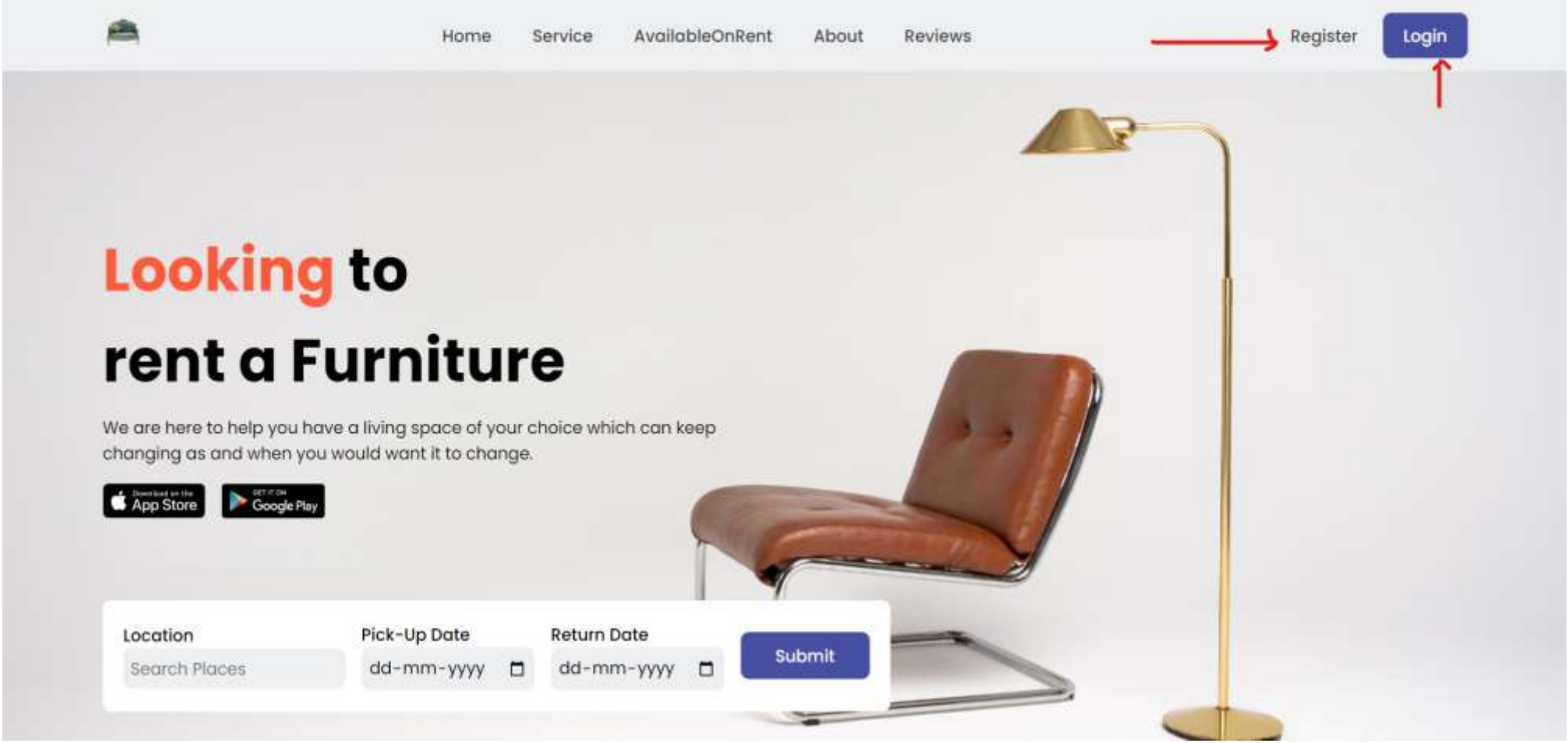
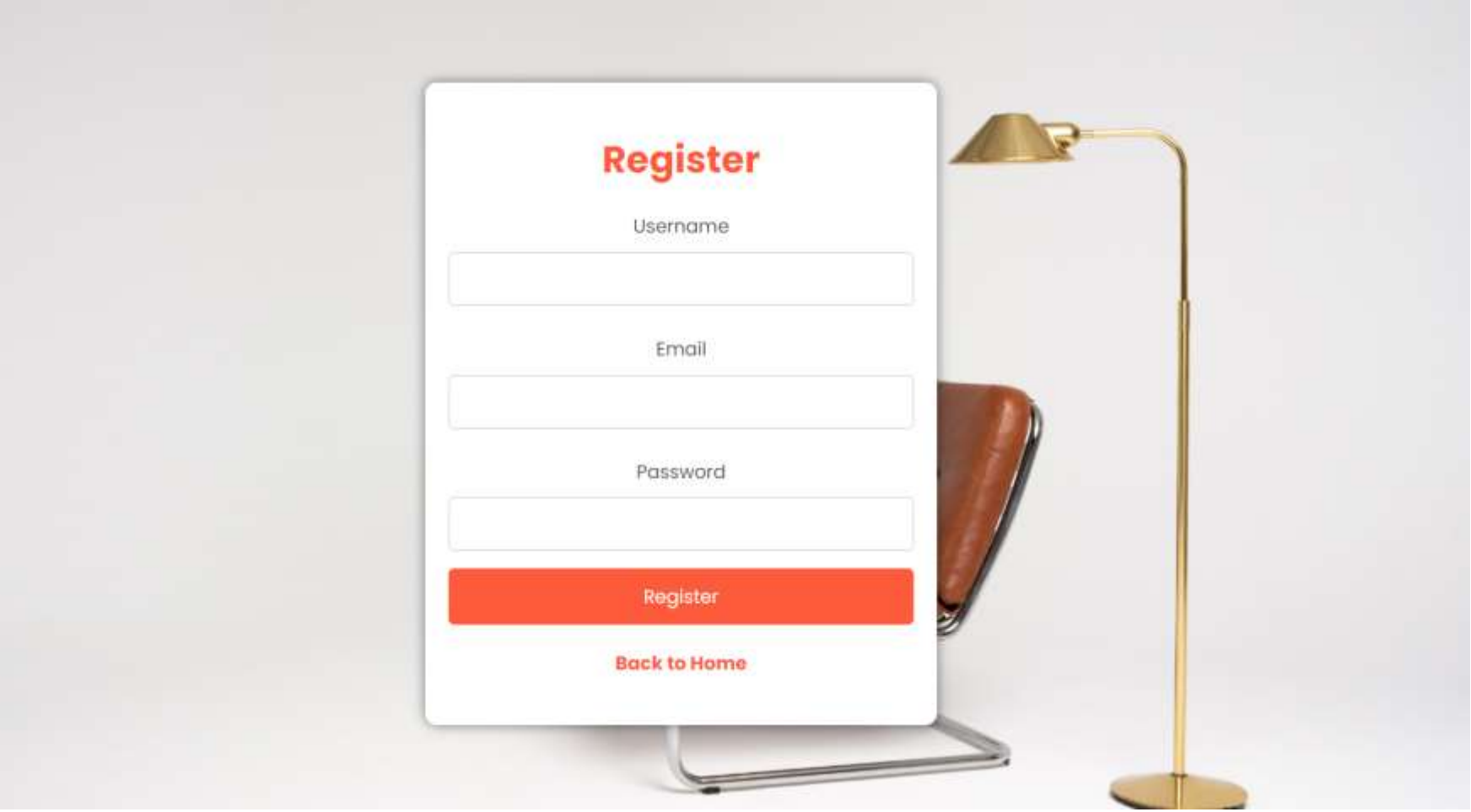
FULL STACK PROJECT

Step:1 Expected out put

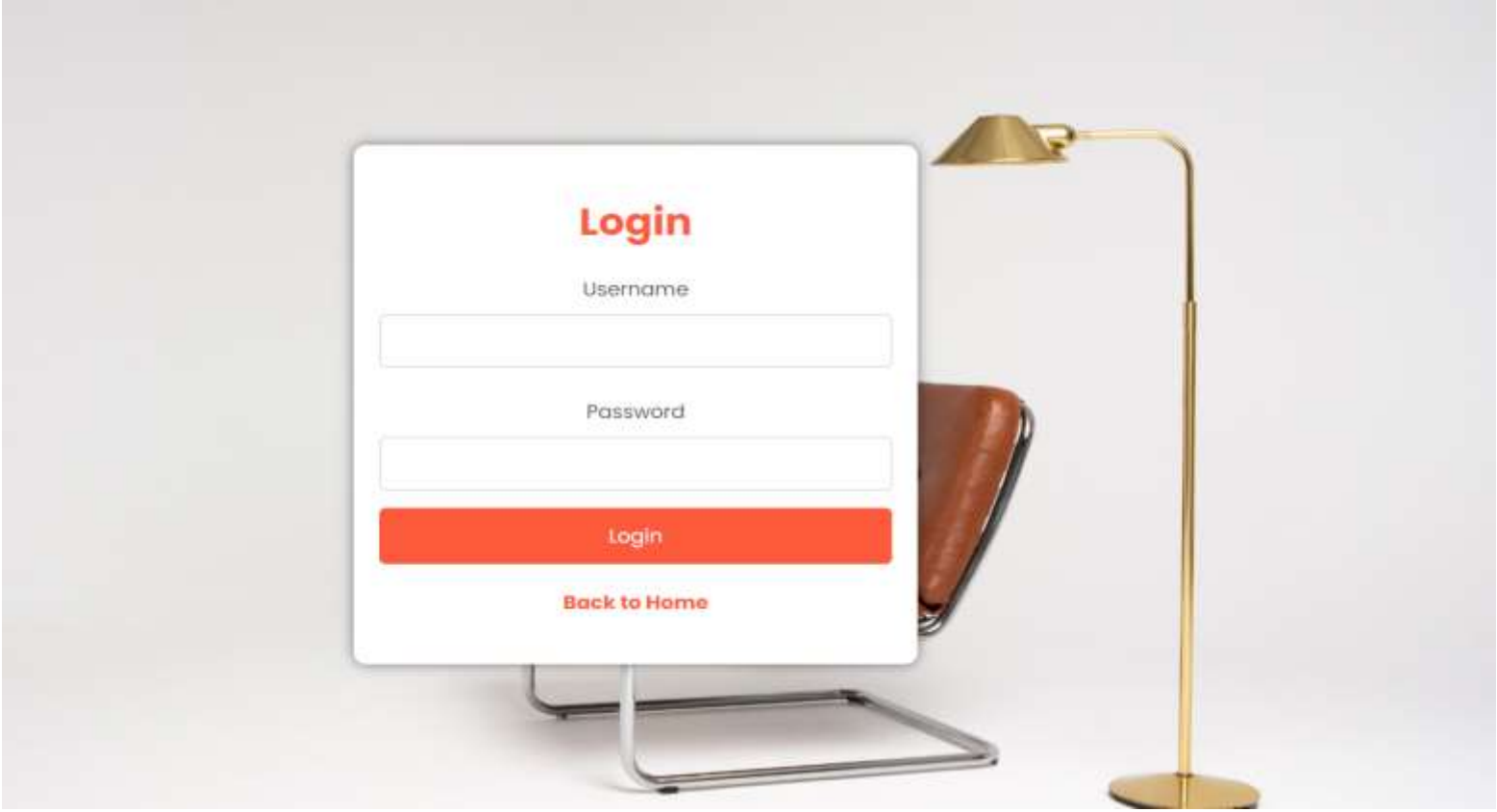
This is our web application with the help of two buttons named Register and Login.



