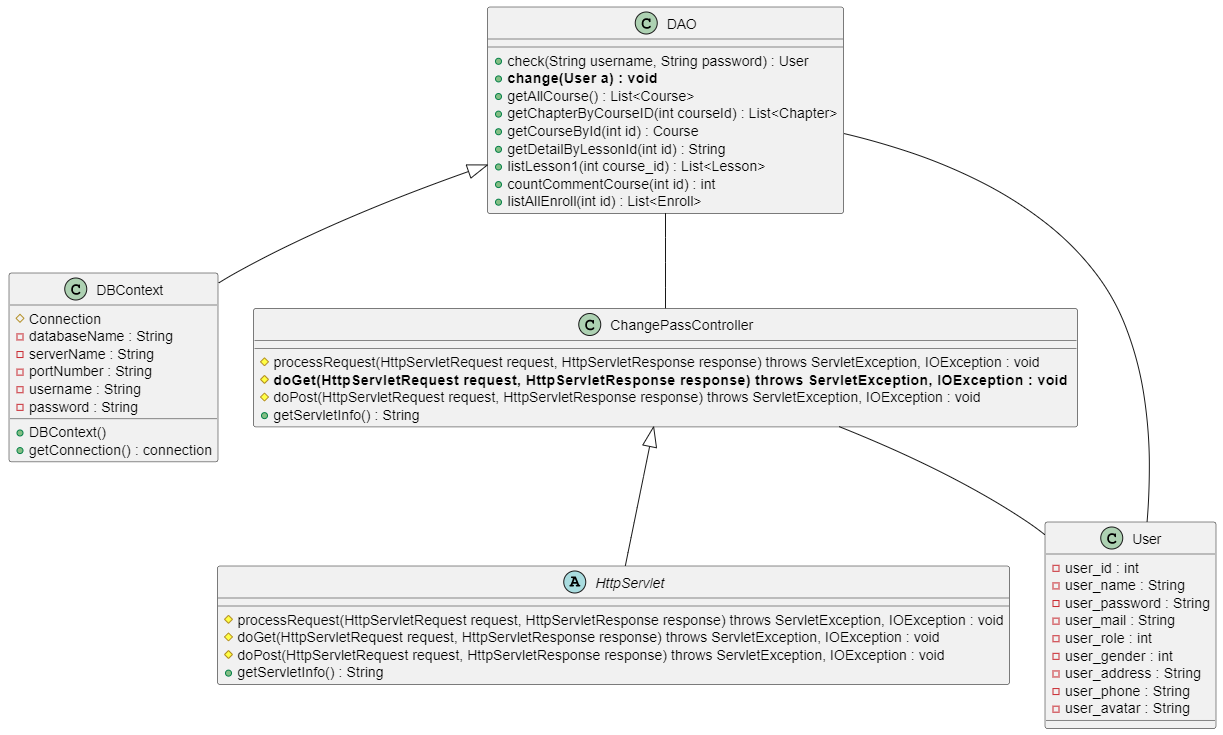
+ " ,[course\_image]\n"

+ " FROM [SWP-Project].[dbo].[Course]\n"

+ " where course\_price =0

## 4. ChangePass

### a. Class Diagram



### b. Class Specifications

#### ChangePassController

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Check user is existed or not, if not then you can change password* |

#### DBContext

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***User***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***DAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Change(User a)* | *Check user in database to change password* |

### c. Sequence Diagram(s)

Diagram, engineering drawing

Description automatically generated

### d. Database Queries

UPDATE [dbo].[User]\n"

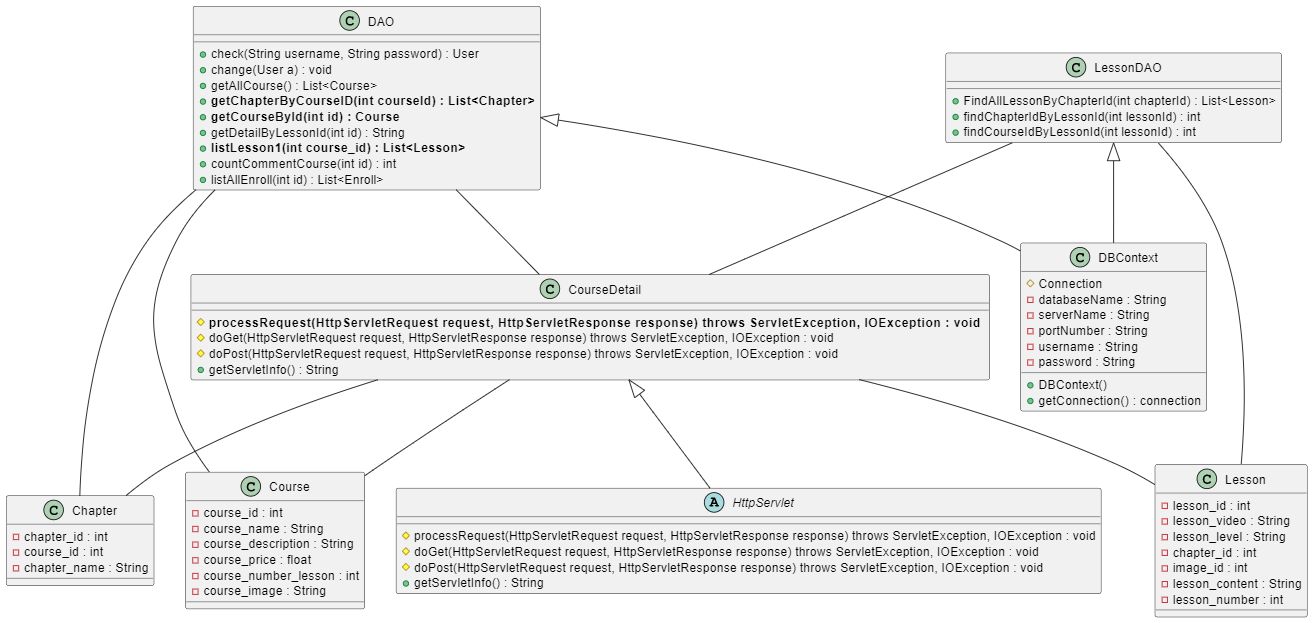
+ " SET \n"

