

**ONLINE CARS SELLING SYSTEM**

**Software Design Specification**

– Hanoi, February 2023 –

Record of changeS

|  |  |  |  |
| --- | --- | --- | --- |
| Date | A\* M, D | In charge | Change Description |
| 01/02/2023 | A | HungNB | Add database design, class diagram and sequence diagram for login |
| 05/02/2023 | A | TuanNA | Add code package |
| 9/2/2022 | M | HungHM | Modified code package |
| 13/2/2022 | M | DatNA | Modified SQL Database |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

\*A - Added M - Modified D - Deleted

[I. Overview 5](#_Toc127743243)

[1. Code Packages 5](#_Toc127743244)

[2. Database Design 6](#_Toc127743245)

[a. Database Schema 6](#_Toc127743246)

[b. Table Description 6](#_Toc127743247)

[II. Code Designs 7](#_Toc127743248)

[1. Register 7](#_Toc127743249)

[a. Class Diagram 7](#_Toc127743250)

[b. Class Specifications 7](#_Toc127743251)

[c. Sequence Diagram(s) 9](#_Toc127743252)

[d. Database Queries 9](#_Toc127743253)

[2. Login 10](#_Toc127743254)

[a. Class Diagram 10](#_Toc127743255)

[b. Class Specifications 10](#_Toc127743256)

[c. Sequence Diagram(s) 11](#_Toc127743257)

[d. Database Queries 11](#_Toc127743258)

[3. Home Page 12](#_Toc127743259)

[4. Forgot Password 12](#_Toc127743260)

[5. Change Password 12](#_Toc127743261)

[6. Create Car 14](#_Toc127743262)

[a. Class Diagram 14](#_Toc127743263)

[b. Class Specifications 14](#_Toc127743264)

[c. Sequence Diagram(s) 16](#_Toc127743265)

[d. Database Queries 16](#_Toc127743266)

[7. View List Car 17](#_Toc127743267)

[a. Class Diagram 17](#_Toc127743268)

[b. Class Specifications 17](#_Toc127743269)

[c. Sequence Diagram(s) 18](#_Toc127743270)

[d. Database Queries 18](#_Toc127743271)

[8. Search Car 19](#_Toc127743272)

[a. Class Diagram 19](#_Toc127743273)

[b. Class Specifications 19](#_Toc127743274)

[c. Sequence Diagram(s) 20](#_Toc127743275)

[d. Database Queries 21](#_Toc127743276)

[9. View Detail Car 21](#_Toc127743277)

[a. Class Diagram 21](#_Toc127743278)

[b. Class Specifications 21](#_Toc127743279)

[c. Sequence Diagram(s) 22](#_Toc127743280)

[d. Database Queries 22](#_Toc127743281)

[10. Update Car 22](#_Toc127743282)

[a. Class Diagram 22](#_Toc127743283)

[b. Class Specifications 23](#_Toc127743284)

[c. Sequence Diagram(s) 24](#_Toc127743285)

[d. Database Queries 25](#_Toc127743286)

[11. Create Car 25](#_Toc127743287)

[a. Class Diagram 25](#_Toc127743288)

[b. Class Specifications 25](#_Toc127743289)

[c. Sequence Diagram(s) 27](#_Toc127743290)

[d. Database Queries 27](#_Toc127743291)

[12. View Detail Car 27](#_Toc127743292)

[13. Update Car 27](#_Toc127743293)

[14. Delete Car 27](#_Toc127743294)

[15. Search Car 27](#_Toc127743295)

[16. User Detail 27](#_Toc127743296)

[a. Class Diagram 27](#_Toc127743297)

[b. Sequence Diagram(s) 28](#_Toc127743298)