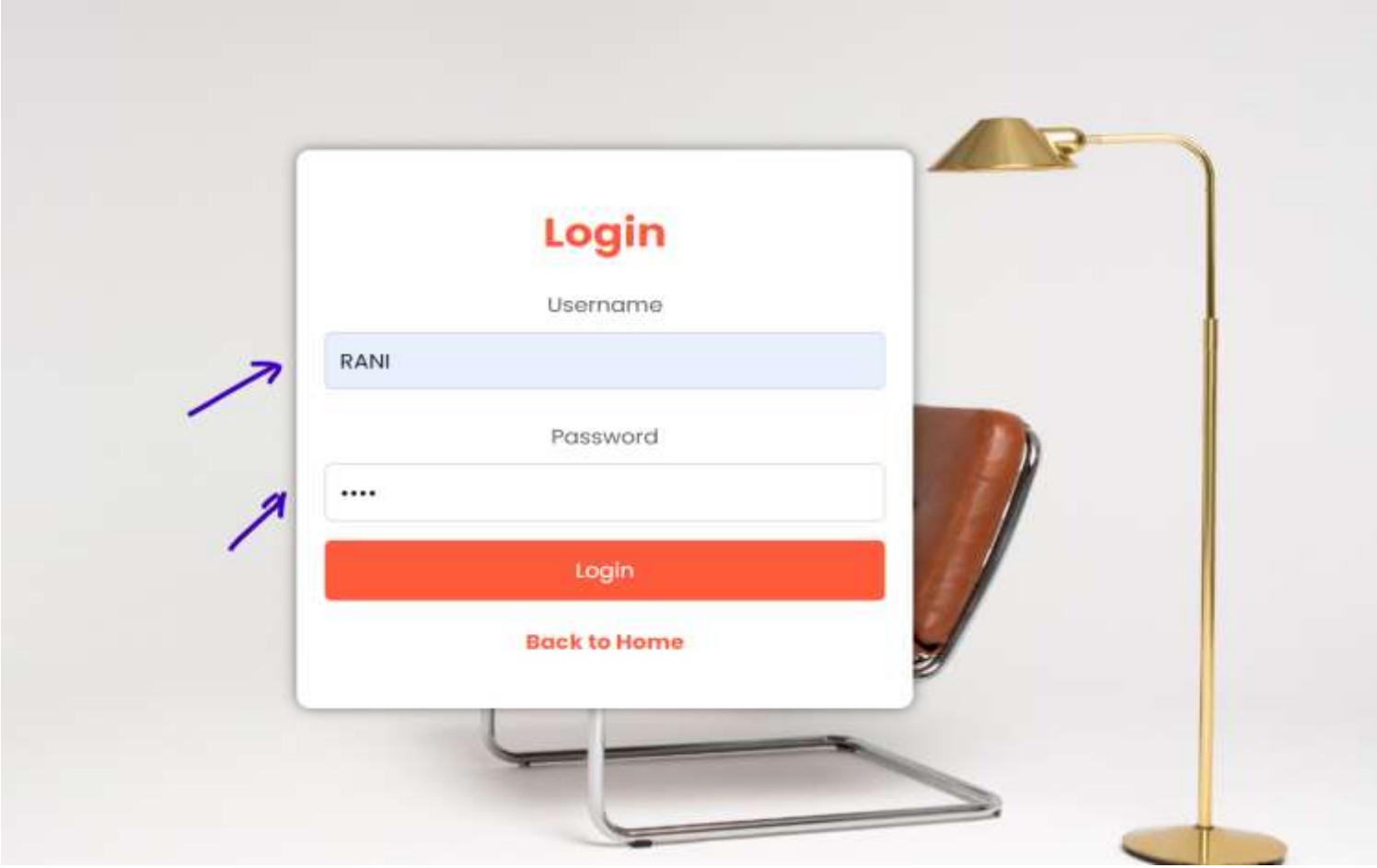
This is our register page by giving the name and email id and by creating the password we can register



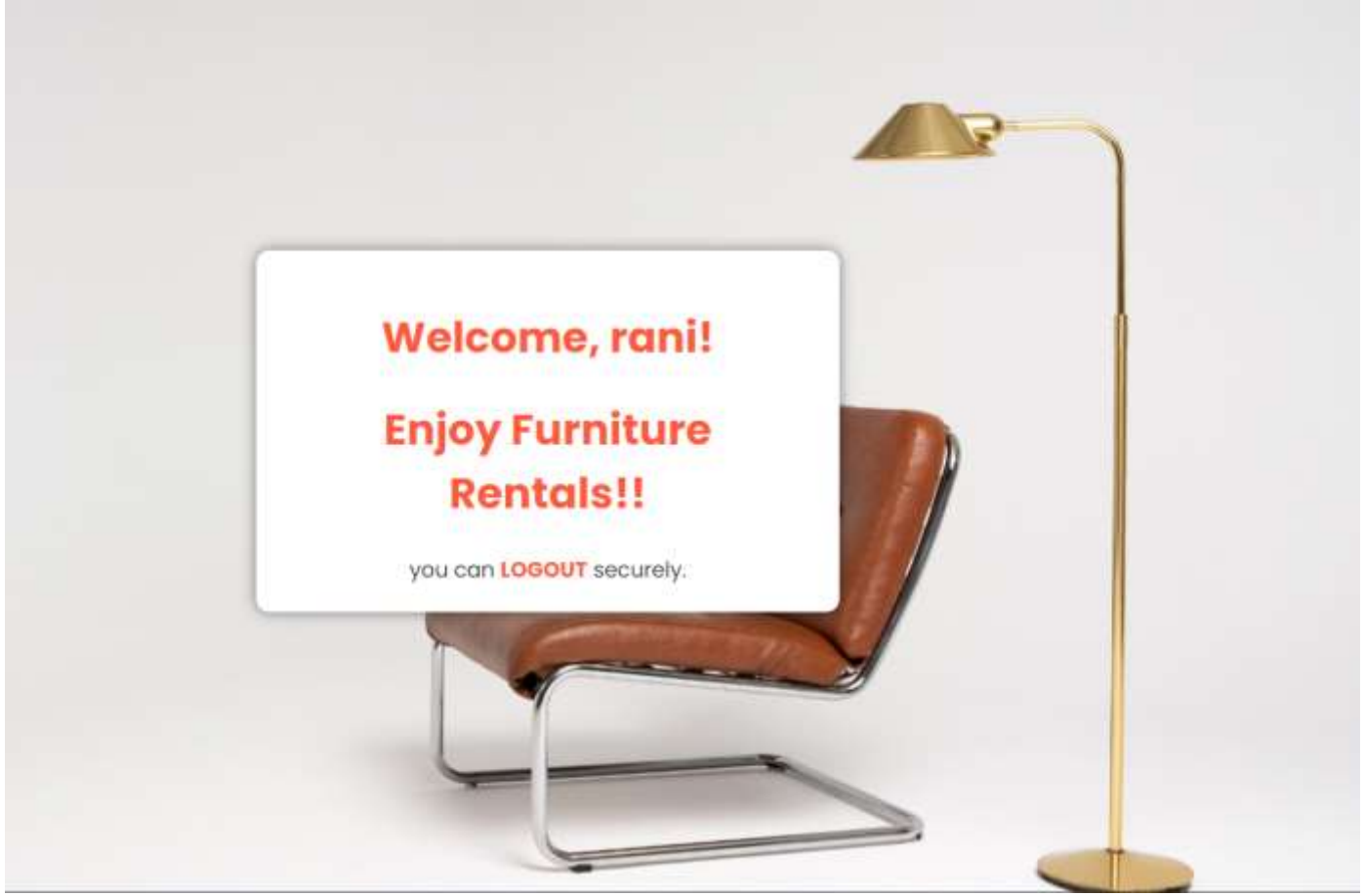
After registering by using the credentials we can login in the application.



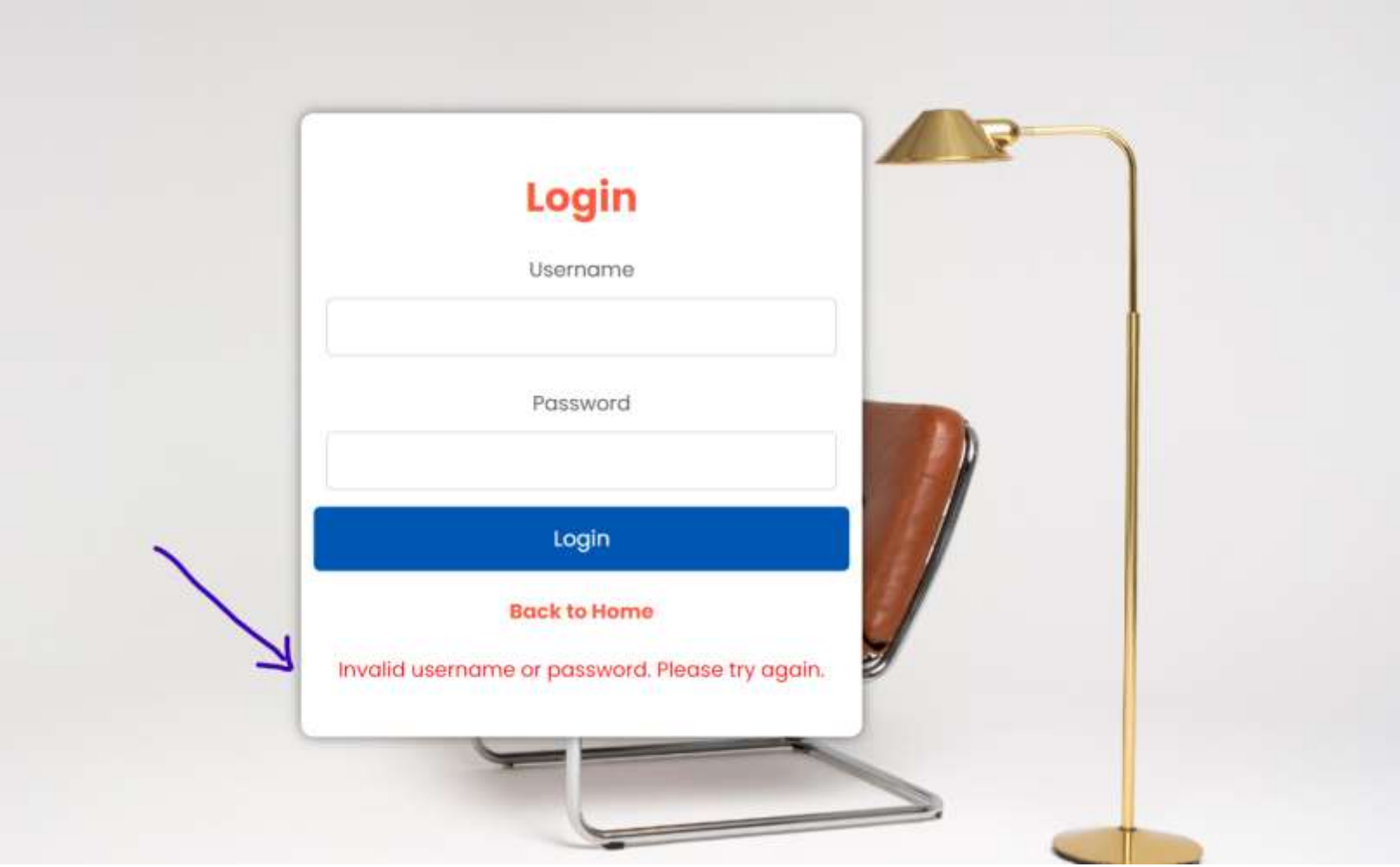
This is how we are giving the credentials



