SPRING ASSIGNMENT Day_7

Harshit Kushmakar| 16896

Q1.

User Class:

```
package com.springcore;

import org.hibernate.validator.constraints.NotBlank;

import javax.validation.constraints.*;
import java.util.List;

public class User {

    private Integer userId ;

    @NotBlank(message =            "Username   cannot   be
empty!!"                     ) ^[a-zA-    @Pattern(regexp   =   "
",message = "Invalid    Z]*$
Username")
    private String userName;

    @NotBlank(message = "Password cannot be empty!!" )
@Size(min = 8 , message = "Password must be greater
than 8 characters")
    private String password;

    @NotBlank(message = "Gender cannot be empty!!" )
private String gender;

    @Size(min = 2 ,message = "Minimum hobbies selected
should be 2")
    private List<String> hobbies;

    private String preferredHolidayLocation;

    @Min(value = 18)
private Integer age;
```

```
    private String dateOfBirth;

    static int count=100;

    public String getUserName() {
return userName;
    }
    public void setUserName(String userName) {
this.userName = userName;
    }
    public String getPassword() {
return password;
    }
    public void setPassword(String password) {
this.password = password;
    }
    public String getGender() {
return gender;
    }
    public void setGender(String gender) {
this.gender = gender;
    }
    public List<String> getHobbies() {
return hobbies;
    }
    public void setHobbies(List<String> hobbies) {
this.hobbies = hobbies;
    }
    public Integer getUserId() {
return userId;
    }
    public void setUserId(Integer count) {
this.userId = count+1;
    }
    public Integer getAge() {
return age;
    }
    public void setAge(Integer age) {
this.age = age;
    }
    public String getPreferredHolidayLocation() {
return preferredHolidayLocation;
    }
    public void setPreferredHolidayLocation(String
preferredHolidayLocation) {
        this.preferredHolidayLocation =
```

```
preferredHolidayLocation;
    }
    public String getDateOfBirth() {
return dateOfBirth;
    }
    public void setDateOfBirth(String dateOfBirth) {
this.dateOfBirth = dateOfBirth;
    }
    public User(){
        this.userId= count++;

    }
    public User(String userName, String password, String
gender, List<String> hobbies, String
preferredHolidayLocation, Integer age, String
dateOfBirth) {
        this.userId= count++;
this.userName = userName;
this.password = password;
this.gender = gender;
this.hobbies = hobbies;
this.preferredHolidayLocation =
preferredHolidayLocation;
this.age = age;
        this.dateOfBirth = dateOfBirth;
    }

    @Override
    public String toString() {
return "User{" +
                "userId=" + userId +
                ", userName='" + userName + '\'' +
                ", password='" + password + '\'' +
                ", gender='" + gender + '\'' +
                ", hobbies=" + hobbies +
                ", preferredHolidayLocation='"
+ preferredHolidayLocation + '\'' +
", age=" + age +
                ", dateOfBirth='" + dateOfBirth + '\'' +
'}';

    }

}
```

User Controller Class:

```
package com.springcore;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model; import
org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.*; import
org.springframework.web.servlet.ModelAndView;

import javax.validation.Valid; import
java.util.List;

@Controller
public class UserController {

//    @RequestMapping(value = "/", method =
RequestMethod.GET)
//    public String displayLogin(Model model) {
//
//        return "index";
//
//    }

//
//    @RequestMapping(value = "/userDetail", method =
RequestMethod.POST) //
public ModelAndView
getUserData(@RequestParam("userName") String userName ,
@RequestParam("password") String password ,
@RequestParam("gender") String gender ,
@RequestParam("hobbies") List<String> hobby ,
@RequestParam("preferredHolidayLocation") String
preferredHolidayLocation, @RequestParam("dateOfBirth")
String dob ){
//
//
//        AgeCalculation age = new AgeCalculation(); //
DateEditor dateEditor = new DateEditor();
//        dateEditor.setAsText(dob);
//
```

```
//         User user = new User(userName,password,gender,
hobby, preferredHolidayLocation,age.age(dob) , (String)
dateEditor.getValue());
```

```
//         ModelAndView modelAndView = new
ModelAndView("userDetails");
//
//         modelAndView.addObject("user", user);
//         return modelAndView;
//
//     }

    @RequestMapping("/Form")
    public String showForm(Model theModel) {
theModel.addAttribute("user", new User());
return "login";
    }


    @RequestMapping(value = "/userForm",method =
RequestMethod.POST)
    public String processForm(@Valid
@ModelAttribute("user") User user , BindingResult result,
ModelMap model) {

        AgeCalculation ageCalculation = new
AgeCalculation();
int age =
ageCalculation.age(user.getDateOfBirth());
user.setAge(age);

        if (result.hasErrors() || age < 21) {
return "login";
        } else {
            return "userdata";
        }
    }

}
```

AgeCalculation:

```java
package com.springcore;

import java.time.LocalDate;

public class AgeCalculation {

    public int age(String date) {
        Integer currYear = LocalDate.now().getYear();
        Integer dobYear =
Integer.parseInt(date.substring(0, 4));
return currYear - dobYear;
    }

}
```

index.jsp

```html
<html>
<head>
    <title>User Registration</title>
</head>
<body>
<h1> User Registration Form </h1>

<a href='Form'> Click here </a>

</body>
</html>
```

Login.jsp:

```jsp
<%@ taglib prefix="form"
uri="http://www.springframework.org/tags/form" %>
```

```
<html>
<head>
    <title>User Registration</title>
     <style>
        .error
{
            color: #ff0000;
font-weight: bold;
        }
     </style>
</head>
<body>
<h1> User Registration Form </h1>
<form:form method="POST" action="userForm"
modelAttribute="user">

    User name: <form:input   path="userName"  />
    <form:errors path="userName" cssClass="error" />
<br><br>

    Password : <form:password  path="password" />
    <form:errors path="password" cssClass="error" />
<br><br>

    Gender :
    Male   <form:radiobutton path="gender" value="Male"
/>
    Female <form:radiobutton path="gender" value="Female"
/>
    <form:errors path="gender" cssClass="error" />
<br><br>

    Hobbies :
    Travelling <form:checkbox path="hobbies"
value="travelling"/><br><br>
    Sleeping <form:checkbox path="hobbies"
value="sleeping"/><br><br>
    Swimming <form:checkbox path="hobbies"
value="swimming"/>
    <form:errors path="hobbies" cssClass="error" />
<br><br>

    Preferred Holiday Location : <form:select
path="preferredHolidayLocation">
```

```
    <form:option
            value="Singapore"></form:option>
<form:option
```

```
            value="Bangkok"></form:option>
    <form:option
            value="Switzerland"></form:option>
</form:select>

    <br><br>

    Date of Birth : <form:input path="dateOfBirth"
type="date"/><br><br>

    <input type="submit" value="Submit" />
</form:form>
</body>
</html>
```

UserData.jsp:

```
<%@ page isELIgnored = "false" %>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="ISO-8859-1">
        <title>User Data</title>
    </head>
    <body>

       <h3>User Details Submitted Successfully!!</h3>
       <table>
            <tr>
                <td>User Id:</td>
                <td>${user.userId}</td>
            </tr>
            <tr>
                <td>User Name:</td>
                <td>${user.userName}</td>
            </tr>

            <tr>
                <td>Your Password:</td>
<td>${user.password}</td>             </tr>
```

```
            <tr>
                <td>Gender:</td>
                <td>${user.gender}</td>
            </tr>


            <tr>
                <td>Hobbies:</td>
                <td>${user.hobbies}</td>
            </tr>


            <tr>
                <td>Preferred Holiday Location:  </td>
                <td>${user.preferredHolidayLocation}</td>
            </tr>


            <tr>
                <td>Age:</td>
                <td>${user.age}</td>
            </tr>

            <tr>
                <td>DateOfBirth:</td>
                <td>${user.dateOfBirth}</td>
            </tr>

         </table>
    </body>

</html>
```

Output:

Q2.