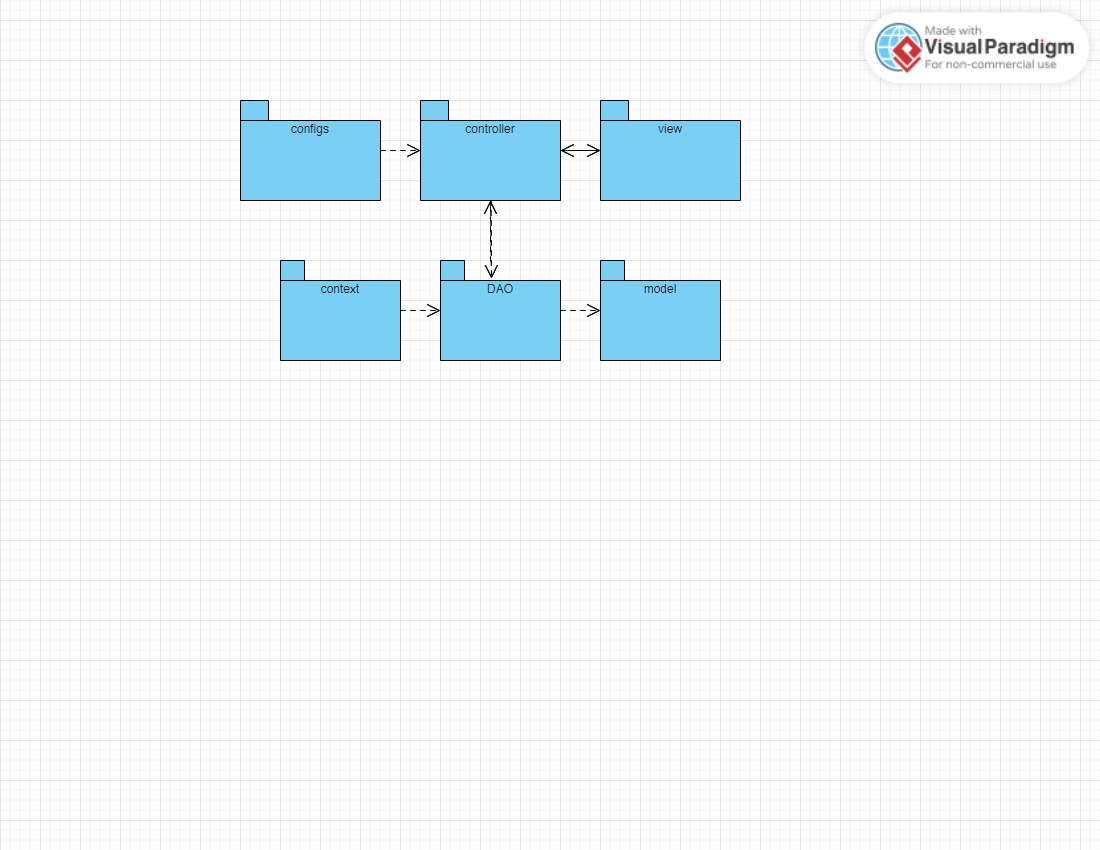
[17. Update Profile 30](#_Toc127743299)

[a. Class Diagram 30](#_Toc127743300)

[b. Sequence Diagram(s) 31](#_Toc127743301)