after login in to the page we are seeing a welcome message as shown below



If we are giving any wrong credentials we can see the error message.



Step 2: Required resources

- HTML
- CSS
- JSP
- SERVLETS
- JS
- JDBC
- MYSQL

HTML

- HTML files serve as the backbone of web pages, providing the structure and content that web browsers use to display information on the internet.

- In simple terms, HTML files define the layout, text, images, links, and other elements that make up a webpage, allowing users to interact with and consume content online.
- Index.html

CSS

- CSS files complement HTML by controlling the presentation and style of web pages. They define how HTML elements should appear on the screen, specifying attributes like colors, fonts, layout, and spacing.

- In simple terms, CSS files make websites visually appealing and user-friendly by applying design rules and enhancing the overall aesthetic of the content presented in HTML files
- Style.css and oldstyle.css

JSP

- JSP (JavaServer Pages) files are used in web development to dynamically generate web pages. They combine HTML with Java code, allowing developers to create dynamic content that can interact with databases, handle user input, and perform other server-side tasks.

- In simple terms, JSP files enable the creation of dynamic and interactive web applications by embedding Java code within HTML, making it possible to generate customized content based on user input or other factors.
- Login.jsp
- Register.jsp
- Welcome.jsp

SERVLETS

- Servlets are Java classes that extend the capabilities of web servers to generate dynamic web content. They handle requests from web clients (such as web browsers) and produce responses based on the request. Servlets are commonly used to process form data, interact with databases, and perform other server-side tasks in web applications.

- In simple terms, servlets serve as the backbone of dynamic web applications, enabling developers to create interactive and data-driven websites by processing requests and generating appropriate responses.
- RegisterServlet
- loginServlet

JS

- JavaScript (JS) files are used to add interactivity and dynamic behavior to web pages.

- contain code that can be executed by web browsers to manipulate HTML elements, respond to user actions, and communicate with servers.

- In simple terms, JS files enable developers to create interactive and engaging web experiences by adding features like animations, form validation, interactive maps, and more, enhancing the functionality and user experience of websites.
- Main.jsp

JDBC

- JDBC (Java Database Connectivity) is a Java API that allows programs written in Java to access and manipulate data stored in relational databases. It provides a standard interface for connecting Java applications to database management systems, executing SQL queries, and handling database transactions.

- In simple terms, JDBC enables Java applications to interact with databases, allowing them to store, retrieve, and manipulate data, which is crucial for building database-driven applications such as web applications, enterprise systems, and more.
- DBUtil.java

MYSQL

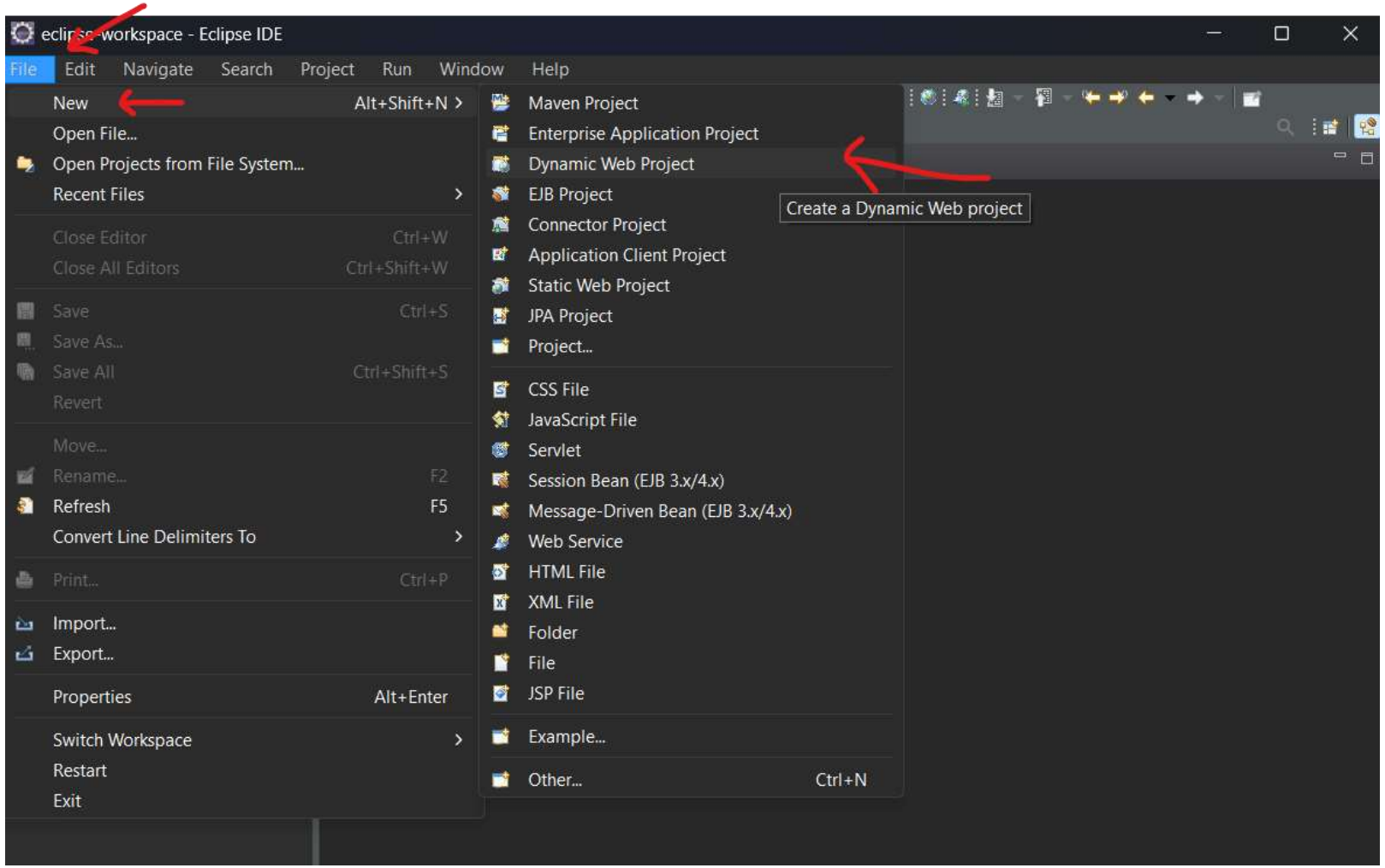
- A "MySQL file" typically refers to the executable file of the MySQL database management system. MySQL is a popular open-source relational database system used to store and manage data in various applications.

- In simple terms, MySQL provides a platform for creating databases, storing data in tables, and performing operations like querying, updating, and deleting data. It's commonly used in web development, business applications, and other software where structured data storage and retrieval are required.
- UserDaoImpl.java
- UserDao.java

FULL STACK PROJECT
VIDEO-2

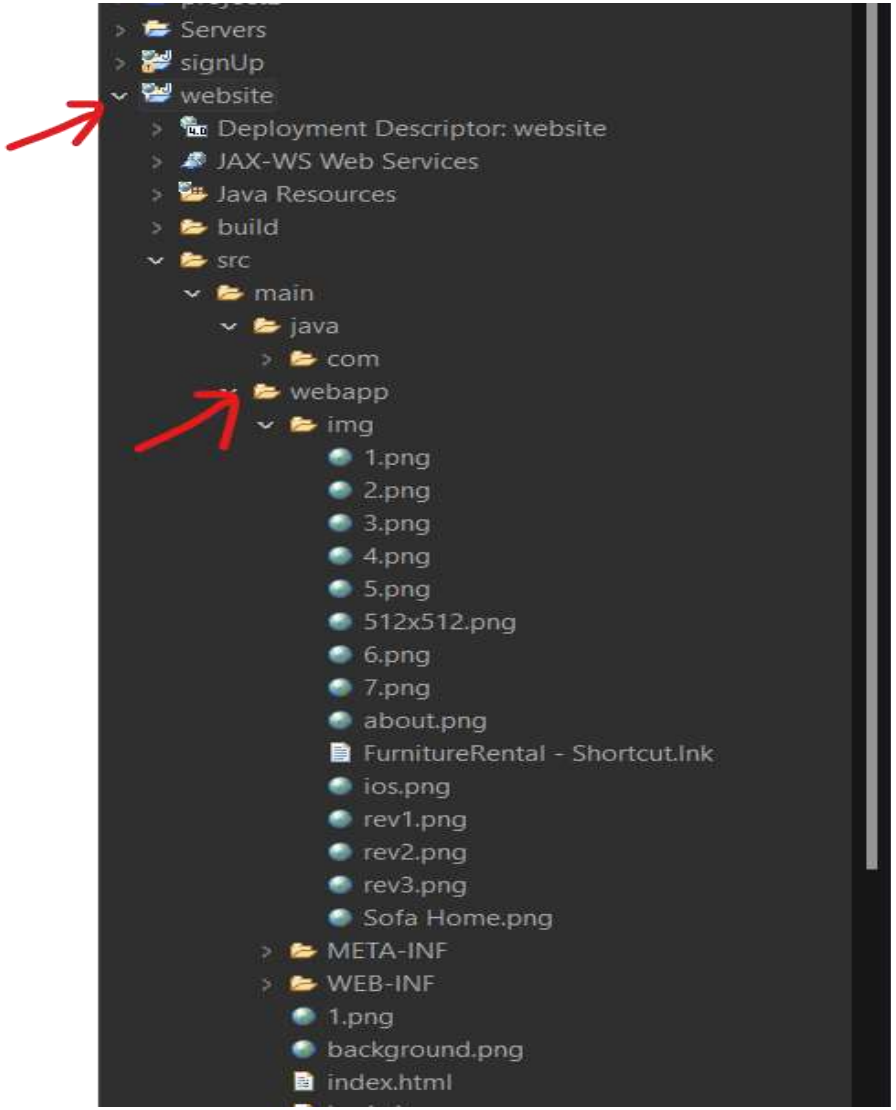
Step 3: Create a dynamic web project

Go to file→ new→dynamic web project→clickon next→ click on next→ generate web.xml file→ click on finish



Step4:Add a folder of furniture rental website in to the eclipse IDE

Copy the folder and paste it in the project WEBAPP folder.