Index.jsp

```
<html>
<body>
<h2><a href="customer/register">Click Here to Register the
Customer</a></h2>
<h2><a href="customer/applyLoan">Click Here to Apply for
Loan </a></h2>
<h2><a href="customer/search">Click Here to Search for
Details</a></h2>
</body>
</html>
```

Controller Class:

```
package com.springprj.controller;

import javax.validation.Valid;
 import com.springprj.dao.CustomerDAO;
import com.springprj.dao.LoanDAO; import
com.springprj.model.Customer; import
com.springprj.model.LoanAgreement;
import
org.springframework.beans.propertyeditors.CustomDateEdito
r;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import
org.springframework.web.bind.annotation.InitBinder;
import
org.springframework.web.bind.annotation.ModelAttribute;
import
org.springframework.web.bind.annotation.RequestMapping;
```

```
import
org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;

import java.text.SimpleDateFormat;
import java.util.Date;


@Controller
@RequestMapping("/customer")
public class ControllerClass {

    @InitBinder
    public void initBinder(WebDataBinder dataBinder) {
        SimpleDateFormat dateFormat = new
SimpleDateFormat("dd-MM-yyyy");
        dataBinder.registerCustomEditor(Date.class, new
CustomDateEditor(dateFormat, true));
    }


    @RequestMapping("/register")
    public String showForm(Model theModel) {
theModel.addAttribute("customer", new Customer());
        return "registercustomer";
    }


    @RequestMapping("/applyLoan")
    public String showApplyForm(Model theModel) {
        theModel.addAttribute("loan", new
LoanAgreement());
        return "applyloan";
    }


    @RequestMapping("/search")
    public String search(Model theModel) {
theModel.addAttribute("page","display");
return "search";
    }
    @RequestMapping("/processCustomerForm")
public String processCustomerForm(@Valid
```

[redacted]

[redacted]

[redacted]

```
@Modtribute("customer") Customer customer,
BindingResult theBindingResult) {
if (theBindingResult.hasErrors()) {
return "registercustomer";
        } else {
            CustomerDAO customerDAO = new CustomerDAO();
customerDAO.save(customer);

            return "display";
        }

    }



    @RequestMapping("/processLoanForm")
public String processLoanForm(@Valid
@ModelAttribute("loan") LoanAgreement
loanAgreement, BindingResult theBindingResult) {
if (theBindingResult.hasErrors()) {
return "applyloan";
        } else {
            LoanDAO loanDAO = new LoanDAO();
loanDAO.save(loanAgreement);
return "display1";
        }
    }

    @RequestMapping("/searchByCustomerCode")
public ModelAndView
searchByCustomerCode(@RequestParam("lesseeId")String id )
{



        LoanDAO loanDAO = new LoanDAO();
ModelAndView modelAndView = new
ModelAndView("display2");
        String loanData = loanDAO.select(id);

        modelAndView.addObject("loan",loanData);
return modelAndView;

    }
    @RequestMapping("/searchByLoanId")
public ModelAndView
searchByLoanId(@RequestParam("agreementId") String id ) {
        ModelAndView modelAndView = new
ModelAndView("display3");
        LoanDAO loanDAO = new LoanDAO();
        String loanData = loanDAO.search(id);

        modelAndView.addObject("customerData",loanData);
return modelAndView;

    }



}
```

CustomerDAO

```
package com.springprj.dao;

import com.springprj.model.Customer;

import java.sql.*;
import java.text.ParseException;
import
java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date; import
java.util.List;

public class CustomerDAO {

    private  ArrayList<Customer> customers = new
ArrayList<>();

    Connection con = null;
    Statement stmt=null;

    public CustomerDAO(){

        try {
```

```
Class.forName("oracle.jdbc.driver.OracleDriver");
con = DriverManager.getConnection(

"jdbc:oracle:thin:@10.1.50.198:1535:nsbt19c", "sh",
"sh");
            stmt = con.createStatement();
        }
         catch (Exception e)
{
            e.printStackTrace();
        }
    }
    public void conn() {
         try
{
Class.forName("oracle.jdbc.driver.OracleDriver");

            con = DriverManager.getConnection(

"jdbc:oracle:thin:@10.1.50.198:1535:nsbt19c", "sh",
"sh");
            stmt = con.createStatement();
        }
         catch (Exception e)
{
            e.printStackTrace();
        }

    }


    public void save(Customer c) {
conn();

        try {

            Date d = new SimpleDateFormat("yyyy-MM-
dd").parse(c.getDateOfBirth());
            String sd = new
SimpleDateFormat("dd/MM/yyyy").format(d);
```

```
            String s = "insert into Customer_16899
```

```
values('"
                    + c.getCustomerId() + "','" +
c.getFirstName() + "','"
                    + c.getLastName() + "','" +
c.getGender() + "','"  + sd + "','"
                    + c.getContactNumber() + "','"
+c.getEmailAddress() + "'," + c.getMonthlyIncome() + ",'"
                    + c.getProfession() + "'," +
c.getTotalMonthlyExpense() + ",'" + c.getDesignation() +
"','" +c.getCompanyName() +"')";

            stmt.execute(s);

        }
         catch (SQLException e1) {
e1.printStackTrace();          } catch
(ParseException e) {              throw
new RuntimeException(e);
        }

    }

    public String
selectAll(){          conn();
try {

            PreparedStatement pstmt =
con.prepareStatement("select * from Customer_16899 ");
            ResultSet rs = pstmt.executeQuery();

            while (rs.next()) {

                customers.add((new
Customer(rs.getString(1), rs.getString(2),
rs.getString(3), rs.getString(4),
rs.getDate(5).toString(), rs.getString(6),
rs.getString(7), rs.getLong(8), rs.getString(9),
rs.getDouble(10), rs.getString(11), rs.getString(12))));
            }

            if(customers.isEmpty()){
return "Data do not exist";
```

```
}

return customers.toString();
```

```
        }

        catch (SQLException e1) {
            return "Database Connection Error";
        }



    }

    public String getAllCustomers(){
selectAll();
        return customers.toString();
    }



    public List<Customer> select(String id) {

        conn();
        ArrayList<Customer> customers1 = new
ArrayList<>();

        try {

            PreparedStatement pstmt =
con.prepareStatement("select * from Customer_16899 where
customer_id = ? ");

            pstmt.setString(1,id);

            ResultSet rs = pstmt.executeQuery();

            while (rs.next()) {

                customers1.add((new
Customer(rs.getString(1), rs.getString(2),
rs.getString(3), rs.getString(4),
rs.getDate(5).toString(), rs.getString(6),
rs.getString(7), rs.getLong(8), rs.getString(9),
rs.getDouble(10), rs.getString(11), rs.getString(12))));
```

```
}
```

```
            return customers1;

        }
         catch (SQLException e1) {
return new ArrayList<>();
        }

    }


    public String select1(String id) {

        conn();
        ArrayList<Customer> customers1 = new
ArrayList<>();

        try {

            PreparedStatement pstmt =
con.prepareStatement("select * from Customer_16899 where
customer_id = ? ");

            pstmt.setString(1,id);

            ResultSet rs = pstmt.executeQuery();

            while (rs.next()) {

                customers1.add((new
Customer(rs.getString(1), rs.getString(2),
rs.getString(3), rs.getString(4),
rs.getDate(5).toString(), rs.getString(6),
rs.getString(7), rs.getLong(8), rs.getString(9),
rs.getDouble(10), rs.getString(11), rs.getString(12))));
            }

            return customers1.toString();

        }
        catch (SQLException e1) {
            return new ArrayList<>().toString();
        }

    }


    public String delete(String id) {

        conn();

        try {

            PreparedStatement pstmt =
con.prepareStatement("Delete from Customer_16899 where
customer_id = ? ");

            pstmt.setString(1,id);

            ResultSet rs = pstmt.executeQuery();

            return selectAll();

        }
        catch (SQLException e1) {
            return "Database connection error";
        }

    }
 }
```