# I. Overview

## 1. Code Packages

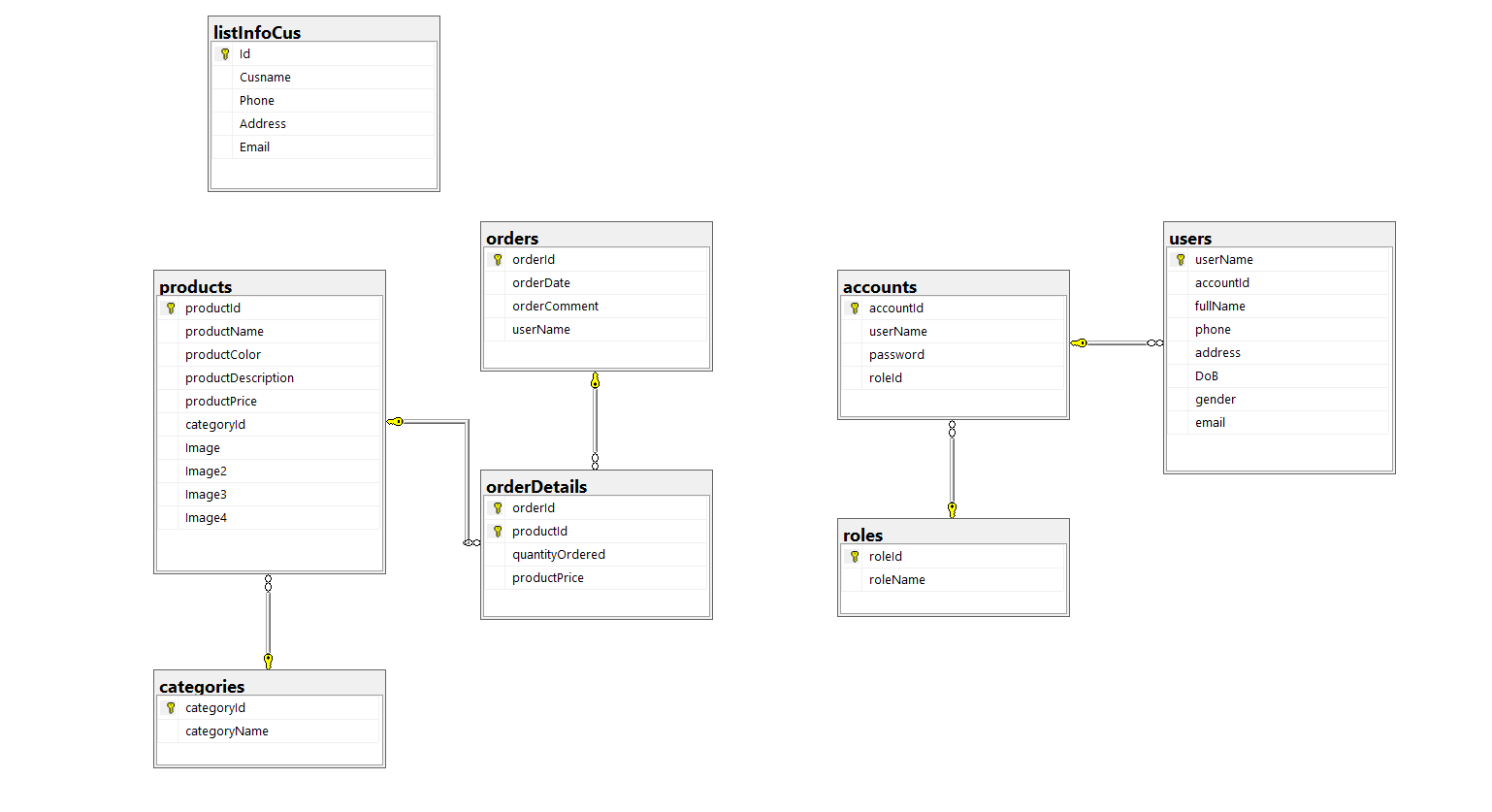


***Package descriptions***

|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| *01* | *controller* | *Acts as a go-between for the service layer and the view, passing on different input requests* |
| *02* | *DAO* | *This component is used to get data directly FROM the database* |
| *03* | *configs* | *This component helps define configuration* |
| *04* | *context* | *This component helps connect database* |
| *05* | *model* | *This component includes interfaces used to receive data retrieved FROM the database and classes used to receive data* |
| *06* | *view* | *This component display view.* |

## 2. Database Design

### a. Database Schema



### b. Table Description

|  |  |  |
| --- | --- | --- |
| **No** | **Table** | **Description** |
| *01* | *user* | *User of the system* |
| *02* | *Accounts* | *The management system of users of admin and manager* |
| *03* | *Roles* | *Account functions* |
| *04* | *Orders* | *Where contains product information: name, date, time, product code* |
| 05 | orderDetails | Contains price, quantity, product code |
| 06 | Products | Products are presented to users in this table |
| 07 | Categories | Where to classify vehicles and choose |
| 08 | listInfoCus | Manage contact information to care and consult with customers |

# II. Code Designs

## Register

### a. Class Diagram

**

### b. Class Specifications

#### RegisterController

***Class Methods***

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

#### DBContext

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **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* |

#### UserDAO

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *register(String email, String password, String fullName, int genderId, Timestamp createdAt, Timestamp updatedAt, boolean isActive)* | *Interface abstract method register with email, password, and fullName* |
| *02* | *isEmailExisted(String email)* | *Interface abstract method check email is existed in the system or not* |