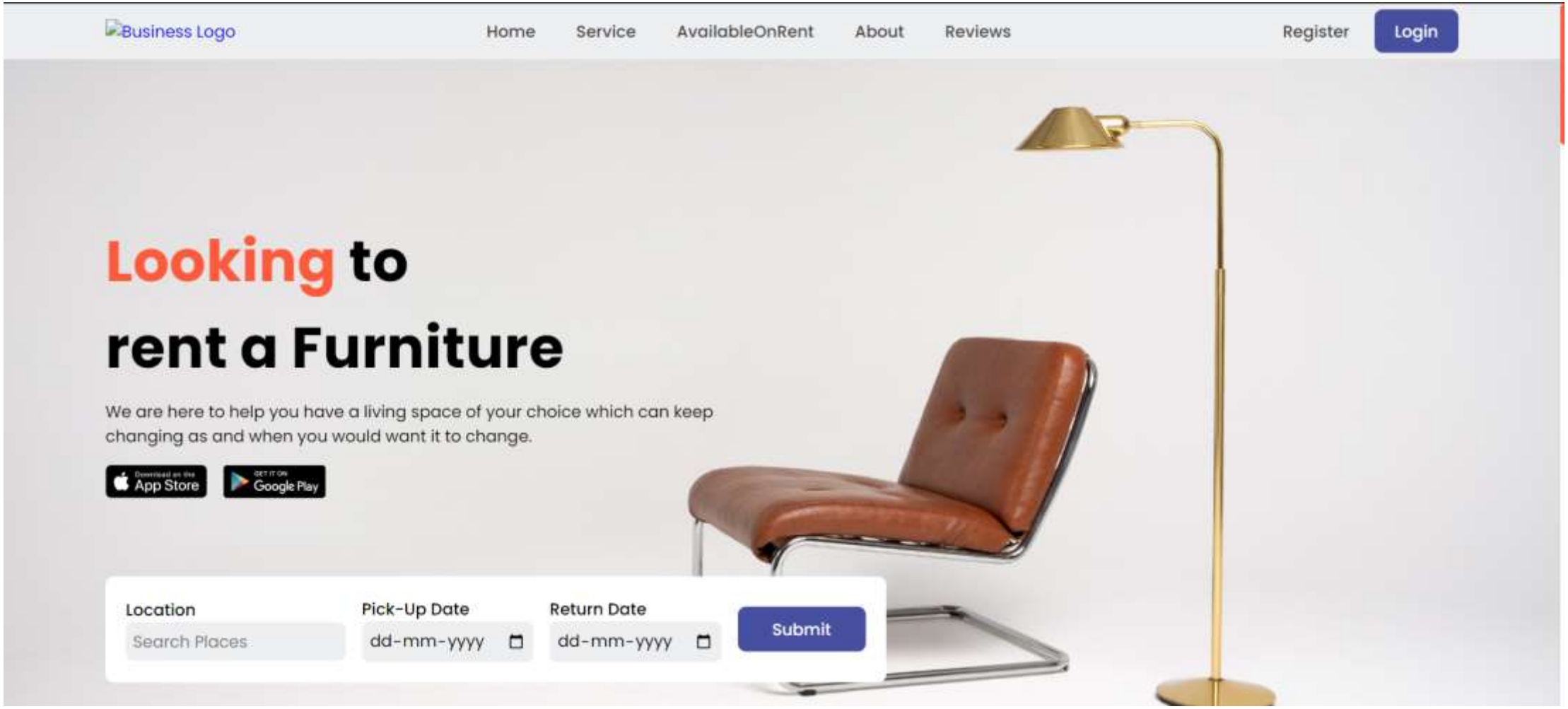
Create index.html file

- Link the register.jsp file and login.jsp file in index.html file.

index.html

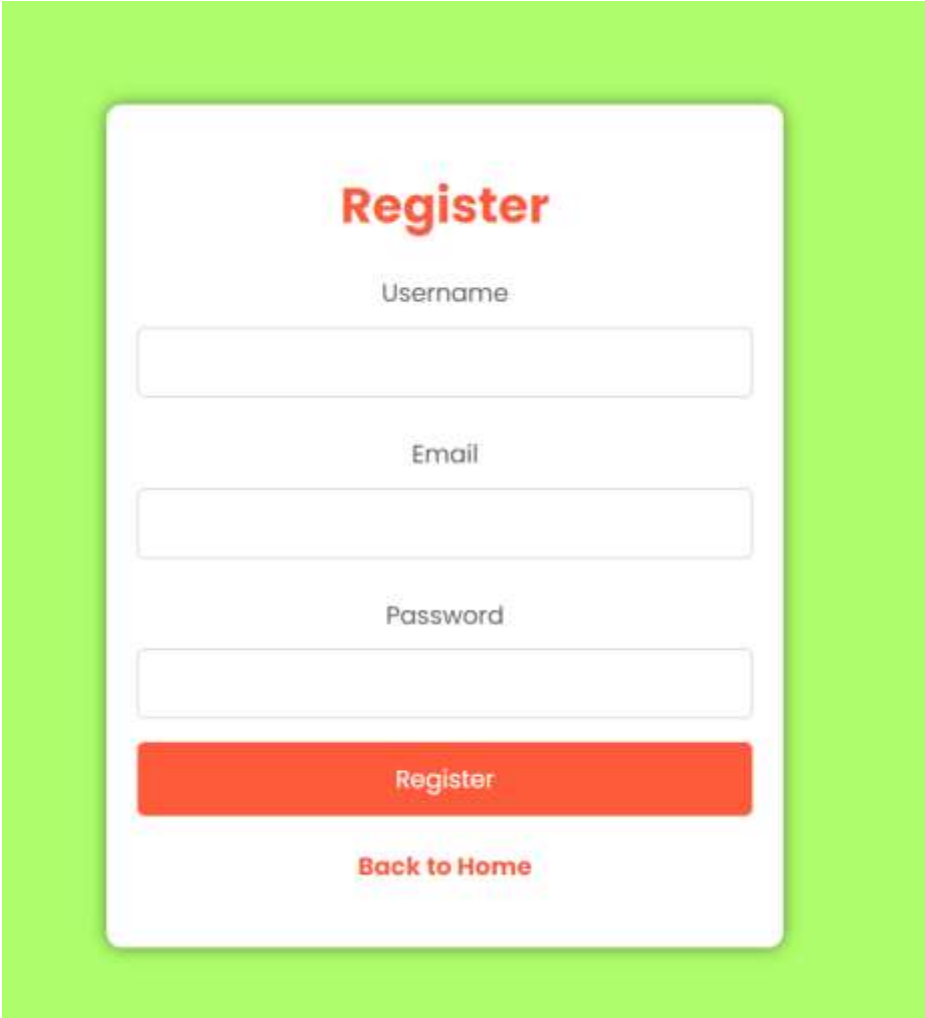
```html

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Furniture Rental</title>

    <link rel="stylesheet" href="style.css" />
    <link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>


</head>
<body>

    <header>
        <a href="#" class="logo"><img src="logo.png" alt="Business Logo"/></a>

        <div class="bx bx-menu" id="menu-icon"></div>

        <ul class="navbar">
            <li><a href="#home">Home</a></li>
            <li><a href="#service">Service</a></li>
            <li><a href="#availableOnRent">AvailableOnRent</a></li>
            <li><a href="#about">About</a></li>
            <li><a href="#reviews">Reviews</a></li>
        </ul>

        <div class="header-btn">
            <a href="register.jsp" class="sign-up">Register</a>
            <a href="login.jsp" class="sign-in">Login</a>
        </div>
    </header>

```

This is how we can see the home page



FULL STACK PROJECT
VIDEO-3

This is our expected Register page

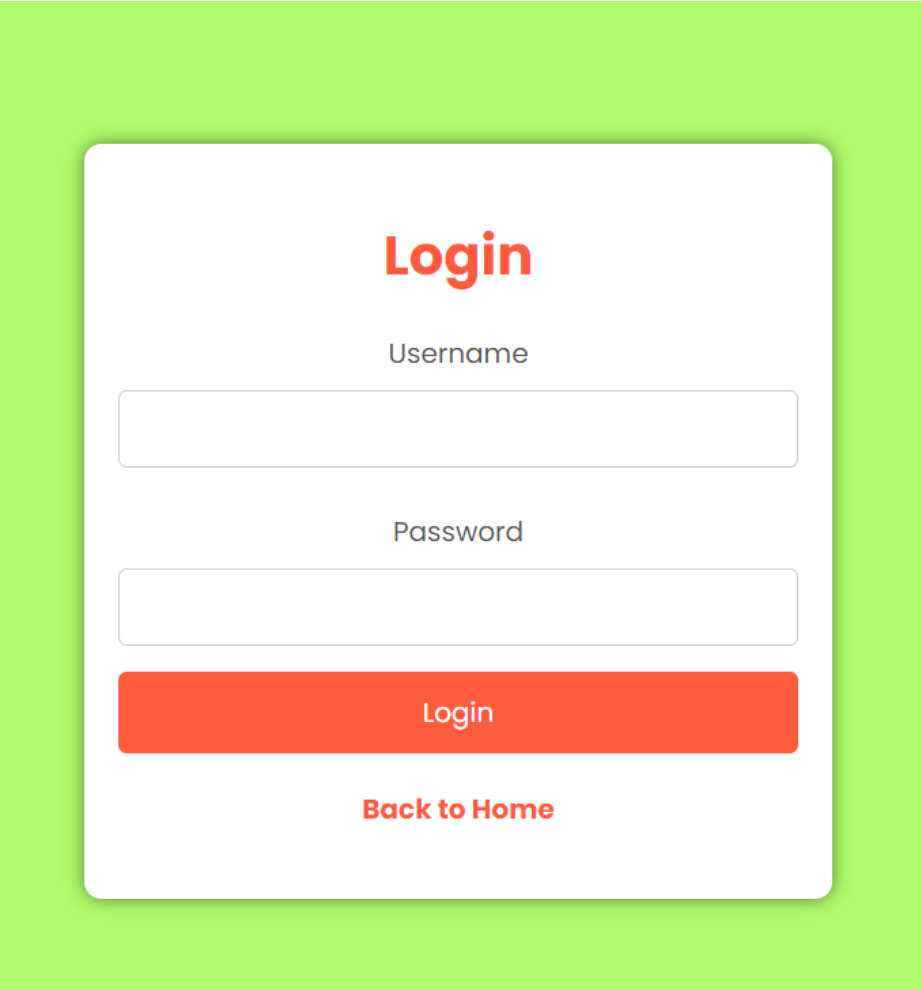


Create register.jsp file

register.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Register</title>
<link rel="stylesheet" type="text/css" href="Oldstyle.css">
</head>
<body>