LoanDAO

```
package com.springprj.dao;
```

[redacted]

[redacted]

```
import com.springprj.model.LoanAgreement;

import java.sql.*;
import java.text.ParseException;
import
java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;

public class LoanDAO {

    Connection con = null;
    Statement stmt=null;

    public LoanDAO(){
         try
{
Class.forName("oracle.jdbc.driver.OracleDriver");
con = DriverManager.getConnection(

"jdbc:oracle:thin:@10.1.50.198:1535:nsbt19c", "sh",
"sh");
            stmt = con.createStatement();
        }
         catch (Exception e)
{
            e.printStackTrace();
        }

    }
    public void conn() {
         try
{
Class.forName("oracle.jdbc.driver.OracleDriver");

            con = DriverManager.getConnection(

"jdbc:oracle:thin:@10.1.50.198:1535:nsbt19c", "sh",
"sh");
            stmt = con.createStatement();
            }
```

```
        catch (Exception e) {
            e.printStackTrace();
        }
```

```
    }

    public int save(LoanAgreement e) {
try {
            Date d = new SimpleDateFormat("yyyy-MM-
dd").parse(e.getLoanDisbursalDate());
            String sd = new
SimpleDateFormat("dd/MM/yyyy").format(d);
            String s = "insert into LoanAgreement_16899
values(" + "'"
                    + e.getAgreementId() + "','" +
e.getLesseeId() + "',"
                    + e.getTenure() + "," + e.getRoi() +
"," + e.getLoanAmount() + ",'"
                    + e.getRepaymentFrequency() + "','"
+sd + "','" + e.getStatus() + "','"
                    + e.getProductCode() + "')";

            return stmt.executeUpdate(s);


        }

        catch (SQLException | ParseException e1) {
e1.printStackTrace();
        }

        return 0;
    }

    public String select(String id) {

        try {

            ArrayList<Object> list = new ArrayList<>();
            PreparedStatement pstmt =
con.prepareStatement("select * from LoanAgreement_16899
where lessee_id = ? ");

            pstmt.setString(1,id);
```

```
ResultSet rs = pstmt.executeQuery();
```

```
            while (rs.next()) {

list.add(rs.getString(1));
list.add(rs.getString(2));
list.add(rs.getInt(3));
list.add(rs.getDouble(4));
list.add(rs.getDouble(5));
list.add(rs.getString(6));
list.add(rs.getDate(7));
list.add(rs.getString(8));
list.add(rs.getString(9));
list.add(" \n~~~\n ");
            }

            if(list.isEmpty()){
                return "Data do not exist";
            }

            return list.toString();

        }

        catch (SQLException e1) {
            return "Database Connection Error";
        }

    }

    public String search(String id){

        try {

            ArrayList<Object> list = new ArrayList<>();
            PreparedStatement pstmt =
con.prepareStatement("select Customer_Id ,FirstName ,
LastName  from Customer_16899 where Customer_id =
(Select lessee_id from LoanAgreement_16899 where
Agreement_ID = ?)");
            pstmt.setString(1,id);
```

```
                ResultSet rs = pstmt.executeQuery();

                while (rs.next()) {

list.add(rs.getString(1));
list.add(rs.getString(2));
list.add(rs.getString(3));
                }

                if(list.isEmpty()){
                    return "Data do not exist";
                }

                return list.toString();

            }
             catch (SQLException e1)
{
e1.printStackTrace();
            }

            return "";
        }



}
```

Register Customer.jsp

```
<%@ taglib prefix="form"
uri="http://www.springframework.org/tags/form" %>
<html>
<head>
<title>Customer Registration Form</title>

<style>
.error {
color: red;
}
</style>
</head>
```

```
<body>
<h1> Customer Registration Form</h1>
<i>Fill out the form. Asterisk (*) means required.</i>
<br><br>

<form:form action="processCustomerForm"
modelAttribute="customer">

First name(*): <form:input path="firstName" />
<form:errors path="firstName" cssClass="error" />
<br><br>

Last Name: <form:input path="lastName" />
<form:errors path="lastName" cssClass="error" /> <br><br>
Gender(*) : Male: <form:radiobutton path="gender"
value="M" />
Female: <form:radiobutton path="gender" value="F" />
<form:errors path="gender" cssClass="error" /> <br><br>
Date of Birth : <form:input path="dateOfBirth"
type="date"/><br><br>

Contact Number(*): <form:input path="contactNumber" />
<form:errors path="contactNumber" cssClass="error" />
<br><br>

Email Address(*): <form:input path="emailAddress" />
<form:errors path="emailAddress" cssClass="error" />
<br><br>

Monthly Income(*): <form:input path="monthlyIncome" />
<form:errors path="monthlyIncome" cssClass="error" />
<br><br>

Profession: <form:input path="profession" />
<form:errors path="profession" cssClass="error" />
<br><br>

Total Monthly Expense(*): <form:input
path="totalMonthlyExpense" />
<form:errors path="totalMonthlyExpense" cssClass="error"
/> <br><br>

Designation: <form:input path="designation" />
<form:errors path="designation" cssClass="error" />
<br><br>
```

```
Company Name: <form:input path="companyName" />
<form:errors path="companyName" cssClass="error" />
<br><br>

<input type="submit" value="Register" />
</form:form>

</body>
</html>
```