+ " [user\_password] = ?\n"

+ " WHERE user\_name=?

## 5. CourseDetail

### a. Class Diagram



### b. Class Specifications

#### CourseDetail

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *processRequest(HttpServletRequest request, HttpServletResponse response)* | *Get Course by id from database and show in page* |

#### DBContext

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***Course***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Chapter***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Lesson***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***DAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getCourseById(int id)* | *Get course from database by id and display on website* |

### c. Sequence Diagram(s)

**Diagram

Description automatically generated**

### d. Database Queries

SELECT [Course\_id] ,[course\_name] ,[course\_description]\n"

+ " ,[course\_price]\n"

+ " ,[course\_number\_lesson]\n"

+ " ,[course\_image]\n"

+ " FROM [dbo].[Course]\n"

+ " where Course\_id=?

## 6. Lesson Detail

### a. Class Diagram

### b. Class Specifications

***LessonServlet***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doPost(HttpServletRequest request, HttpServletResponse response)* | *Get all lesson from course by FindAllLessonByChapterId* |
| *02* | *processRequest(HttpServletRequest request, HttpServletResponse response)* | *get a percentage at the end of a lesson* |

***DBContext***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***User***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Lesson***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Percentage***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***LessonDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *FindAllLessonByChapterId(int chapterId)* | *Get each lesson in each Chapter containing that lesson* |
| *02* | *getPercentageByUserID(int user\_id)* | *Get the percentage each time the user finishes each lesson* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

- SELECT [lesson\_id]\n"

+ " ,[lesson\_video]\n"

+ " ,[lesson\_level]\n"

+ " ,[chapter\_id]\n"

+ " ,[image\_id]\n"

+ " ,[lesson\_content]\n"

+ " ,[lesson\_number]\n"

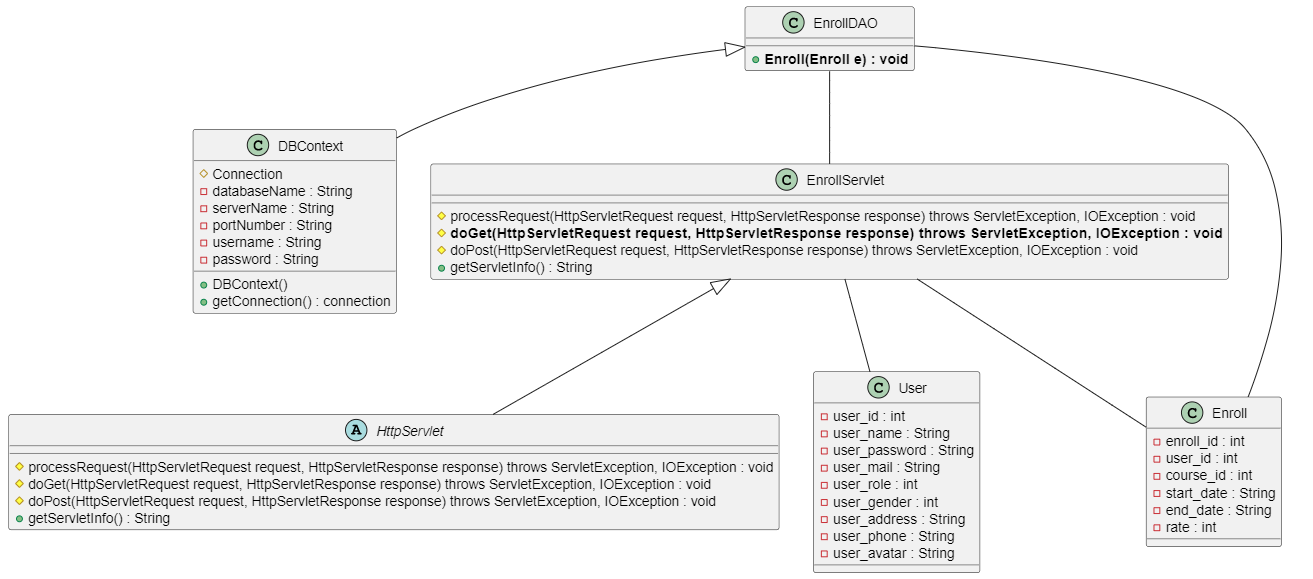
+ " FROM [SWP-Project].[dbo].[Lesson] where chapter\_id = ?