#### UserDAOImpl

***Class Methods***

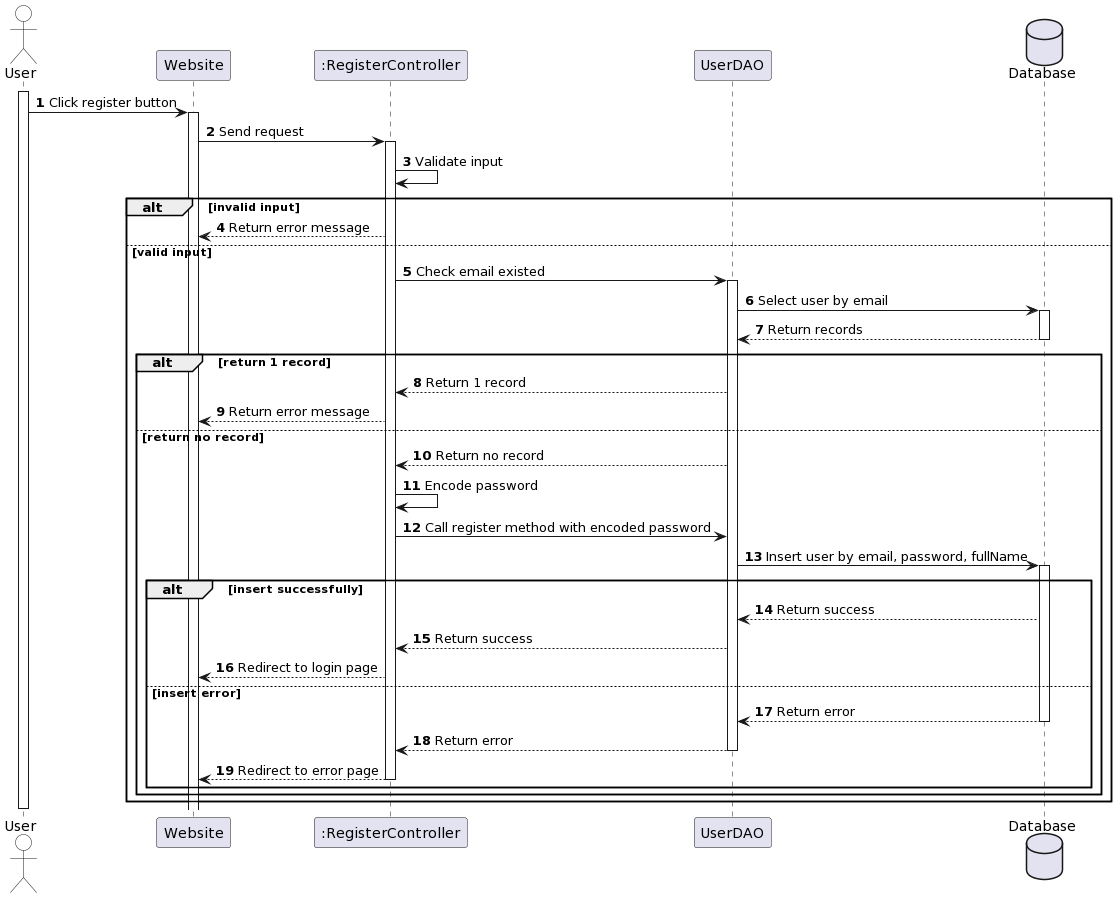
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *register(String email, String password, String fullName, int genderId, Timestamp createdAt, Timestamp updatedAt, boolean isActive)* | *Register account of the system with email, password and fullName* |
| *02* | *isEmailExisted(String email)* | *Check email is existed in the system or not* |

#### User

***Class Methods***

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

### c. Sequence Diagram(s)



### d. Database Queries

**- SELECT \* FROM [user] WHERE [email] = ?**

**- insert into [user] (email, password, full\_name, gender\_id, created\_at, updated\_at, is\_active) values (?, ?, ?, ?, ?, ?, ?)**

## Login

### a. Class Diagram



### b. Class Specifications

#### LoginController

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doPost(HttpServletRequest request, HttpServletResponse response)* | *Get email and password from Login Form and compare data in database. If user exist in database with correct password, user will login successfully* |

#### DBContext

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **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* |

#### UserDAO

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *login(String email, String password)* | *Interface abstract method login with email and password* |

#### UserDAOImpl

***Class Methods***

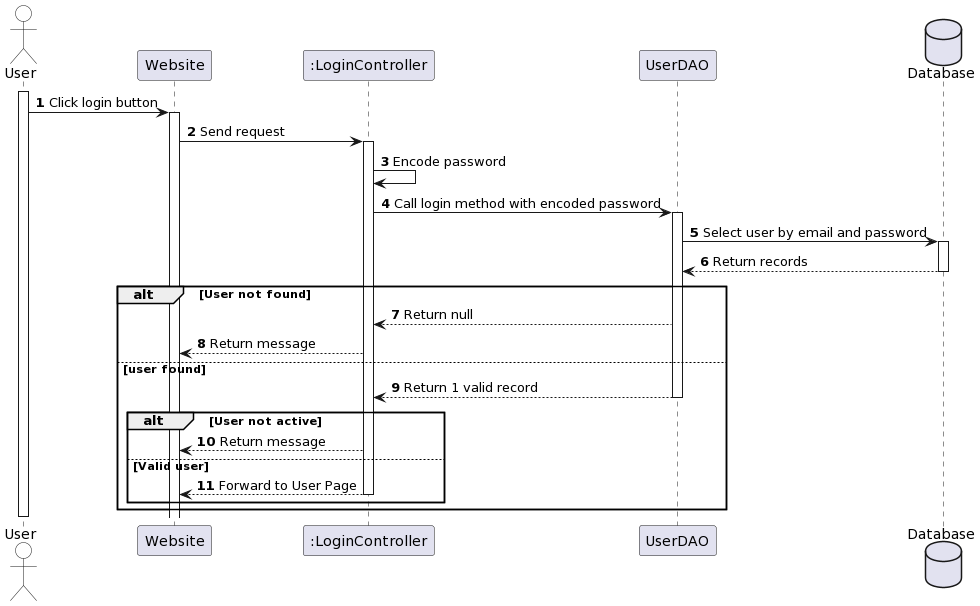
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *login(String email, String password)* | *Login to the system with email and password* |