    <div class="container">
        <h1>Register</h1>
        <form action="RegisterServlet" method="post">
            <label for="username">Username</label>
            <input type="text" id="username" name="username" required><br>
            <label for="email">Email</label>
            <input type="text" id="username" name="email" required><br>
            <label for="password">Password</label>
            <input type="password" id="password" name="password" required><br>
            <button type="submit">Register</button>
        </form>

        <p><a href="index.html">Back to Home</a></p>
        </div>
</body>
</html>
```

This is our expected Login page

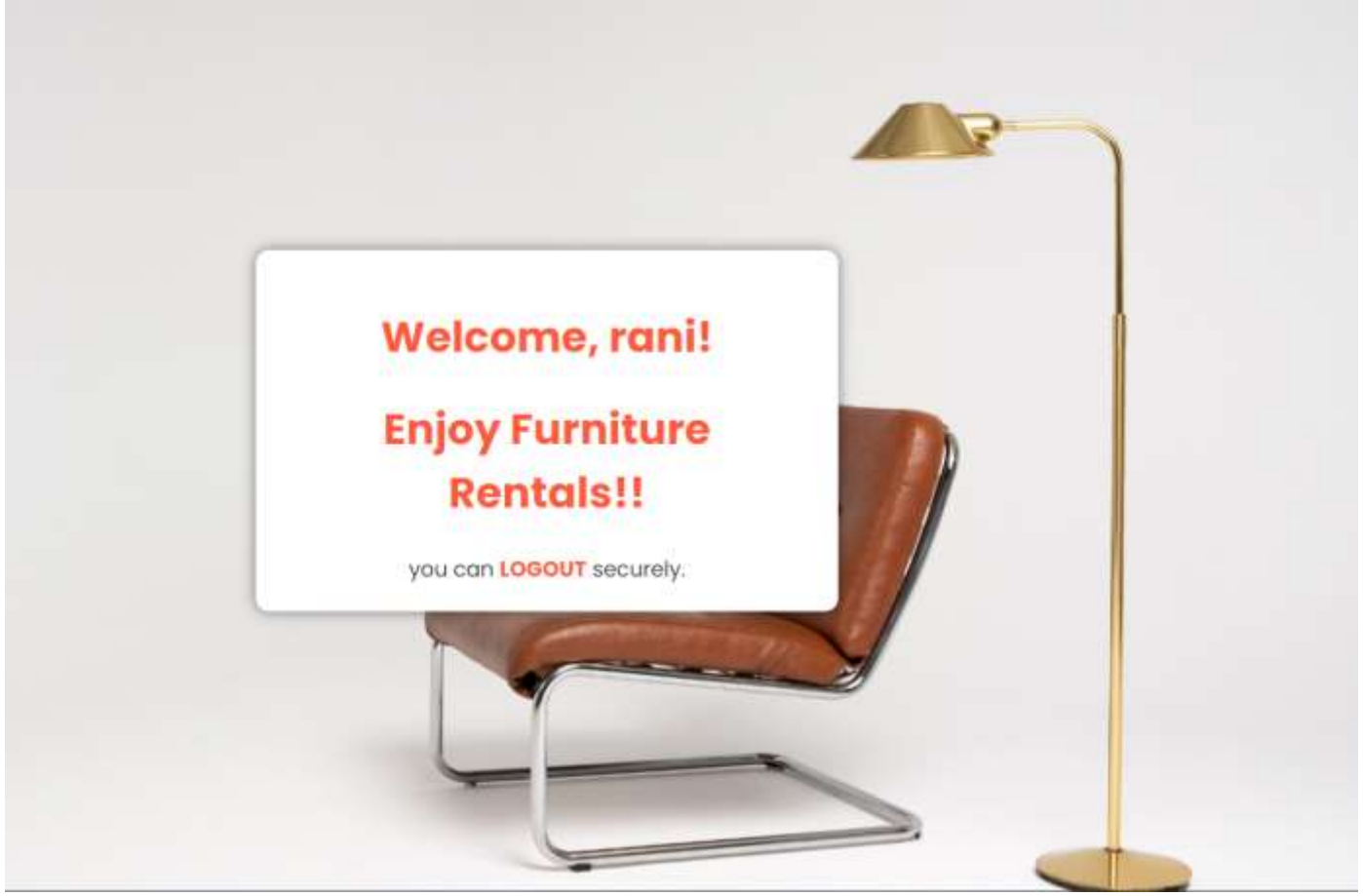


Create login.jsp

login.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Login</title>
<link rel="stylesheet" type="text/css" href="Oldstyle.css">
</head>
<body>

    <div class="container">
        <h1>Login</h1>
        <form action="LoginServlet" method="post"> <!-- Change method to "post" -->
            <label for="username">Username</label>
            <input type="text" id="username" name="username" required><br>
            <label for="password">Password</label>
            <input type="password" id="password" name="password" required><br>
            <button type="submit">Login</button>
        </form>

        <p><a href="index.html">Back to Home</a></p>
        <%-- Display error message if login fails --%>
        <% String error = request.getParameter("error");
            if (error != null && error.equals("1")) { %>
                <p style="color: red;">Invalid username or password. Please try again.</p>
        <% } %>
        </div>
</body>
</html>
```

This is our expected welcome message



Create welcome.jsp file

welcome.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Welcome</title>
<link rel="stylesheet" type="text/css" href="Oldstyle.css">
</head>
<body>
<body>
 <%
        // Retrieve the session object
        HttpSession session1 = request.getSession(false);

        // Check if the session is not null and the username attribute is set
        if (session1 != null && session1.getAttribute("username") != null) {
            // Get the username from the session
            String username = (String) session1.getAttribute("username");
    %>
      <div class="container">
        <h1>Welcome, <%=username%>!</h1>
        <h1> Enjoy Furniture Rentals!!</h1>
        you can <a href="logout.jsp">LOGOUT</a> securely.

    </div>


     <%
        } else {
            // Redirect to the login page if the session is not valid
            response.sendRedirect("login.jsp");
        }
    %>

</body>
</html>
```

Create logout.jsp file

logout.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>

<body>
 <%
        // Retrieve the session object
        HttpSession currentSession = request.getSession(false);
 if(currentSession != null){
     currentSession.invalidate();
 }


            // Redirect to the login page if the session is not valid
            response.sendRedirect("index.html");

    %>

</body>
</html>
```

FULL STACK PROJECT
VIDEO-4

Create one css file to add styles to the home page

style.css

```
@charset "ISO-8859-1";/* Google Fonts */
@import url("https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&display=swap");
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  scroll-padding-top: 2rem;
  scroll-behavior: smooth;
  list-style: none;
  text-decoration: none;
  font-family: "Poppins", sans-serif;
}
:root {
    --main-color: #fe5b3d;
    --second-color: #ffac38;
    --text-color: #444;
    --gradient: linear-gradient(#fe5b3d, #ffac38);
  }
/* Custom Scroll Bar */
html::-webkit-scrollbar {
    width: 0.5rem;
  }
  html::-webkit-scrollbar-track {
    background: transparent;
  }
  html::-webkit-scrollbar-thumb {
    background: var(--main-color);
    border-radius: 5rem;
  }
  section {
    padding: 50px 100px;
  }
```

style.css

```
header {
  position: fixed;
  width: 100%;
  top: 0;
  right: 0;
  z-index: 1000;
  display: flex;
  align-items: center;
  justify-content: space-between;
  background: #eeeff1;
  padding: 15px 100px;
}
.logo img {
  width: 40px;
}
.navbar {
  display: flex;
}
.navbar li {
  position: relative;
}
.navbar a {
  font-size: 1rem;
  padding: 10px 20px;
  color: var(--text-color);
  font-weight: 500;
}
.navbar a::after {
  content: "";
  width: 0;
  height: 3px;
  background: var(--gradient);
  position: absolute;
  bottom: -4px;
  left: 0;
  transition: 0.5s;
}
```

style.css

```
.navbar a:hover::after {
  width: 100%;
}
#menu-icon {
  font-size: 24px;
  cursor: pointer;
  z-index: 10001;
  display: none;
}
.header-btn a {
  padding: 10px 20px;
  color: var(--text-color);
  font-weight: 500;
}
.header-btn .sign-in {
  background: #474fa0;
  color: #fff;
  border-radius: 0.5rem;
}
.header-btn .sign-in:hover {
  background: var(--main-color);
}
.home {
  width: 100%;
  min-height: 100vh;
  position: relative;
  background: url(img/Sofa\ Home.png);
  background-repeat: no-repeat;
  background-position: center right;
  background-size: cover;
  display: grid;
  align-items: center;
  grid-template-columns: repeat(2, 1fr);
}
.text h1 {
  font-size: 3.5rem;
  letter-spacing: .25px;
}
```

style.css

```
}
.text span {
  color: var(--main-color);
}
.text p {
  margin: 0.5rem 0 1rem;
}
.app-stores {
  display: flex;
}
.app-stores img {
  width: 100px;
  margin-right: 1rem;
  cursor: pointer;
}
.form-container form {
  display: flex;
  flex-wrap: wrap;
  align-items: center;
  gap: 1rem;
  position: absolute;
  bottom: 4rem;
  left: 100px;
  bottom: 30px;
  background: #fff;
  padding: 20px;
  border-radius: 0.5rem;

}
.input-box {
  flex: 1 1 7rem;
  display: flex;
  flex-direction: column;
}
.input-box span {
  font-weight: 500;
}
```