- SELECT \*\n"

+ " FROM [SWP-Project].[dbo].[Percentage] where user\_id =?

## 7. Enroll

### a. Class Diagram



### b. Class Specifications

#### EnrollServlet

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doPost(HttpServletRequest request, HttpServletResponse response)* | *Enroll the user with the course* |

#### DBContext

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***User***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Enroll***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***EnrollDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Enroll(Enroll e)* | *Get Enroll of user with course* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

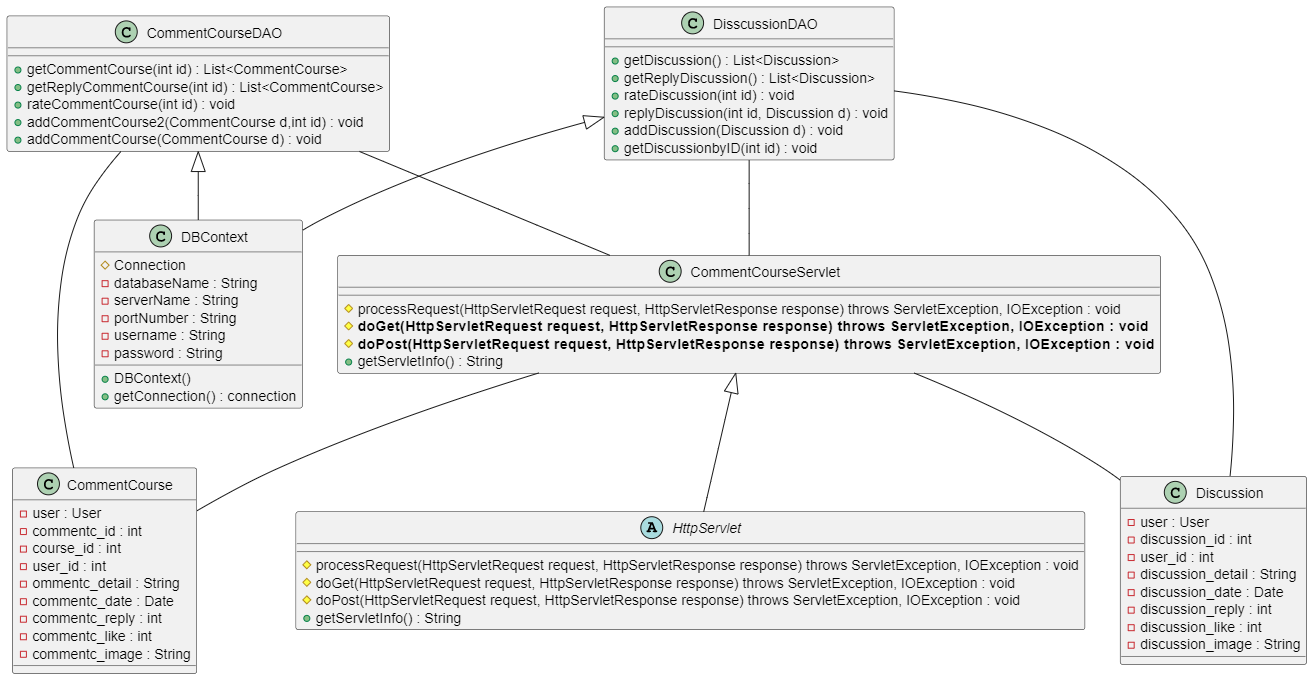
INSERT INTO [dbo].[Enroll]\n"

+ " ([user\_id]\n"

+ " ,[course\_id]) VALUES (?,?)”

## 8. Comment course

### a. Class Diagram



### b. Class Specifications

***CommentCourseServlet***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *processing comments for the course* |
| *02* | *doPost(HttpServletRequest request, HttpServletResponse response)* | *processing discussion for the course* |

***DBContext***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***CommentCourse***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Discussion***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***CommentDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getCommentCourse(int id)* | *Get Comment of user in CommentCourse* |
| *02* | *rateCommentCourse(int id)* | *handle the like for comments, the number of likes increases every time the like button is pressed* |

***DiscussionDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *addDiscussion(Discussion d)* | *add details of discussion* |
| *02* | *replyDiscussion(int id, Discussion d)* | *reply to the user's discussion* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

- Select \* from [CommentCourse] as CC\n"

+ " Join [User] as US on CC.[user\_id] =US.User\_Id\n"

+ " Join [Course] as CO on CC.course\_id =CO.Course\_id \n"

+ " where CC.comment\_repply=0 and CO.Course\_id=?

+ " UPDATE [dbo].[CommentCourse]\n"

+ " SET [comment\_like] +=1 WHERE [comment\_id]=?

+" INSERT INTO [dbo].[Discussion]\n"

+ " ([user\_id] ,[discussion\_detail ,[discussion\_date] ,[discussion\_image])\n"

+ " VALUES (?,?,?,?)