ApplyLoan.jsp

```
<%@ taglib prefix="form"
uri="http://www.springframework.org/tags/form" %>
<html>
<head>
<title>Customer Registration Form</title>

<style>
.error {
color: red;
}
</style>
</head>

<body>
<h1> Apply For Loan</h1>
<i>Fill out the form. Asterisk (*) means required.</i>
<br><br>

<form:form action="processLoanForm"
modelAttribute="loan">

Agreement ID(*): <form:input path="agreementId" />
<form:errors path="agreementId" cssClass="error" />
<br><br>

Customer Id(*) : <form:input path="lesseeId" />
<form:errors path="lesseeId" cssClass="error" /> <br><br>
Tenure : <form:input path="tenure" />
```

```
<form:errors path="tenure" cssClass="error" /> <br><br>

Rate : <form:input path="roi" />
<form:errors path="roi" cssClass="error" /> <br><br>

Loan Amount : <form:input path="loanAmount" />
<form:errors path="loanAmount" cssClass="error" />
<br><br>

Repayment Frequency(*) : <form:input path="repaymentFrequency" />
<form:errors path="repaymentFrequency" cssClass="error"
/> <br><br>

loanDisbursalDate : <form:input path="loanDisbursalDate"
type="date"/><br><br>

Status(*) : <form:input path="status" />
<form:errors path="status" cssClass="error" /> <br><br>
Product Code(*) : <form:input path="productCode" />
<form:errors path="productCode" cssClass="error" />
<br><br>

<input type="submit" value="Apply" />
</form:form>

</body>
</html>
```

Search.jsp

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>
        Search
    </title>
</head>

<body>
```

```html

<h1>Search by Customer Code </h1>
<br><br>
<div>
<form action="searchByCustomerCode" >

    <label for="lesseeId">Customer Code </label><br>
    <input type="text" id ="lesseeId" name="lesseeId"
><br><br>

    <input type="submit"  value="Search">

</form>
</div>



<br>
<hr>
<br>



<h1>Search by Loan ID </h1>
<br><br>
<div>

<form action="searchByLoanId" >

    <label for="agreementId">Loan Id</label><br>
    <input type="text" id ="agreementId"
name="agreementId" ><br><br>

    <input type="submit"  value="Search">

</form>
</div>
</body>

</html>
```

Display.jsp:

```
<html>
<body>
<h2> Customer has successfully registered </h2>
</body>
</html>
```

Display1.jsp

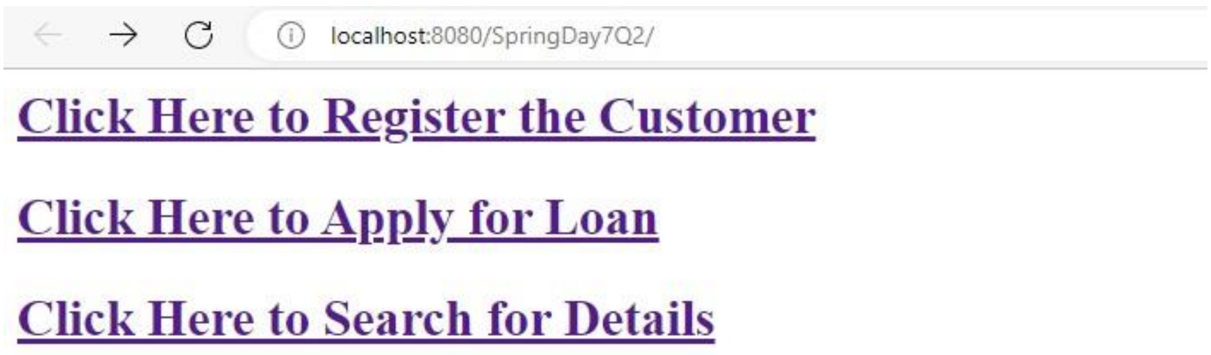
```
<html>
<body>
<h2> Applied for loan successfully!!! </h2>
</body>
</html>
```

Display2.jsp

```
<%@ page isELIgnored = "false" %>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="ISO-8859-1">
        <title>User Data</title>
    </head>
    <body>

       <h3>Loan Details as per Customer Code :</h3>

       ${loan}

    </body>

</html>
```

Display3.jsp

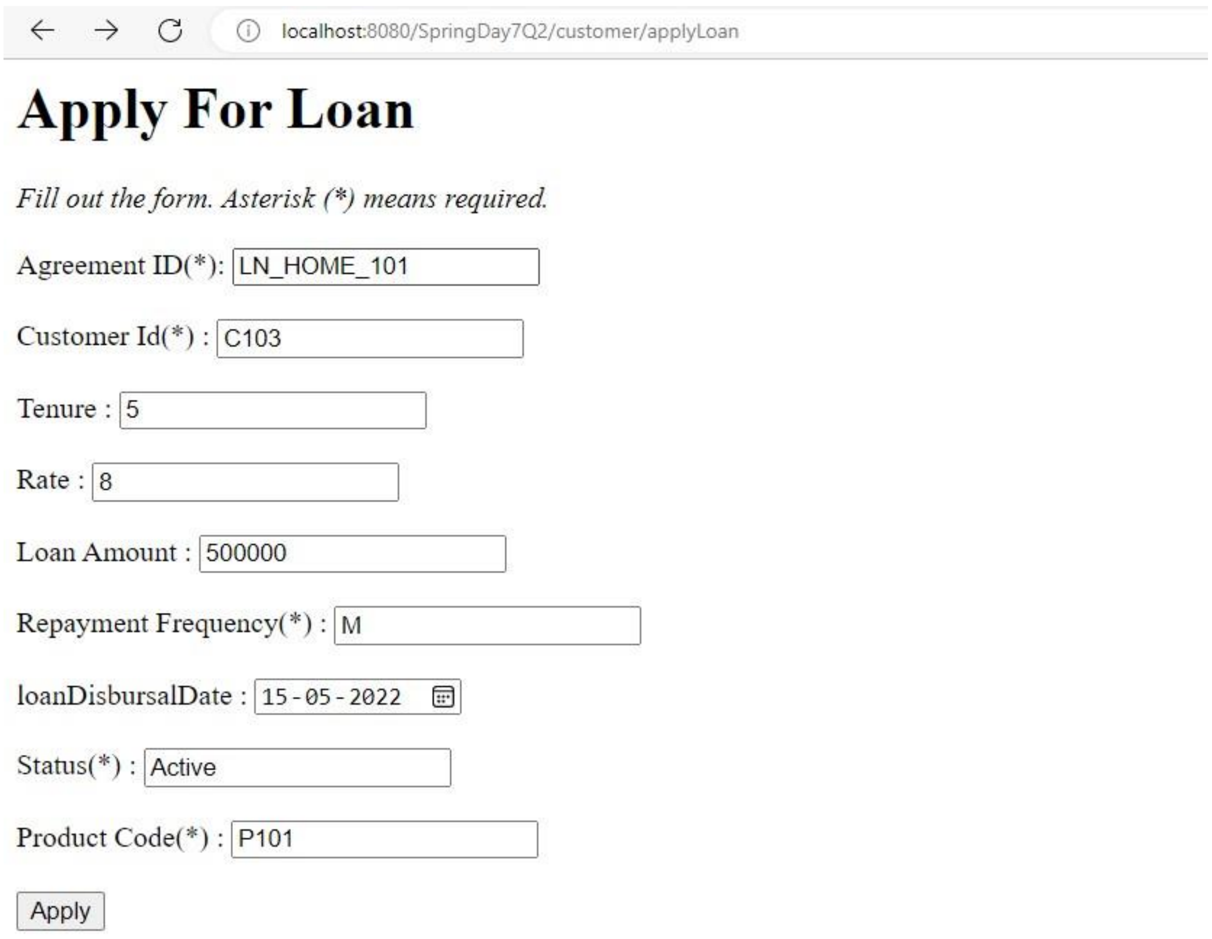
```
<%@ page isELIgnored = "false" %>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="ISO-8859-1">
        <title>User Data</title>
    </head>
    <body>