#### User

***Class Methods***

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

### c. Sequence Diagram(s)



### d. Database Queries

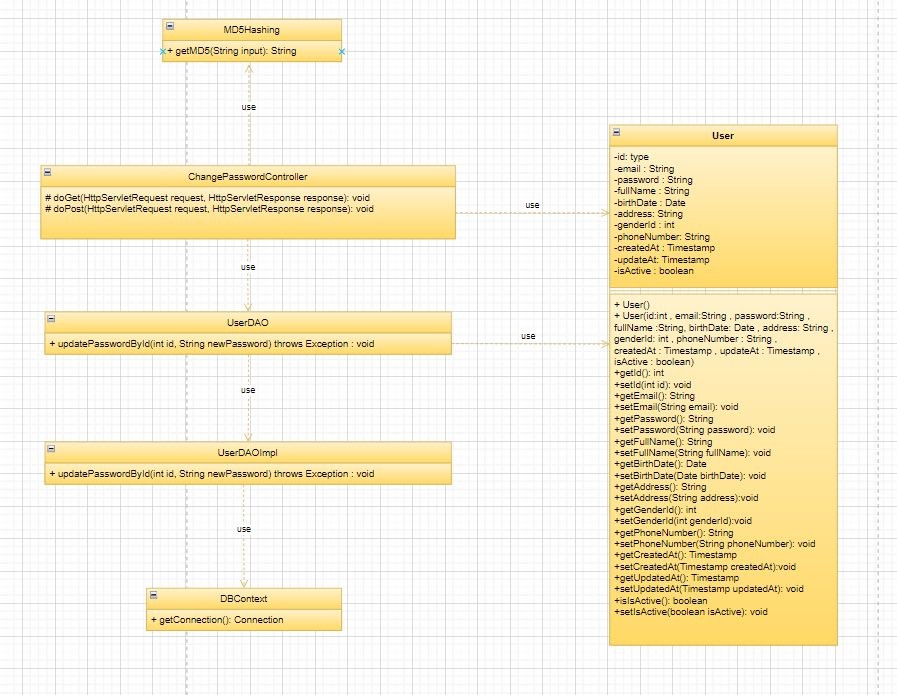
**SELECT \* FROM [user] WHERE [email] = ? and [password] = ?**