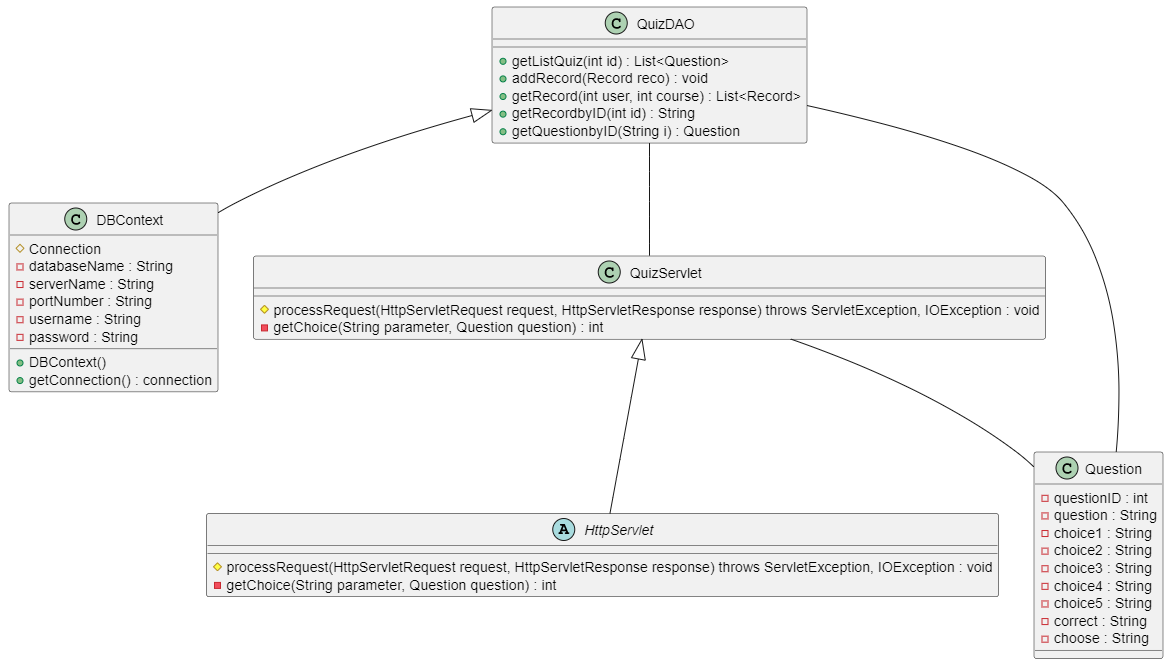
+ " INSERT INTO [dbo].[Discussion]\n"

+ " ([user\_id] ,[discussion\_detail] ,[discussion\_date] ,[discussion\_reply])\n"

+ " VALUES (?,?,?,?)"

## 9. Practice

### a. Class Diagram



### b. Class Specifications

**QuizzServlet**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getChoice(String parameter, Question question)* | *Get Question, choices and answer from database* |

**DBContext**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

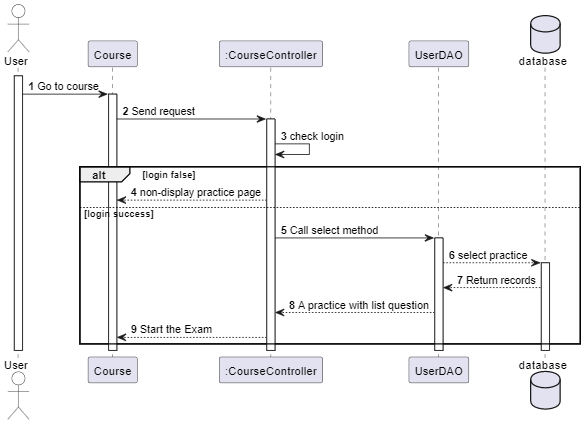
***Quizz***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***QuizzDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getListQuiz(int id)* | *Get List of quizz from database* |

### c. Sequence Diagram(s)



### d. Database Queries

Select TOP 30 \* from Question as qe\n"

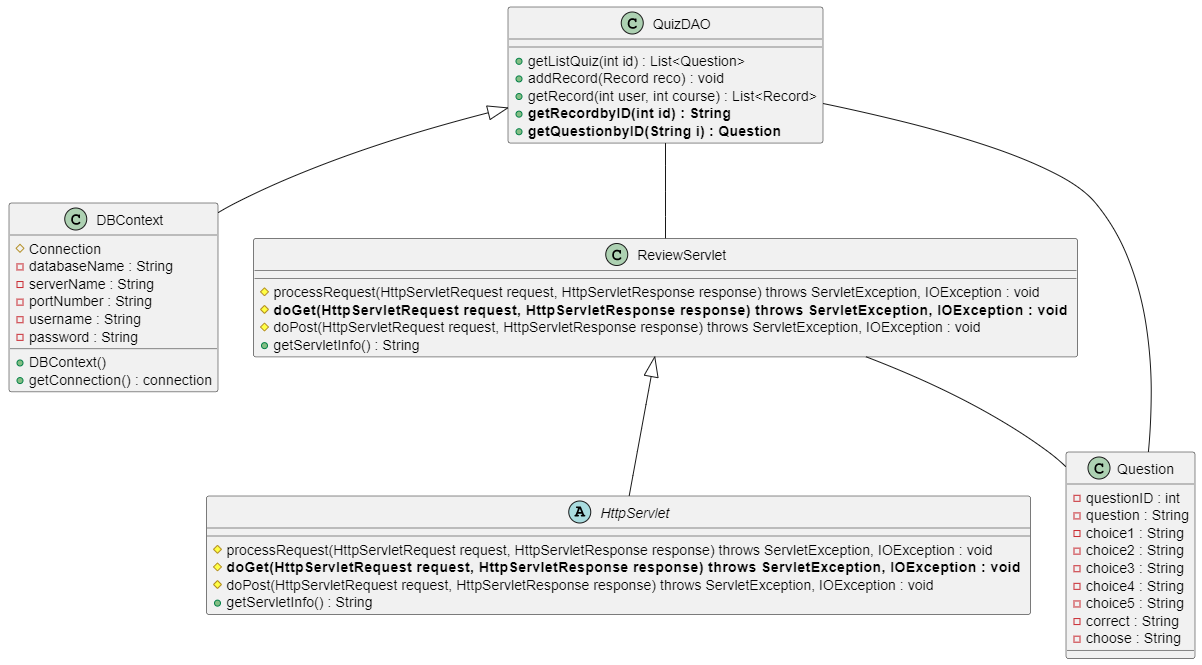
+ " Join [Quizz] as qi on qe.Question\_id=qi.Question\_id\n"