style.css

```css
.input-box input {
  padding: 7px;
  outline: none;
  border: none;
  background: #eeeff1;
  border-radius: 0.5rem;
  font-size: 1rem;
}
.form-container form .btn {
  flex: 1 1 7rem;
  padding: 10px 34px;
  border: none;
  border-radius: 0.5rem;
  background: #474fa0;
  color: #fff;
  font-size: 1rem;
  font-weight: 500;
}
.form-container form .btn:hover {
  background: var(--main-color);
}
.heading {
  text-align: center;
}
.heading span {
  font-weight: 500;
  text-transform: uppercase;
}
.heading h1 {
  font-size: 2rem;
}
.service-container {
  display: grid;
  align-items: center;
  grid-template-columns: repeat(auto-fit, minmax(250px, auto));
  gap: 1rem;
  margin-top: 2rem;
}
```

style.css

```
}
.service-container .box {
  text-align: center;
  padding: 20px;
}
.service-container .box .bx {
  font-size: 34px;
  padding: 10px;
  background: #eeeff1;
  border-radius: 0.5rem;
  color: var(--main-color);
}
.service-container .box h2 {
  font-size: 1.3rem;
  font-weight: 500;
  margin: 1.4rem 0 0.5rem;
}
.service-container .box .bx:hover,
.service-container .box .bxs-calendar-check {
  background: var(--gradient);
  color: #fff;
}
.availableOnRent-container {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(300px, auto));
  gap: 1rem;
  margin-top: 2rem;
}
.availableOnRent-container .box {
  padding: 10px;
  border-radius: 1rem;
  box-shadow: 1px 4px 41px rgba(0, 0, 0, 0.1);
}
.availableOnRent-container .box .box-img {
  width: 100%;
  height: 200px;
}
```

style.css

```
}
.availableOnRent-container .box .box-img img {
  width: 100%;
  height: 100%;
  border-radius: 1rem;
  object-fit: cover;
  object-position: center;
}
.availableOnRent-container .box p {
  padding: 0 10px;
  border: 1px solid var(--text-color);
  width: 58px;
  border-radius: 0.5rem;
  margin: 1rem 0 1rem;
}
.availableOnRent-container .box h3 {
  font-weight: 500;
}
.availableOnRent-container .box h2 {
  font-size: 1.1rem;
  font-weight: 600;
  color: var(--main-color);
  margin: 0.2rem 0 0.5rem;
}
.availableOnRent-container .box h2 span {
  font-size: 0.8rem;
  font-weight: 500;
  color: var(--text-color);
}
.availableOnRent-container .box .btn {
  display: flex;
  justify-content: center;
  background: #474fa0;
  color: #fff;
  padding: 10px;
  border-radius: 0.5rem;
}
```

style.css

```
}
.availableOnRent-container .box .btn:hover {
  background: var(--main-color);
}
.about-container {
  display: grid;
  grid-template-columns: repeat(2, 1fr);
  margin-top: 2rem;
  align-items: center;
  gap: 1rem;
  margin-top: 1rem;
}
.about-img img {
  width: 100%;
}
.about-text span {
  font-weight: 500;
  color: var(--main-color);
  text-transform: uppercase;
}
.about-text p {
  margin: 0.5rem 0 1.4rem;
}
.about-text .btn {
  padding: 10px 20px;
  background: #474fa0;
  color: #fff;
  border-radius: 0.5rem;
}
.about-text .btn:hover {
  background: var(--main-color);
}
.reviews-container {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(250px, auto));
  gap: 1rem;
  margin-top: 2rem;
}
```

style.css

```
.rev-img {
  width: 70px;
  height: 70px;
}
.rev-img img {
  width: 100%;
  height: 100%;
  border-radius: 50%;
  object-fit: cover;
  object-position: center;
  border: 2px solid var(--second-color);
}
.reviews-container .box {
  display: flex;
  flex-direction: column;
  align-items: center;
  text-align: center;
  padding: 20px;
  box-shadow: 1px 4px 41px rgba(0, 0, 0, 0.1);
  border-radius: 0.5rem;
}
.reviews-container .box h2 {
  font-size: 1.1rem;
  font-weight: 600;
  margin: 0.5rem 0 0.5rem;
}
.reviews-container .box p {
  font-style: italic;
}
.reviews-container .box .stars .bx {
  color: var(--main-color);
}
.newsletter {
  background: linear-gradient(to top right, #fe5b3d, #ffac38);
  display: flex;
  flex-direction: column;
  align-items: center;
}
```

style.css

```
.newsletter h2 {
  color: #fff;
  font-size: 1.8rem;
}
.newsletter .box {
  margin-top: 1rem;
  background: #fff;
  border-radius: 0.5rem;
  padding: 4px 8px;
  width: 350px;
  display: flex;
  justify-content: space-between;
}
.newsletter .box input {
  border: none;
  outline: none;
}
.newsletter .box .btn {
  background: #474fa0;
  color: #fff;
  padding: 8px 20px;
  border-radius: 0.5rem;
}
.copyright {
  padding: 20px;
  display: flex;
  justify-content: space-between;
  align-items: center;
}
.social a {
  color: #444;
  padding: 10px;
  font-size: 21px;
}
/* Making Responsive */
```

style.css

```
@media (max-width: 991px) {
  header {
    padding: 18px 40px;
  }
  section {
    padding: 50px 40px;
  }
}
@media (max-width: 881px) {
  .home {
    background-position: left;
  }
  .form-container form {
    bottom: 0.2rem;
    left: 40px;
  }
}
@media (max-width: 768px) {
  header {
    padding: 11px 40px;
  }
  #menu-icon {
    display: initial;
  }
  .sign-up {
    display: none;
  }
  .text h1 {
    font-size: 2.5rem;
  }
  .home {
    grid-template-columns: 1fr;
  }
  .form-container form {
    position: unset;
  }
```

style.css

```
header .navbar {
  position: absolute;
  top: -500px;
  left: 0;
  right: 0;
  display: flex;
  flex-direction: column;
  background: #fff;
  box-shadow: 0 4px 4px rgba(0, 0, 0, 0.1);
  transition: 0.2s ease;
  text-align: left;
}
.navbar.active {
  top: 100%;
}
.navbar a {
  padding: 1rem;
  border-left: 2px solid var(--main-color);
  margin: 1rem;
  display: block;
}
.navbar a:hover {
  color: #fff;
  background: var(--main-color);
  border: none;
}
.navbar a::after {
  display: none;
}
.heading span {
  font-size: 0.9rem;
  font-weight: 600;
}
.heading h1 {
  font-size: 1.3rem;
}
```

style.css

```
435   }
436   .about-container {
437     grid-template-columns: 1fr;
438     text-align: center;
439   }
440   .about-img {
441     padding: 1rem;
442     order: 2;
443   }
444 }
445 @media (max-width: 568px) {
446   .copyright {
447     flex-direction: column;
448   }
449   .newsletter .box {
450     width: 284px;
451   }
452   .form-container {
453     padding-top: 2rem;
454   }
455 }
456 @media (max-width: 350px) {
457   header {
458     padding: 4px 14px;
459   }
460   .logo img {
461     width: 30px;
462   }
463   section {
464     padding: 50px 14px;
465   }
466   .header-btn .sign-in {
467     padding: 7px 10px;
468     font-size: 14px;
469     font-weight: 400;
470   }
471   .text h1 {
472     font-size: 2rem;
473   }
474   .availableOnRent-container {
475     grid-template-columns: repeat(auto-fit, minmax(254px, auto));
476   }
477 }
```

Create another css file to add styles to the login, register and welcome pages

Oldstyle.css

```css
@charset "ISO-8859-1";
@import url("https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&display=swap");
* {
  font-family: "Poppins", sans-serif;
}
body {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    height: 100vh;
    margin: 0;
     background-color: #AEFD6C;
     background-image: url('1.png');
     background-size: cover; /* or 'contain' based on your preference*/
     background-position: center center;
     background-repeat: no-repeat;

}

.container {
    width: 80%;
    max-width: 400px;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0, 0, 1, 0.5);
    background-color: #AEFD6C;
    text-align: center; /* Center the content*/
}

h1 {
    font-weight: bold;
    text-align: center;
    color: #333;
}
```

Oldstyle.css

```css
h1 {
    font-weight: bold;
    text-align: center;
    color: #333;
}

label {
    display: block;
    margin: 10px 0;
    color: #555;
}

input {
    width: 100%;
    padding: 10px;
    margin-bottom: 15px;
    box-sizing: border-box;
    border: 1px solid #ccc;
    border-radius: 5px;
    font-size: 16px;
}

button {
    width: 100%;
    padding: 12px;
    cursor: pointer;
    background-color: #fe5b3d;
    color: #fff;
    border: none;
    border-radius: 5px;
    font-size: 16px;
    transition: background-color 0.3s ease, transform 0.3s ease;
}

button:hover {
    background-color: #0056b3;
    transform: scale(1.05);
}
```

Oldstyle.css

```
input {
    width: 100%;
    padding: 10px;
    margin-bottom: 15px;
    box-sizing: border-box;
    border: 1px solid #ccc;
    border-radius: 5px;
    font-size: 16px;
}

button {
    width: 100%;
    padding: 12px;
    cursor: pointer;
    background-color: #fe5b3d;
    color: #fff;
    border: none;
    border-radius: 5px;
    font-size: 16px;
    transition: background-color 0.3s ease, transform 0.3s ease;
}

button:hover {
    background-color: #0056b3;
    transform: scale(1.05);
}

a {
    text-decoration: none;
    color: #fe5b3d;;
    font-weight: bold;
}

a:hover {
    color: #0056b3;
}

/*@charset "ISO-8859-1";*/
```

Oldstyle.css

```
/*@charset "ISO-8859-1";*/
body {
    font-family: 'Arial', sans-serif;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    height: 100vh;
    margin: 0;
    background-color: #AEFD6C;
    background-image: url('1.png');
    background-size: cover;
    background-position: center center;
    background-repeat: no-repeat;
    color: #333;
}

.container {
    width: 600px;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0, 0, 1, 0.5);
    background-color: white;
    text-align: center;
}

h1 {
    font-weight: bold;
    color: #fe5b3d;
}

p {
    margin: 20px 0;
    line-height: 1.6;
}
```

```
.emoji {
    font-size: 24px;
    margin-right: 5px;
}


.links a {
    margin: 0 10px;
    text-decoration: none;
    color: #fe5b3d;
    font-weight: bold;
    transition: color 0.3s ease;
}

.links a:hover {
    color: #fe5b3d;
    font-size: 20px;
    text-transform: uppercase;
}

```

Create main.js file

main.js

```js
let menu = document.querySelector('#menu-icon');
let navbar = document.querySelector('.navbar');

menu.onclick = () => {
    menu.classList.toggle('bx-x');
    navbar.classList.toggle('active');
}

window.onscroll = () => {
    menu.classList.remove('bx-x');
    navbar.classList.remove('active');
}

const sr = ScrollReveal ({
    distance: '60px',
    duration: 2500,
    delay: 400,
    reset: true
})

sr.reveal('.text',{delay: 200, origin: 'top'})
sr.reveal('.form-container form',{delay: 800, origin: 'left'})
sr.reveal('.heading',{delay: 800, origin: 'top'})
sr.reveal('.service-container .box',{delay: 600, origin: 'top'})
sr.reveal('.availableOnRent-container .box',{delay: 600, origin: 'top'})
sr.reveal('.about-container .box',{delay: 600, origin: 'top'})
sr.reveal('.reviews-container',{delay: 600, origin: 'top'})
sr.reveal('.newsletter .box',{delay: 400, origin: 'bottom'})
```

FULL STACK PROJECT
VIDEO-5

- Till now we have worked on front end development.
- So we have created a home page, register page, login page.
- Let us try to register by giving the credentials.