       <h3>Customer Details as per Loan ID :</h3>
       <table>

            <tr>
                <td>${customerData}</td>
            </tr>

       </table>
    </body>

</html>
```


localhost:8080/SpringDay7Q2/

Click Here to Register the Customer

Click Here to Apply for Loan

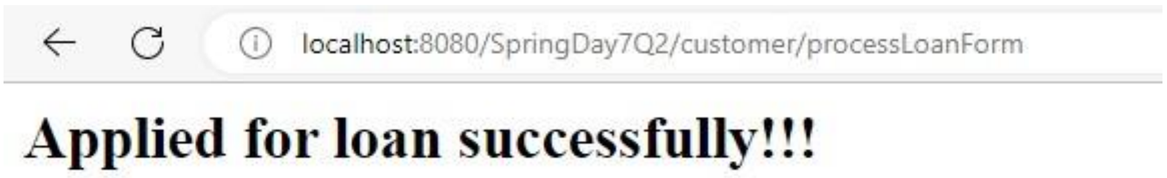
Click Here to Search for Details


localhost:8080/SpringDay7Q2/customer/applyLoan

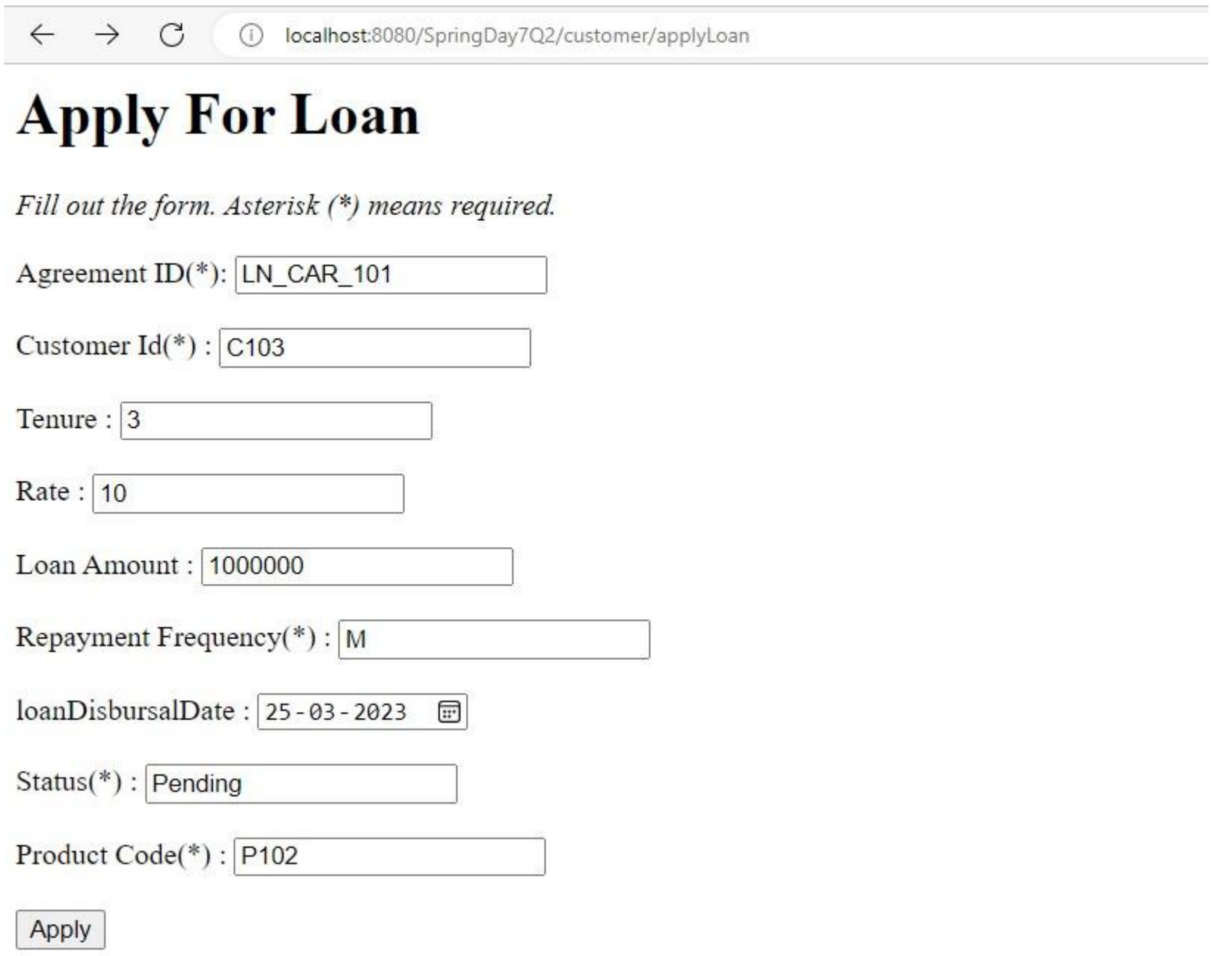
Apply For Loan

Fill out the form. Asterisk (*) means required.

Agreement ID(*):	LN_HOME_101
Customer Id(*) :	C103
Tenure :	5
Rate :	8
Loan Amount :	500000
Repayment Frequency(*) :	M
loanDisbursalDate :	15-05-2022
Status(*) :	Active
Product Code(*) :	P101


localhost:8080/SpringDay7Q2/customer/processLoanForm

Applied for loan successfully!!!

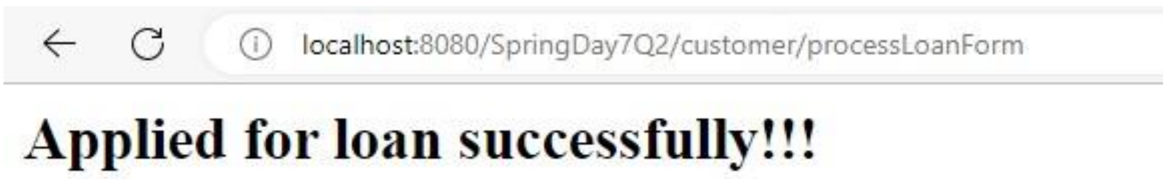