+ " where qi.Course\_id=?\n"

+ " ORDER BY NEWID()

## 10. Mark report

### a. Class Diagram



### b. Class Specifications

***ReviewServlet***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get a list of questions and answers for review* |

**DBContext**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***Question***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***QuizDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getRecordbyID(int id)* | *Get record of quizz to review* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

SELECT [record\_id]\n"

+ " ,[user\_id]\n"

+ " ,[Course\_id]\n"

+ " ,[record\_date]\n"

+ " ,[record\_mark]\n"

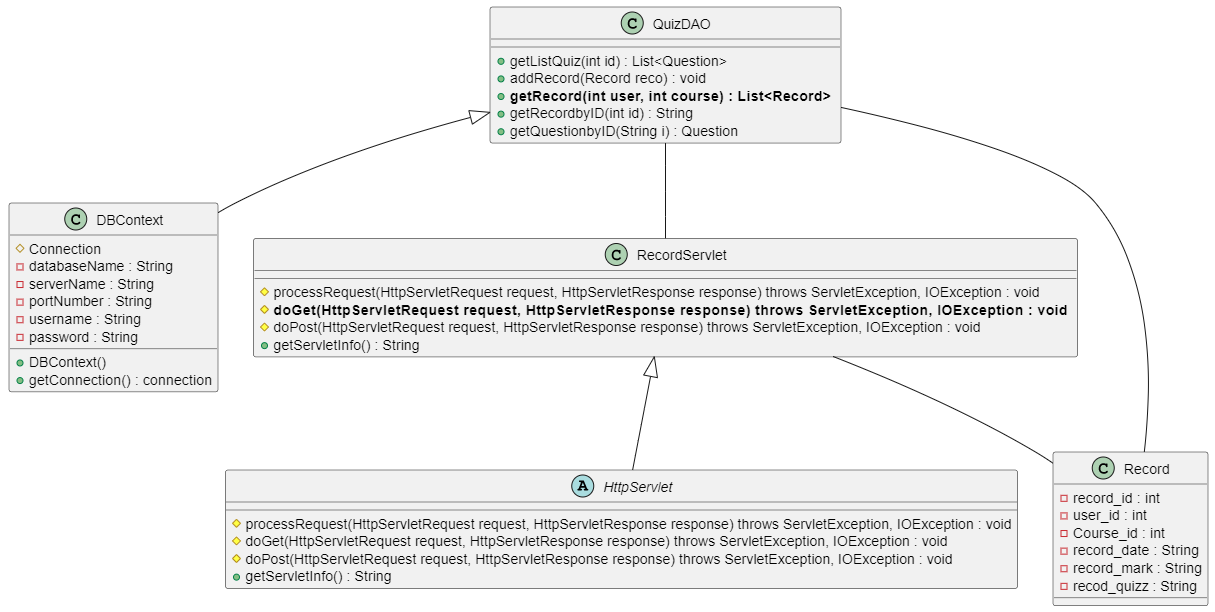
+ " ,[recod\_quizz]\n"

+ " FROM [dbo].[Record]\n"

+ " where [record\_id]=?

## 11. Review answer

### a. Class Diagram



### b. Class Specifications

***RecordServlet***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get the quizz's record* |

***DBContext***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***Question***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***QuizzDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getRecord(int user, int course)* | *Get record of Quizz* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