## Home Page

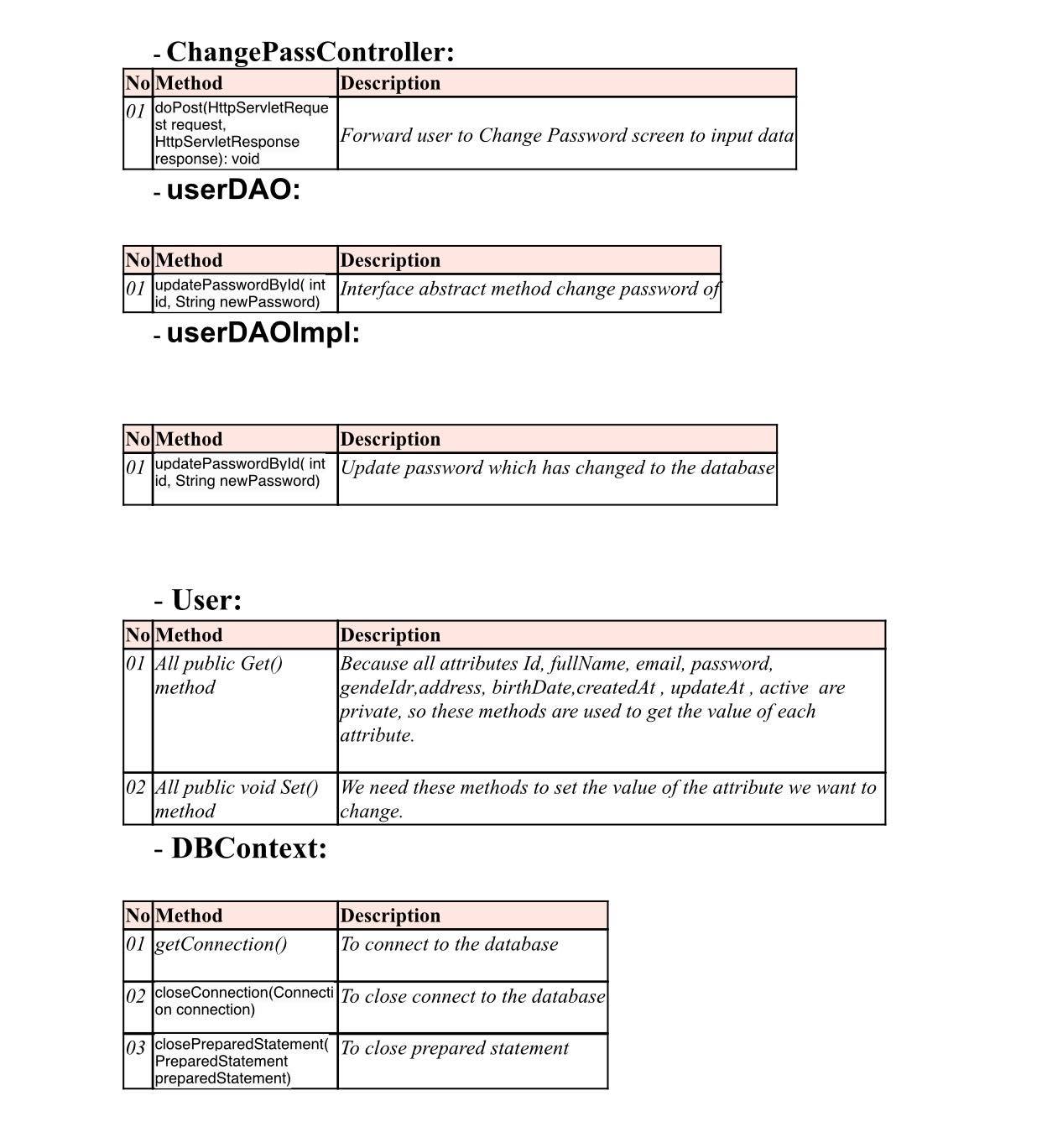
## Forgot Password

## Change Password

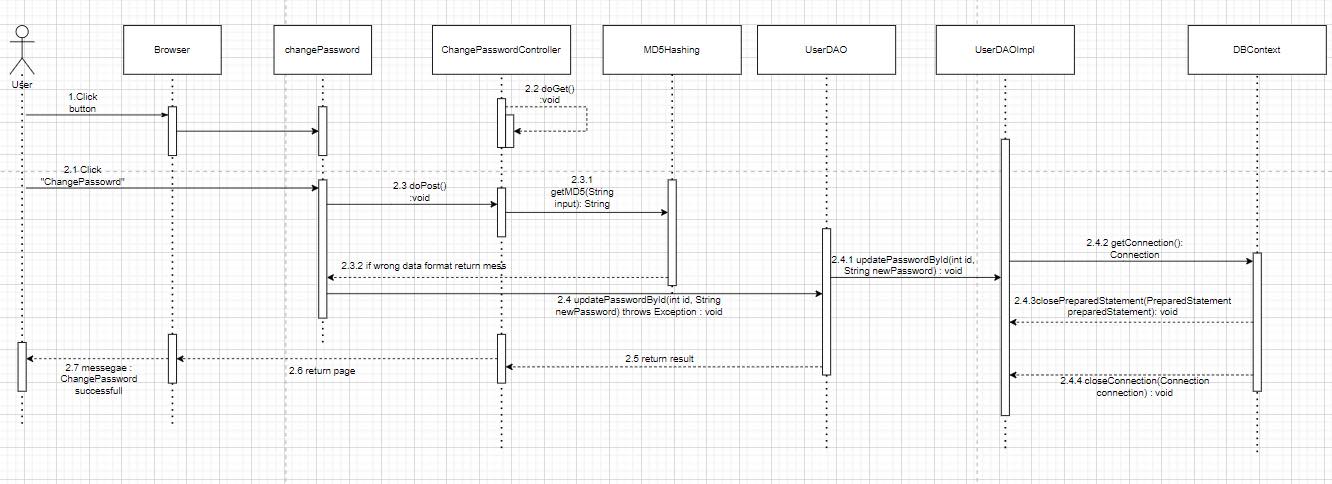
***a. Class Diagram***



***b. Class Specifications***



***c. Sequence Diagram(s)***

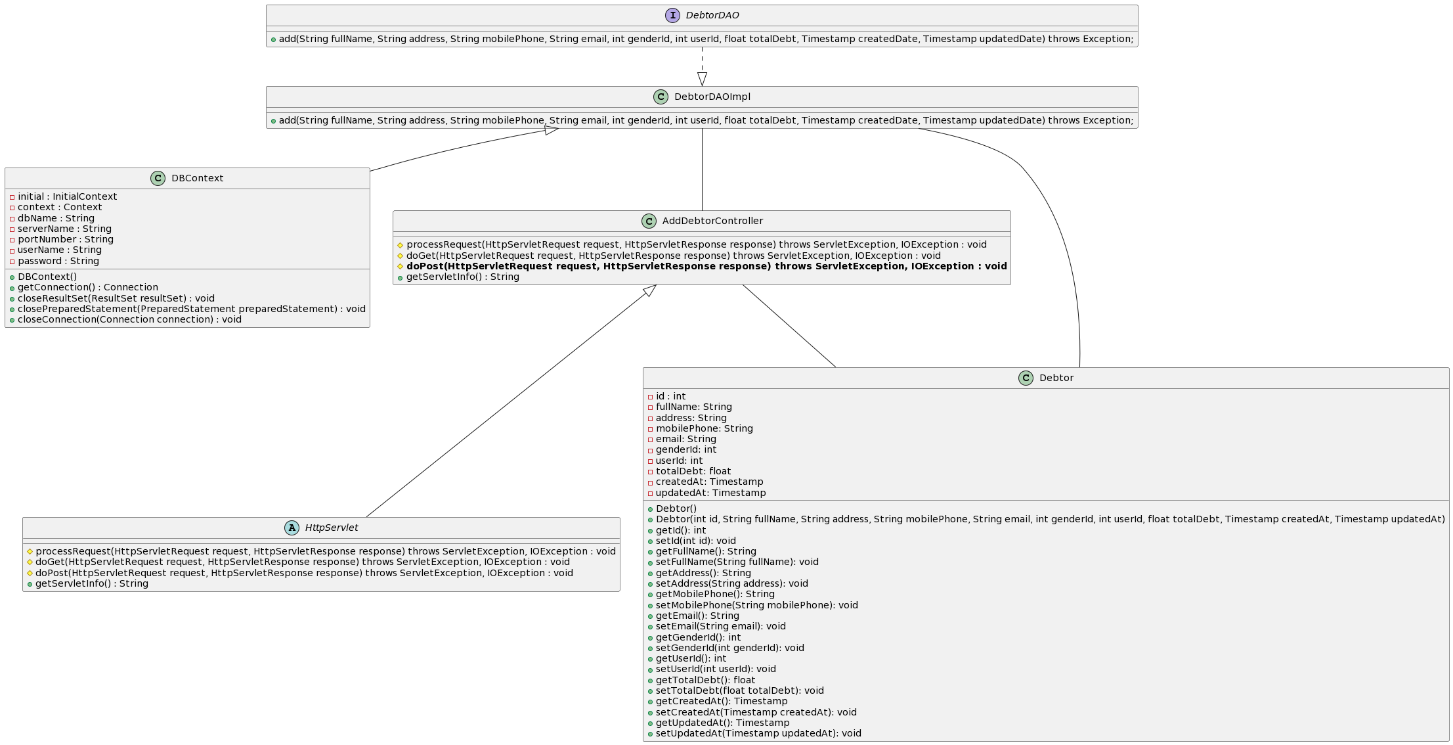


***d. Database Queries***

***SELECT \* FROM WHERE userId = ?;***

## Create Car

### a. Class Diagram



### b. Class Specifications

#### AddCarController

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *doPost(HttpServletRequest request, HttpServletResponse response)* | *Get id, name, address, email, gender, user id, total debt, created at, updatedAt from created debtor form and create a debtor from it. Then save new debtor to database.* |

#### DBContext

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **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* |

#### CarDAO

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *add(String fullName, String address, String email, int genderId, int userId, Timestamp createdAt, Timestamp updatedAt )* | *Interface abstract method add debtor with id, name, address, email, gender, user id, total debt, created at, updated at.* |