localhost:8080/SpringDay7Q2/customer/applyLoan

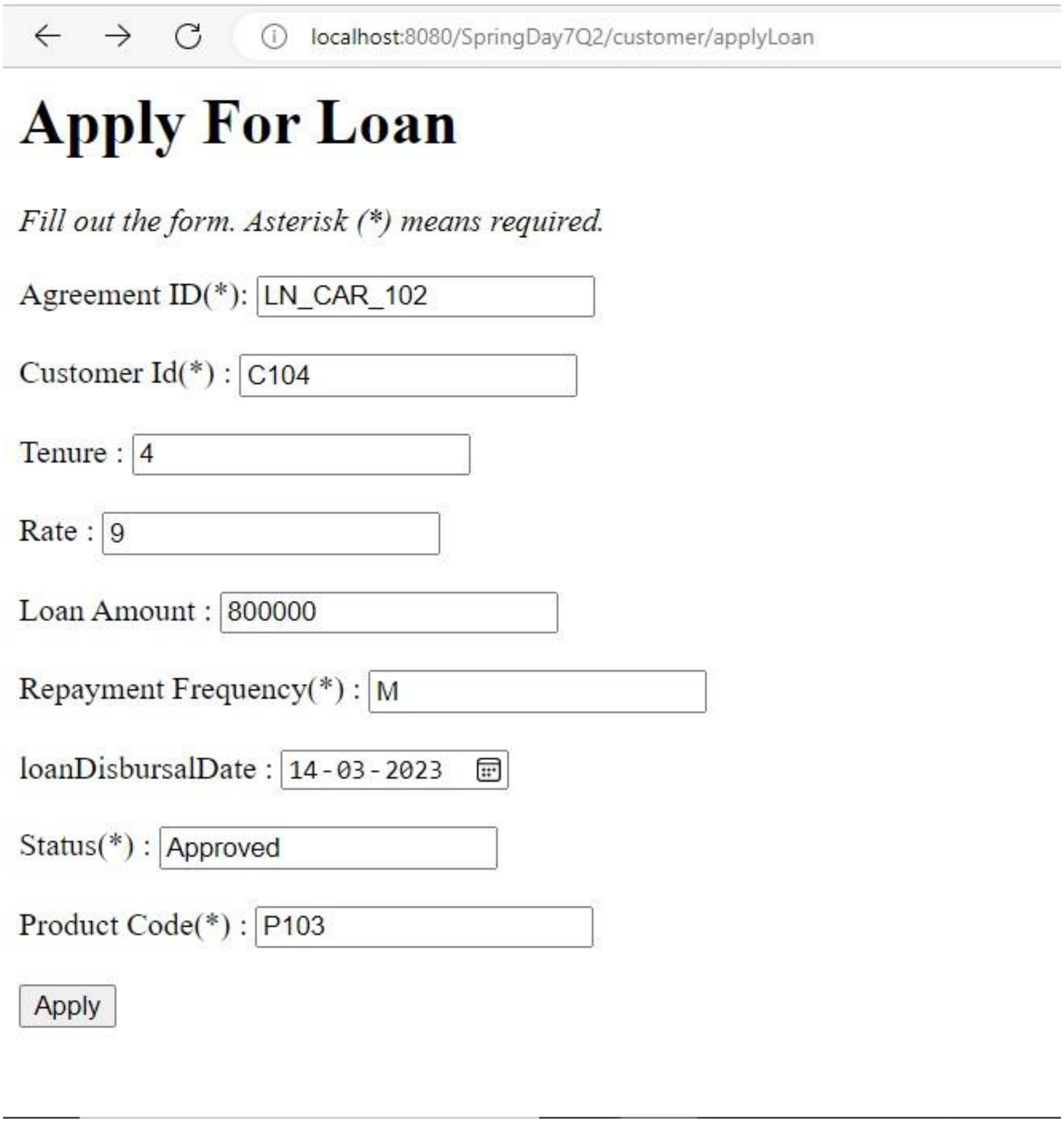
Apply For Loan

Fill out the form. Asterisk (*) means required.

Agreement ID(*):	LN_CAR_101
Customer Id(*) :	C103
Tenure :	3
Rate :	10
Loan Amount :	1000000
Repayment Frequency(*) :	M
loanDisbursalDate :	25-03-2023
Status(*) :	Pending
Product Code(*) :	P102


localhost:8080/SpringDay7Q2/customer/processLoanForm

Applied for loan successfully!!!


localhost:8080/SpringDay7Q2/customer/applyLoan

Apply For Loan

Fill out the form. Asterisk (*) means required.

Agreement ID(*): LN_CAR_102

Customer Id(*) : C104

Tenure : 4

Rate : 9

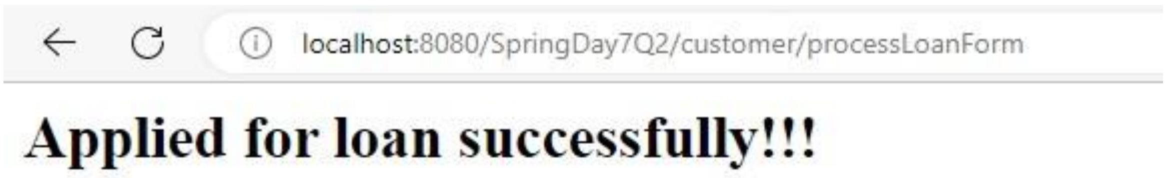
Loan Amount : 800000

Repayment Frequency(*) : M

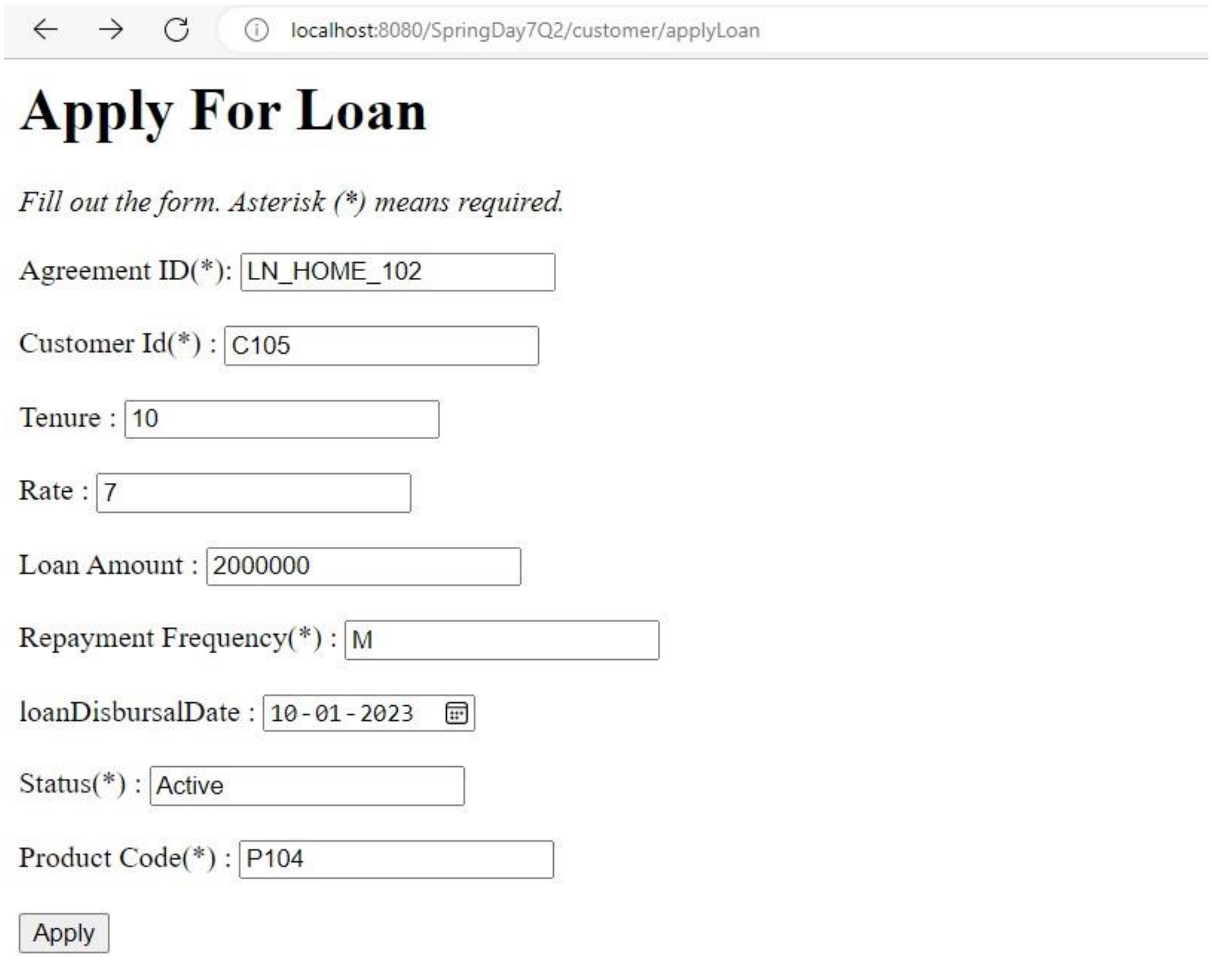
loanDisbursalDate : 14-03-2023

Status(*) : Approved

Product Code(*) : P103