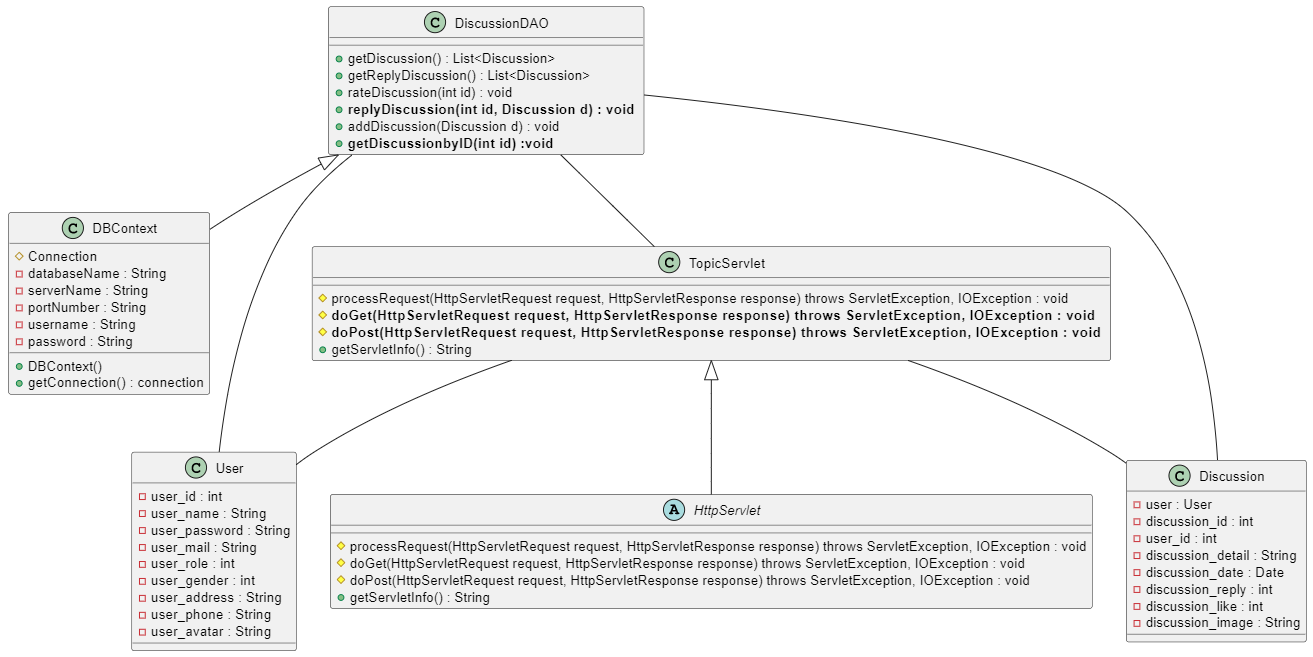
### d. Database Queries

Select \* from Record \n"

+ "where Record.user\_id=? and Record.Course\_id=?

## 12. Discussion

### a. Class Diagram



### b. Class Specifications

#### TopicServelet

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get discussion and reply of user* |
| *02* | *doPost(HttpServletRequest request, HttpServletResponse response)* | *Reply discussion with all discussion detail* |

#### DBContext

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***User***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Discussion***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***DiscussionDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getDiscussionbyID(int id)* | *Get Discussionof user by ID* |
| *02* | *getReplyDiscussion()* | *Get Reply Discussion of user* |
| *03* | *replyDiscussion(int id, Discussion d) {* | *Reply discussion with all discussion detail* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

- Select \* from [Discussion] as DI\n"

+ " Join [User] as US on DI.[user\_id] =US.User\_Id\n"

+ " where DI.discussion\_Id=?

- Select \* from [Discussion] as DI\n"

+ "Join [User] as US on DI.[user\_id] =US.User\_Id\n"

+ "where DI.discussion\_reply >0

- INSERT INTO [dbo].[Discussion]\n"

+ " ([user\_id]\n"

+ " ,[discussion\_detail]\n"

+ " ,[discussion\_date]\n"

+ " ,[discussion\_reply])\n"

+ " VALUES\n"

+ " (?,?,?,?)

## 13. Profile

### a. Class Diagram



### b. Class Specifications

**ProfileServlet**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get profile of course* |
|  |  |  |

***DBContext***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***User***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Course***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Chapter***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Enroll***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***DAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Change(User a)* | *Check user in database to change password* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

- GetAllCourse

SELECT TOP (1000) [Course\_id]\n"

+ " ,[course\_name]\n"

+ " ,[course\_description]\n"

+ " ,[course\_price]\n"

+ " ,[course\_number\_lesson]\n"

+ " ,[course\_image]\n"

+ " FROM [SWP-Project].[dbo].[Course]\n"

+ " where course\_price =0

- ListAllEnroll

SELECT TOP (1000) [Enroll\_id]\n"

+ " ,[user\_id]\n"

+ " ,[course\_id]\n"

+ " ,[start\_date]\n"

+ " ,[end\_date]\n"

+ " ,[rate]\n"

+ " FROM [SWP-Project].[dbo].[Enroll]\n"

+ " where [user\_id]=?

## 14. Search

### a. Class Diagram

Graphical user interface, application

Description automatically generated

### b. Class Specifications

**SearchCourseServlet**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get name and description of course* |

**SearchLessonServlet**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get name and description of Lesosn* |

**SearchChapterServlet**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get name and description of Chapter* |

**DBContext**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***DAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *SearchChapterByName* | *Fine all Chapter where chapter name like what you want to search* |
| *02* | *SearchCourse* | *Fine all Course where chapter name like what you want to search* |
| *03* | *SearchLesson* | *Fine all lesson where chapter name like what you want to search* |

***Course***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Lesson***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***Chapter***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

### c. Sequence Diagram(s)

Diagram, engineering drawing

Description automatically generated

### d. Database Queries

* SearchChapterByName:

SELECT [Chapter\_id]\n"

+ " ,[course\_id]\n"