Till now we have not yet given any dynamic action to the register page. For that we need to create a registerServlet class

Create RegisterServlet.file

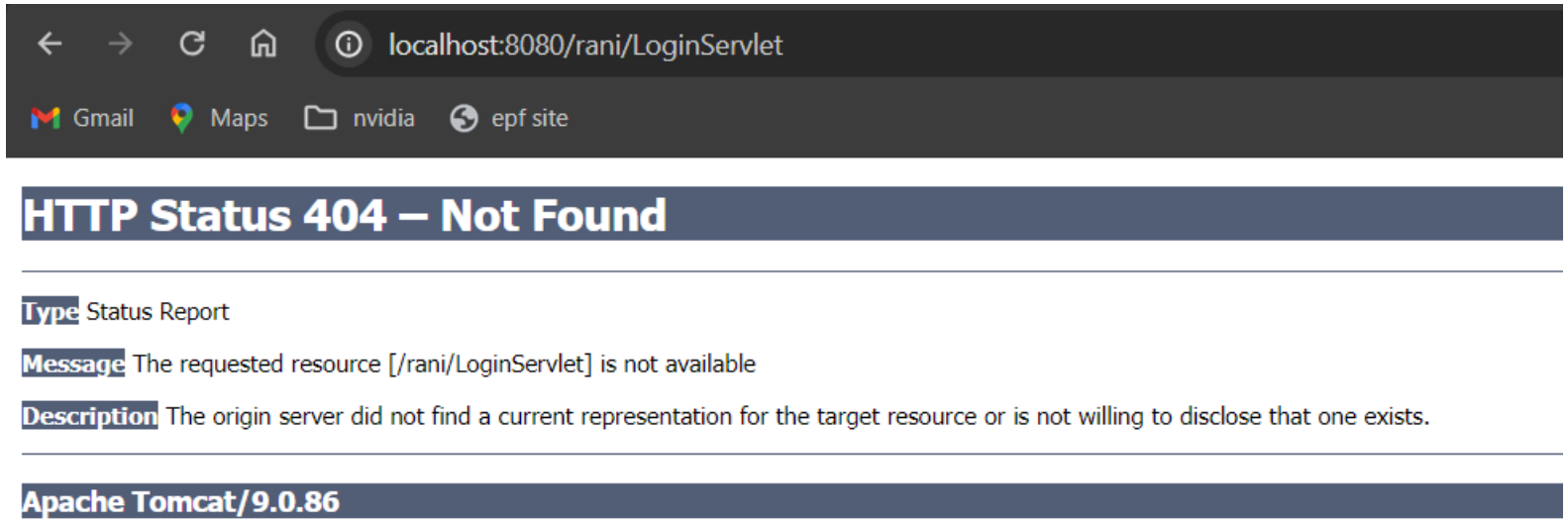
LoginServlet.java | RegisterServlet.java

```java
package com.company.servlet;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.company.dao.UserDaoImpl;
import com.company.model.User;

/**
 * Servlet implementation class RegisterServlet
 */
@WebServlet("/RegisterServlet")
public class RegisterServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    private static UserDaoImpl userDao = new UserDaoImpl();

    /**
     * @see HttpServlet#HttpServlet()
     */
    public RegisterServlet() {
        super();
        // TODO Auto-generated constructor stub
    }


    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
```

```java
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

        String username = request.getParameter("username");
           String password = request.getParameter("password");
           String email = request.getParameter("email");

           User user = new User();
           user.setUsername(username);
           user.setEmail(email);
           user.setPassword(password);

           //UserDao userDao = new UserDaoImpl();

           if (userDao.addUser((User) user)) {
               response.sendRedirect("login.jsp?registration=success");
           } else {
               response.sendRedirect("register.jsp?error=1");
           }
       }
}
```

- We have created register page.
- Try to register by giving the username, email id and create a password.
- After registering try to login the login page.

We have not yet created a LoginServlet class



Create LoginServlet.file

LoginServlet.java

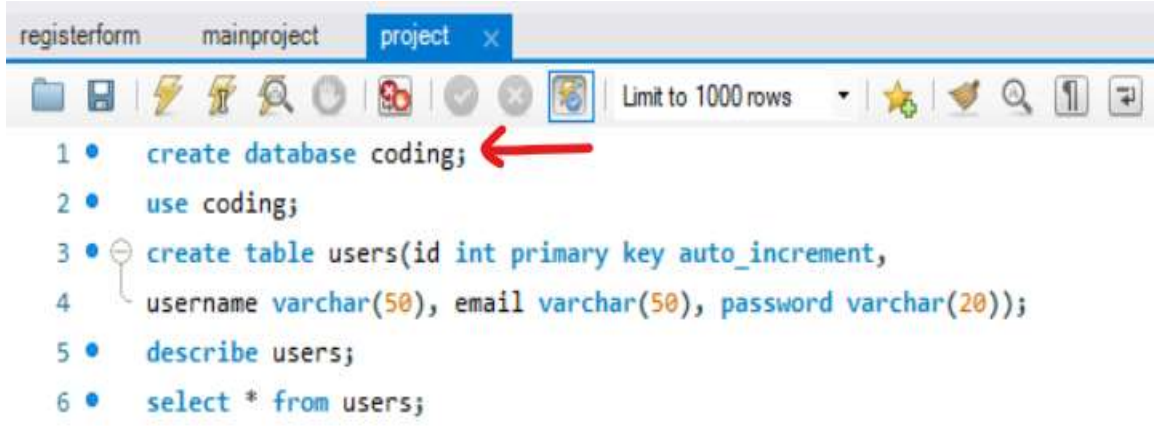
```java
package com.company.servlet;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.company.dao.UserDaoImpl;

/**
 * Servlet implementation class LoginServlet
 */
@WebServlet("/LoginServlet")
public class LoginServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    private static UserDaoImpl userdao = new UserDaoImpl();

    /**
     * @see HttpServlet#HttpServlet()
     */
    public LoginServlet()
    {
        super();
        // TODO Auto-generated constructor stub
    }


    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
```

```java
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub
        String username = request.getParameter("username");
        String password = request.getParameter("password");

        if (userdao.isValidUser(username, password)) {
            HttpSession session = request.getSession();
            session.setAttribute("username", username);
           response.sendRedirect("welcome.jsp");
           // System.out.println("Hi - "+username);
        }

        else {
            response.sendRedirect("login.jsp?error=1");
            System.out.println("Error");
        }

    }
}
```

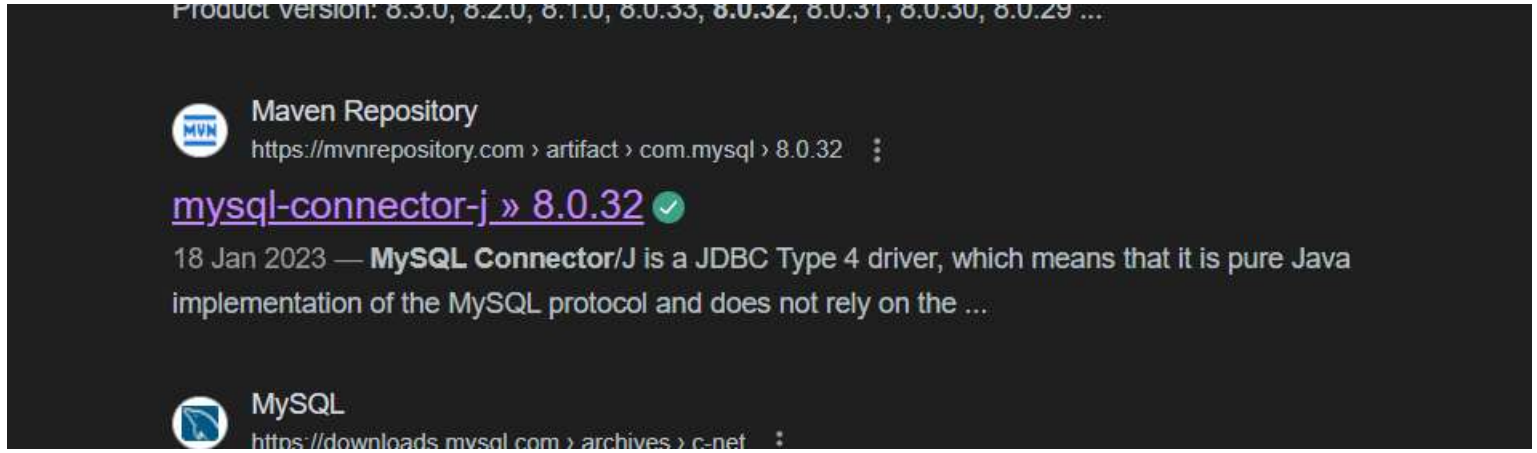
FULL STACK PROJECT
VIDEO-6

step 5 create a database

I have created a database with the name of coding. To communicate with this database we need to add the mysql connector to the eclipse IDE



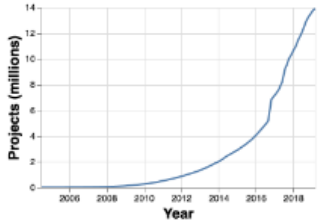
- Go to the browser type mysql connector 8.0.32.jar



MVN REPOSITORY

Search for groups, artifacts, categories

Indexed Artifacts (35.5M)



Popular Categories

- Testing Frameworks & Tools
- Android Packages
- Logging Frameworks
- Java Specifications
- JSON Libraries
- JVM Languages
- Language Runtime
- Core Utilities
- Mocking
- Web Assets
- Annotation Libraries
- HTTP Clients

Home » com.mysql » mysql-connector-j » 8.0.32

MySQL Connector/J » 8.0.32

MySQL Connector/J is a JDBC Type 4 driver, which means that it is pure Java implementation of the MySQL protocol and does the MySQL client libraries. This driver supports auto-registration with the Driver Manager, standardized validity checks, categc SQLExceptions, support for large update counts, support for local and offset date-time variants from the java.time package, su JDBC-4.x XML processing, support for per connection client information and support for the NCHAR, NVARCHAR ...