localhost:8080/SpringDay7Q2/customer/processLoanForm

Applied for loan successfully!!!

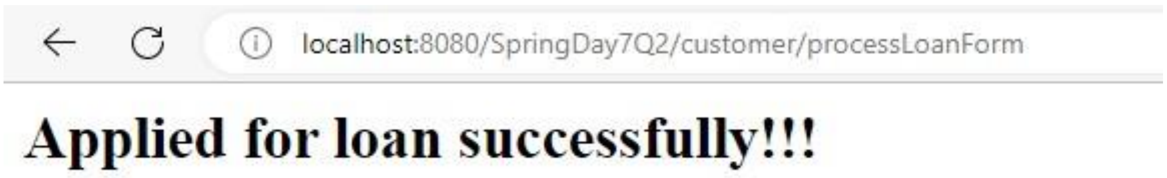

localhost:8080/SpringDay7Q2/customer/applyLoan

Apply For Loan

Fill out the form. Asterisk (*) means required.

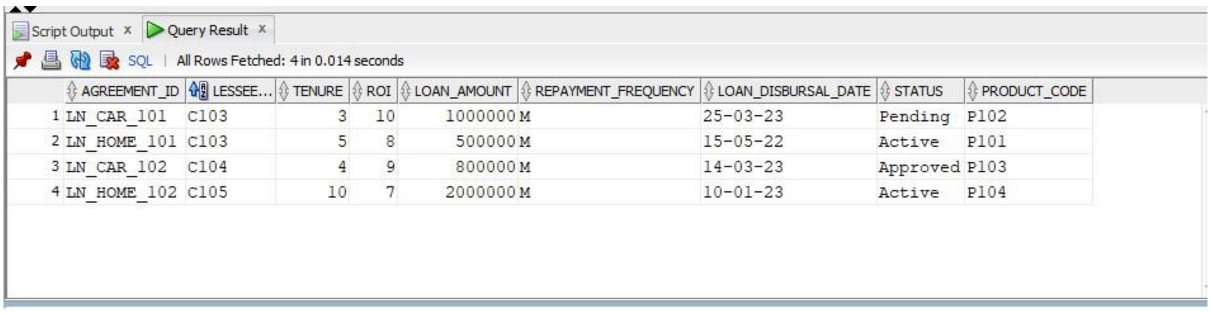
Agreement ID(*):	LN_HOME_102
Customer Id(*) :	C105
Tenure :	10
Rate :	7
Loan Amount :	2000000
Repayment Frequency(*) :	M
loanDisbursalDate :	10-01-2023
Status(*) :	Active
Product Code(*) :	P104

Apply


localhost:8080/SpringDay7Q2/customer/processLoanForm

Applied for loan successfully!!!

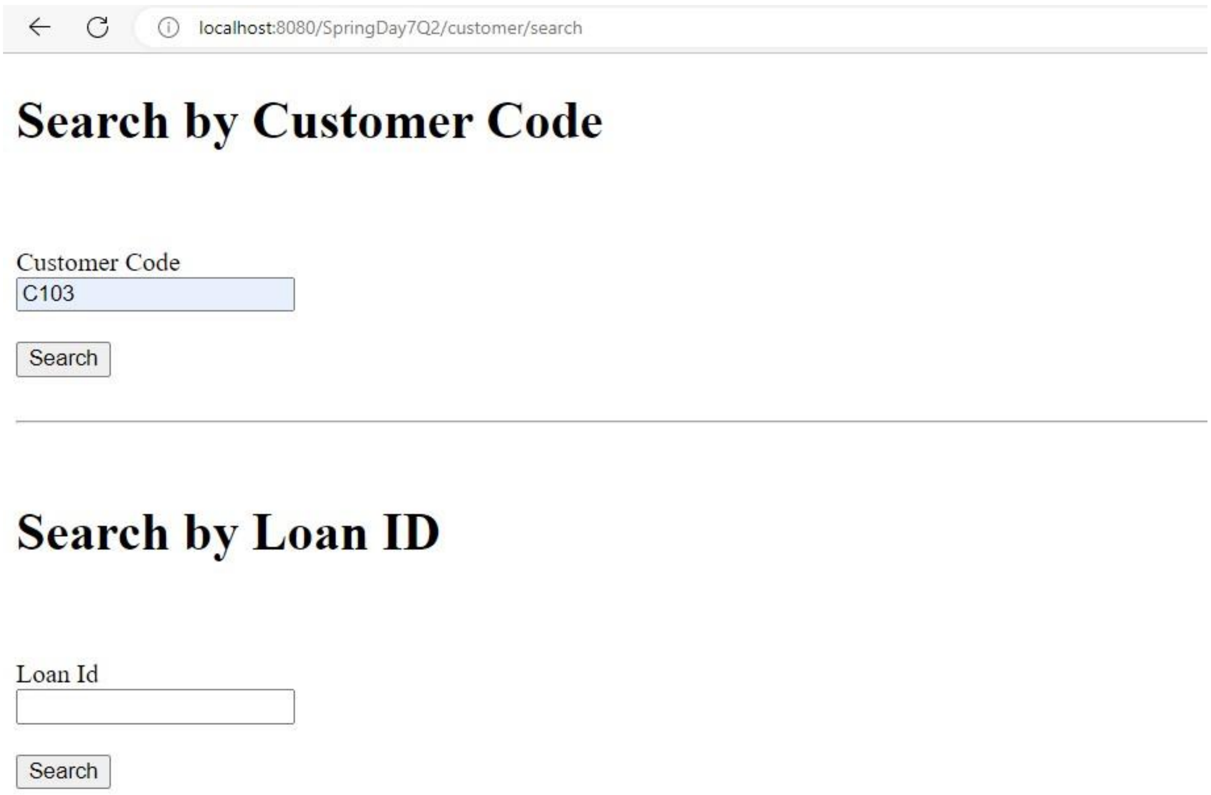
From Database


Script Output × Query Result ×

SQL | All Rows Fetched: 4 in 0.014 seconds

	AGREEMENT_ID	LESSEE...	TENURE	ROI	LOAN_AMOUNT	REPAYMENT_FREQUENCY	LOAN_DISBURSAL_DATE	STATUS	PRODUCT_CODE
1	LN_CAR_101	C103	3	10	1000000	M	25-03-23	Pending	P102