+ " ,[chapter\_name]\n"

+ " FROM [dbo].[Chapter]\n"

+ " where 1=1

* SearchCourse:

SELECT [Course\_id]\n"

+ " ,[course\_name]\n"

+ " ,[course\_description]\n"

+ " ,[course\_price]\n"

+ " ,[course\_number\_lesson]\n"

+ " ,[course\_image]\n"

+ " FROM [dbo].[Course]\n"

+ " where 1=1

* SearchLesson:

SELECT [lesson\_id]\n"

+ " ,[lesson\_video]\n"

+ " ,[lesson\_level]\n"

+ " ,[chapter\_id]\n"

+ " ,[image\_id]\n"

+ " ,[lesson\_content]\n"

+ " ,[lesson\_number]\n"

+ " FROM [dbo].[Lesson]\n"

+ " where 1=1

## 15. User Manage

### a. Class Diagram

Diagram

Description automatically generated with medium confidence

### b. Class Specifications

***DBContext***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |

***UserDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *checkUser* | *Check all users exist in database already.* |
| *02* | *CheckUserById* | *Check only user have their id* |
| *03* | *addUser* | *Add user to the database* |
| *05* | *CheckUserGoogle* | *Check user login with google (check email has exist)* |
| *06* | *GetAllUser()* | *Take all user from database to the system.* |

***UserManage***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get list user from database to display in the screen.* |

***User***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
|  |  |  |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

- GET ALL USER:

SELECT [User\_Id]\n"

+ " ,[user\_name]\n"

+ " ,[user\_mail]\n"

+ " ,[user\_password]\n"

+ " ,[user\_role]\n"

+ " ,[user\_gender]\n"

+ " ,[user\_address]\n"

+ " ,[user\_phone]\n"

+ " ,[user\_avatar]\n"

+ " ,[user\_status]\n"

+ " FROM [dbo].[User]

## 16. Manage

### a. Class Diagram

### b. Class Specifications

***Manage***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get name, gender, email,address,phone,tong in database and show on manage page* |

***DBContext***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***ThongTin***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***DAOAdmin***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *numberOfStudent()* | *Check number of student* |
| *02* | *numberOfCourse()* | *Check number of course* |
| *03* | *numberOfEnroll()* | *Check number of enroll* |
| *04* | *getInfoCustomer()* | *Check all information of customer* |

### c. Sequence Diagram(s)

### Diagram Description automatically generatedd. Database Queries

*- numberOfStudent():*

*select count(u.User\_Id) from [User] as u where u.user\_role = 1*

*- numberOfCourse():*

*select count(Course\_id) from [Course]*

*- numberOfEnroll():*

*select count(Enroll\_id) from [Enroll]*

*- getInfoCustomer():*

select c.user\_gender,c.user\_name,c.user\_mail,c.user\_address,c.user\_phone, d.tong from \n"

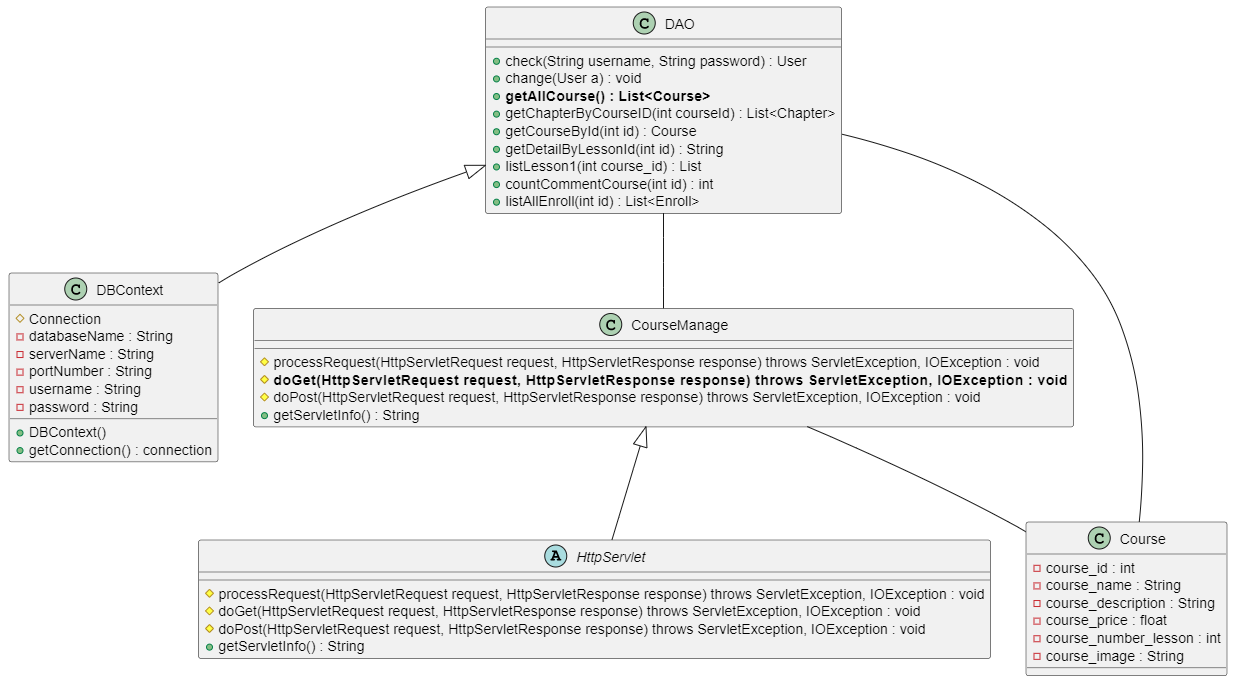
+ "(select u.User\_Id,u.user\_gender,u.user\_name,u.user\_mail,u.user\_address,u.user\_phone from \n"