#### DebtorDAOImpl

***Class Methods***

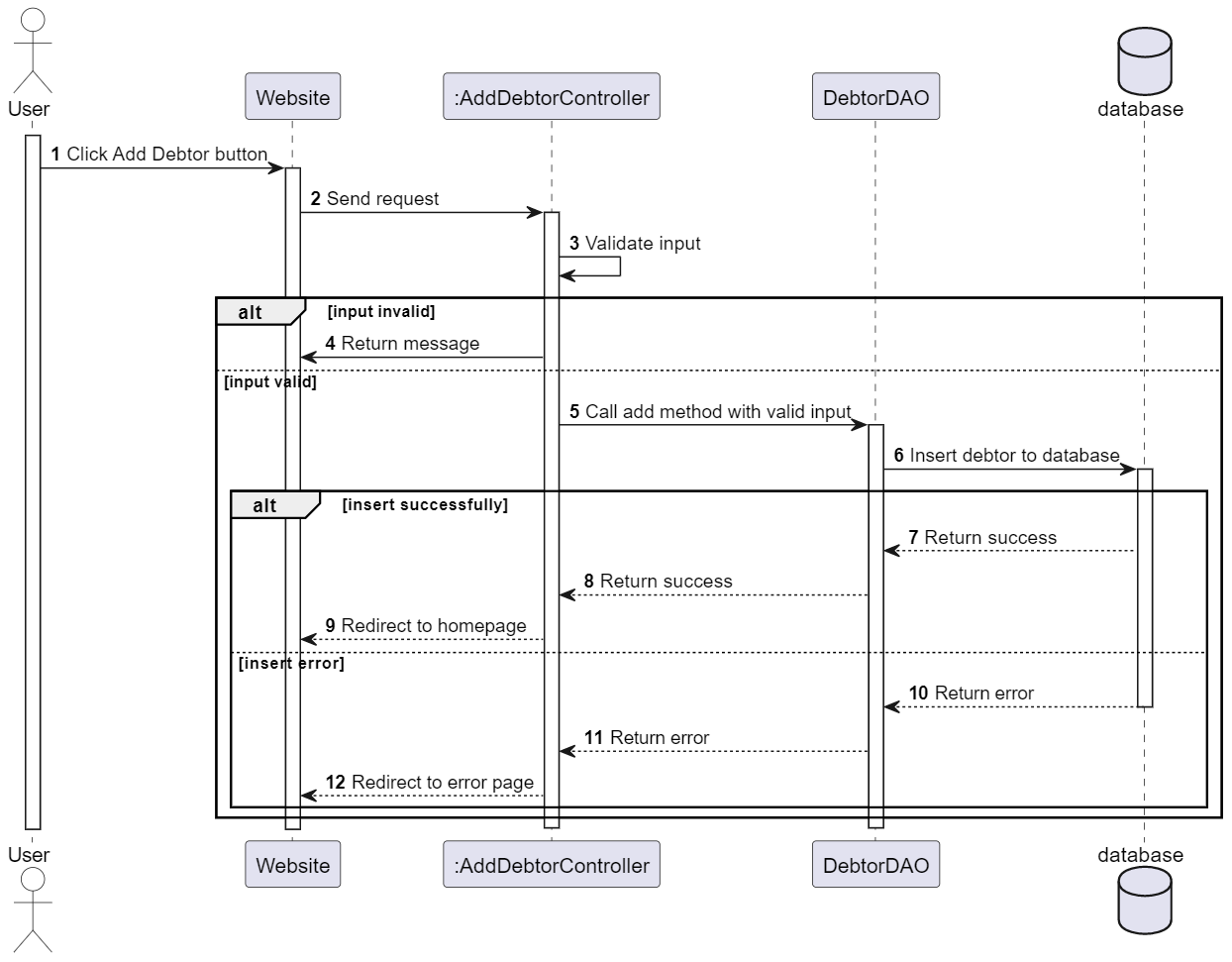
|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *add(String fullName, String address, String email, int genderId, int userId, Timestamp createdAt, Timestamp updatedAt )* | *Add debtor with id, fullName, address, email, genderId, userId, createdAt, updatedAt to database.* |

Car

***Class Methods***

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

### c. Sequence Diagram(s)



### d. Database Queries

**INSERT INTO [debtor] ([full\_name], [address] ,[mobile\_phone] ,[email] ,[gender\_id] ,[total\_debt] ,[created\_at] ,[updated\_at] ,[user\_id]) VALUES (?, ?, ? ,? ,? ,? ,? ,?, ?)**