Categories	JDBC Drivers
Tags	database sql jdbc driver connector rdbms mysql connection
Organization	Oracle Corporation
HomePage	http://dev.mysql.com/doc/connector-j/en/
Date	Jan 18, 2023
Files	pom (3 KB) jar (2.4 MB) View All
Repositories	Central
Ranking	**#787 in MvnRepository (See Top Artifacts)** **#9 in JDBC Drivers**
Used By	**630 artifacts**

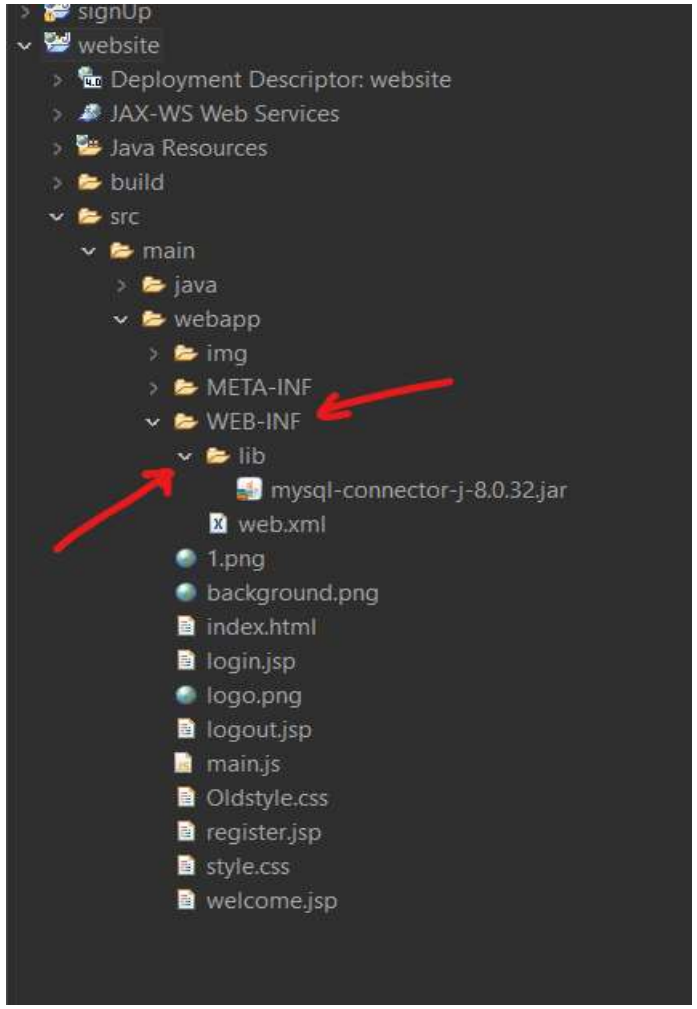
Note: There is a new version for this artifact

New Version	8.4.0

com/mysql/mysql-connector-j/8.0.32

../		
mysql-connector-j-8.0.32-javadoc.jar	2023-01-18 16:04	1091416
mysql-connector-j-8.0.32-javadoc.jar.asc	2023-01-18 16:04	836
mysql-connector-j-8.0.32-javadoc.jar.md5	2023-01-18 16:04	32
mysql-connector-j-8.0.32-javadoc.jar.sha1	2023-01-18 16:04	40
mysql-connector-j-8.0.32-sources.jar	2023-01-18 16:04	1607222
mysql-connector-j-8.0.32-sources.jar.asc	2023-01-18 16:04	836
mysql-connector-j-8.0.32-sources.jar.md5	2023-01-18 16:04	32
mysql-connector-j-8.0.32-sources.jar.sha1	2023-01-18 16:04	40
mysql-connector-j-8.0.32.jar	2023-01-18 16:04	2480823
mysql-connector-j-8.0.32.jar.asc	2023-01-18 16:04	836
mysql-connector-j-8.0.32.jar.md5	2023-01-18 16:04	32
mysql-connector-j-8.0.32.jar.sha1	2023-01-18 16:04	40
mysql-connector-j-8.0.32.pom	2023-01-18 16:04	3180
mysql-connector-j-8.0.32.pom.asc	2023-01-18 16:04	836
mysql-connector-j-8.0.32.pom.md5	2023-01-18 16:04	32
mysql-connector-j-8.0.32.pom.sha1	2023-01-18 16:04	40

- Copy that downloaded mysql connect and paste it in to the project in WEB-INF under lib folder



Step:6 Establish the connection to the database we need to create a class called DBUtil

UserDaoImpl.java | UserDao.java | User.java | DBUtil.java

```java
package com.company.util;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DBUtil {
     private static final String URL = "jdbc:mysql://localhost:3306/coding";
     private static final String USERNAME = "root";
     private static final String PASSWORD = "Chinni@27";
    static {
        try {
            // Load the MySQL JDBC driver
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL, USERNAME, PASSWORD);
    }
}
```

Interact with the database create a class called UserDaoImpl

UserDaoImpl.java

```java
package com.company.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import com.company.model.User;
import com.company.util.DBUtil;

public class UserDaoImpl implements UserDao {
    @Override
        public boolean addUser(User user) {
            String query = "INSERT INTO users (username, email, password) VALUES (?, ?, ?)";

            try (Connection connection = DBUtil.getConnection();
                 PreparedStatement preparedStatement = connection.prepareStatement(query)) {

                preparedStatement.setString(1, user.getUsername());
                preparedStatement.setString(2, user.getEmail());
                preparedStatement.setString(3, user.getPassword());

                int rowsAffected = preparedStatement.executeUpdate();

                return rowsAffected > 0;
            } catch (SQLException e) {
                e.printStackTrace();
                return false;
            }
        }

```

```java
    }

    @Override
    public boolean isValidUser(String username, String password) {
        String query = "SELECT * FROM users WHERE username = ? AND password = ?";
         try (Connection connection = DBUtil.getConnection();
             PreparedStatement preparedStatement = connection.prepareStatement(query)) {

            preparedStatement.setString(1, username);
            preparedStatement.setString(2, password);

            ResultSet resultSet = preparedStatement.executeQuery();

            return resultSet.next();
        } catch (SQLException e) {
            e.printStackTrace();
            return false;
        }
    }

}
```

Create a java interface called UserDao for handling the user data in a system.

UserDao.java

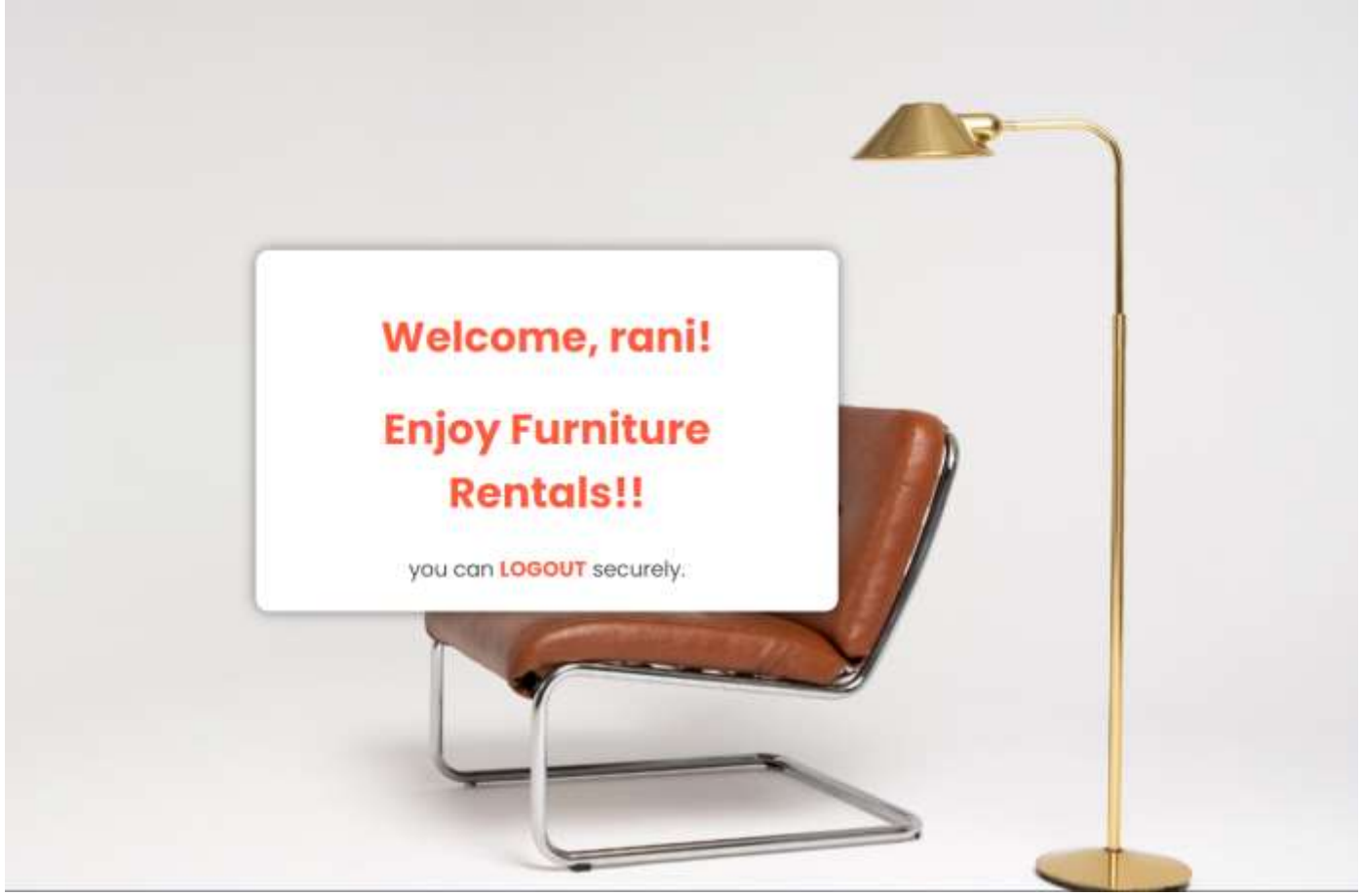
```java
package com.company.dao;

import com.company.model.User;

public interface UserDao {

    boolean addUser(User user);
    boolean isValidUser(String username, String password);

}
```

- Create a class called User.java this class helps to provides a basic structure to represent a user in a system. It is used to encapsulates the users data(username, password and email)

User.java

```java
package com.company.model;

public class User {
     private String username;
     private String password;
     private String email;
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }


}
```

Final output



And this is how user data is stored in database

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	username	email	password
1	rani	raninanu1227@gmail.com	1234
2	chinni	chinni123@gmail.com	chinni
NULL	NULL	NULL	NULL

All the best