+ "(select e.user\_id from Enroll as e group by e.user\_id) as a left join [User] as u on a.user\_id=u.User\_Id) as c ,\n"

+ "(select e.user\_id, count(e.user\_id) as tong from Enroll as e group by e.user\_id) as d where c.User\_Id=d.user\_id

## 17. CourseManage

### a. Class Diagram



### b. Class Specifications

**CourseManage**

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get all course in database and show on courseManager page* |

***DBContext***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***Course***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***DAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getAllCourse()* | *Get all course from database and display on page* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

- getAllCourse():

SELECT TOP (1000) [Course\_id]\n"

+ " ,[course\_name]\n"

+ " ,[course\_description]\n"

+ " ,[course\_price]\n"

+ " ,[course\_number\_lesson]\n"

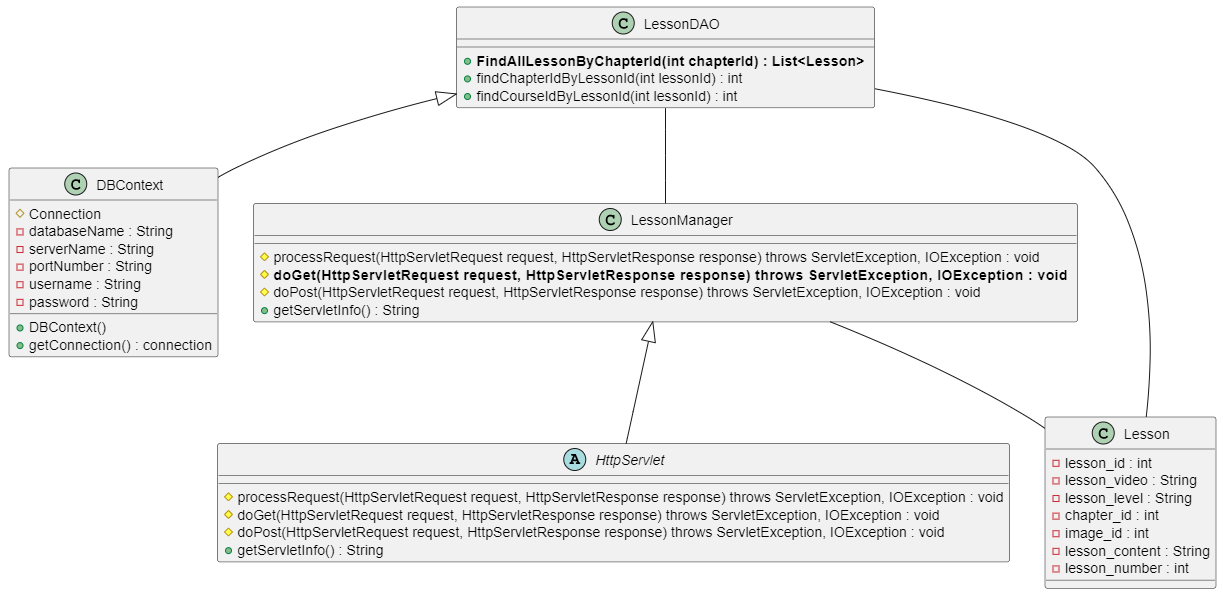
+ " ,[course\_image]\n"

+ " FROM [SWP-Project].[dbo].[Course]\n"

+ " where course\_price =0

## 18. LessonManager

### a. Class Diagram



### b. Class Specifications

#### LessonManager

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doGet(HttpServletRequest request, HttpServletResponse response)* | *Get all lesson in database and show on lesson manager page* |

#### DBContext

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getConnection()* | *To connect with database* |
| *02* | *closeResultSet(ResultSet resultSet)* | *To close result set* |
| *03* | *closePreparedStatement(PreparedStatement preparedStatement)* | *To close prepared statement* |
| *04* | *closeConnection(Connection connection)* | *To close connection* |

***Lesson***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *Constructors* | *Entity initialization* |
| *02* | *Getter* | *Get value of each attributes* |
| *03* | *Setter* | *Set value of each attributes* |

***LessonDAO***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *FindAllLessonByChapterId(int chapterId)* | *Get each lesson in each Chapter containing that lesson* |

### c. Sequence Diagram(s)

Diagram

Description automatically generated

### d. Database Queries

- FindAllLessonByChapterId(int chapterId):

SELECT [lesson\_id]\n"

+ " ,[lesson\_video]\n"

+ " ,[lesson\_level]\n"

+ " ,[chapter\_id]\n"

+ " ,[image\_id]\n"

+ " ,[lesson\_content]\n"

+ " ,[lesson\_number]\n"

+ " FROM [SWP-Project].[dbo].[Lesson] where chapter\_id = ?