2	LN_HOME_101	C103	5	8	500000	M	15-05-22	Active	P101
3	LN_CAR_102	C104	4	9	800000	M	14-03-23	Approved	P103
4	LN_HOME_102	C105	10	7	2000000	M	10-01-23	Active	P104

Search by Customer Code:


localhost:8080/SpringDay7Q2/customer/search

Search by Customer Code

Customer Code

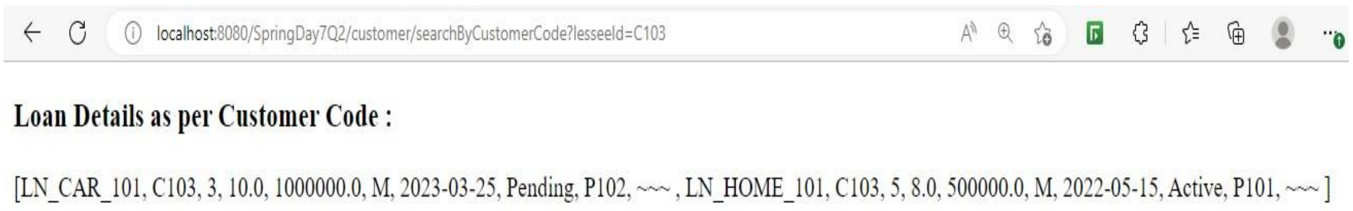
C103

Search

Search by Loan ID

Loan Id

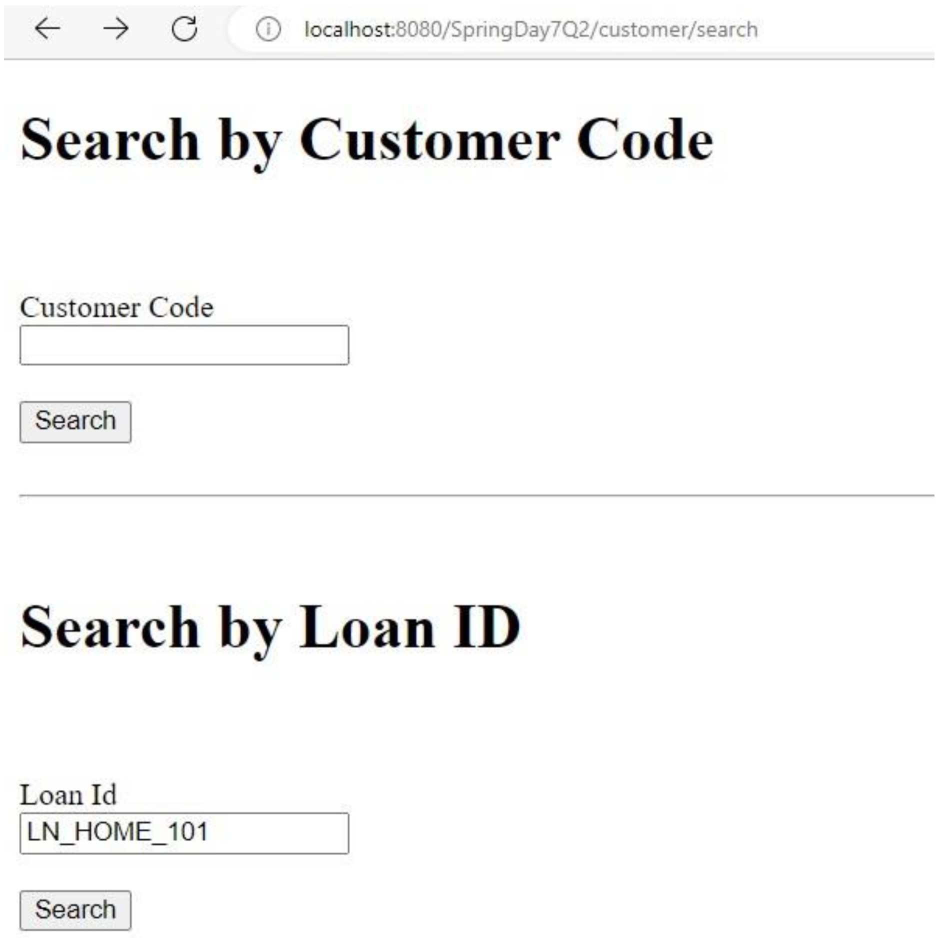
Search


localhost:8080/SpringDay7Q2/customer/searchByCustomerCode?lesseeId=C103

Loan Details as per Customer Code :

```
[LN_CAR_101, C103, 3, 10.0, 1000000.0, M, 2023-03-25, Pending, P102, ~~~ , LN_HOME_101, C103, 5, 8.0, 500000.0, M, 2022-05-15, Active, P101, ~~~ ]
```

Search by Loan ID:



localhost:8080/SpringDay7Q2/customer/search

Search by Customer Code

Customer Code

Search

Search by Loan ID

Loan Id

LN_HOME_101

Search