Name : Mohil Mokaria

Class : 6CE

Division : E

Subject : Java Experiment 6

Q – Create a web application where users can save and organize their favorite recipes. The application should use cookies to remember user preferences, such as display settings and recently viewed recipes.

 Perform Following steps to implement Cookie in Servlet and JSP.

1. **Recipe Servlet and JSP:**
   * Create a servlet to manage recipes (create, read, update, delete).
   * Design a JSP page to display recipes and allow users to interact with them.
2. **User Preferences Servlet and Cookies:**
   * Implement a servlet to handle user preferences.
   * Use cookies to store and retrieve user preferences, such as theme and display settings.
3. **Recently Viewed Recipes Servlet and Cookies:**
   * Develop a servlet to manage recently viewed recipes.
   * Use cookies to store and update the list of recently viewed recipes.
4. **Search and Filter Servlet:**
   * Create a servlet to handle recipe searches and filtering.
   * Use cookies to store search and filter preferences.
5. **User Authentication:**
   * Implement user registration and login functionalities.
   * Use cookies for session management after successful login.

### ingredient.java

package myClass;

public class Ingredient {

private String name;

private String quantity;

public Ingredient() {

}

public Ingredient(String name, String quantity) {

this.name = name;

this.quantity = quantity;

}

// Getters and setters

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getQuantity() {

return quantity;

}

public void setQuantity(String quantity) {

this.quantity = quantity;

}

}

### recipe.java

package myClass;

import java.util.List;

public class Recipe {

private Integer id;

private String auth;

private String title;

private List<Ingredient> ingredients;

private String steps;

private Boolean isVeg;

public Recipe() {

}

// Parameterized constructor

public Recipe(Integer id, String auth, String title, List<Ingredient> ingredients, String steps, Boolean isVeg) {

this.id = id;

this.auth = auth;

this.title = title;

this.ingredients = ingredients;

this.steps = steps;

this.isVeg = isVeg;

}

public Integer getId() {

return this.id;

}

public String getAuth() {

return this.auth;

}

public String getTitle() {

return this.title;

}

public String getSteps() {

return this.steps;

}

public Boolean getIsVeg() {

return this.isVeg;

}

public void setId(Integer id) {

this.id = id;

}

public void setAuth(String auth) {

this.auth = auth;

}

public void setTitle(String title) {

this.title = title;

}

public void setSteps(String steps) {

this.steps = steps;

}

public void setIsVeg(Boolean isVeg) {

this.isVeg = isVeg;

}

public List<Ingredient> getIngredients() {

return ingredients;

}

public void setIngredients(List<Ingredient> ingredients) {

this.ingredients = ingredients;

}

}

# index.jsp :

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Registration Form</title>

<!-- BOOTSTRAP CDNs -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL" crossorigin="anonymous"></script>

<!-- LOCAL CSS LINK -->

<link rel="stylesheet" href="./index\_styles.css"/>

<script>

function validator(){

var emailRegex = /^[^\s@]+@[^\s@]+\.[a-z]{2,}$/i;

var email = document.getElementById("mail").value;

var pass1 = document.getElementById("pass1").value;

var pass2 = document.getElementById("pass2").value;

if(!emailRegex.test(email)){

alert("Invalid Email !");

return false;

}

if(pass1 !== pass2){

alert("Password Doesn't Match ! Try Again");

return false;

}

if(pass1.length < 6 || pass1.length > 12 ){

alert("Password must be 6 to 12 characters long !");

return false;

}

document.getElementById("regform").submit();

return true;

}

</script>

</head>

<body>

<form id="regform" action="SignupServlet" method="post" onsubmit="return validator()">

<h1>User Sign-up Form</h1>

<%

String error = (String) request.getAttribute("error");

if(error != null && !error.isEmpty()){

%>

<p class="alert alert-danger"><%= error %></p>

<%

}

%>

<label for="mail">Email ID : </label>

<input type="email" id="mail" name="mail" autofocus required>

<label for="pass1">New Password : </label>

<input type="password" id="pass1" name="pass1" required>

<label for="pass2">Confirm Password : </label>

<input type="password" id="pass2" name="pass2" required>

<center><button type="submit">Sign-up</button></center>

<p>Already Signed-up? <a href="./login.jsp">Login</a></p>

<%

String exceptionMsg = (String) request.getAttribute("exception");

if(exceptionMsg != null && !exceptionMsg.isEmpty()){

%>

<p class="alert alert-danger"><%= exceptionMsg %></p>

<%

}

%>

</form>

</body>

</html>

# login.jsp :

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Registration Form</title>

<!-- BOOTSTRAP CDNs -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL" crossorigin="anonymous"></script>

<!-- LOCAL CSS LINK -->

<link rel="stylesheet" href="./index\_styles.css"/>

</head>

<body>

<form id="regform" action="LoginServlet" method="post">

<h1>User Login Form</h1>

<%

String error = (String) request.getAttribute("error");

if(error != null && !error.isEmpty()){

%>

<p class="alert alert-danger"><%= error %></p>

<%

}

%>

<label for="mail">Email ID : </label>

<input type="email" id="mail" name="mail" autofocus required>

<label for="pass">Your Password : </label>

<input type="password" id="pass" name="pass" required>

<center><button type="submit">Login</button></center>

<p>Don't have a account? <a href="./index.jsp">Sign-up</a></p>

<%

String exceptionMsg = (String) request.getAttribute("exception");

if(exceptionMsg != null && !exceptionMsg.isEmpty()){

%>

<p class="alert alert-danger"><%= exceptionMsg %></p>

<%

}

%>

</form>

</body>

</html>

### script.js

function disableButton() {

document.getElementById("submitButton").disabled = true;

}

function refreshPage() {

location.reload();

}

function setRecentRecipeCookie(recipeName) {

// Retrieve the existing cookie value

var existingCookie = getCookie("recentRecipe");

// Check if the recipeName already exists in the cookie

if (existingCookie && existingCookie.includes(recipeName)) {

return; // Exit function if the recipeName already exists

}

// Append the new recipe name to the existing value

var updatedValue = existingCookie ? existingCookie + "," + recipeName : recipeName;

// Set the cookie with the updated value

document.cookie = "recentRecipe=" + encodeURIComponent(updatedValue) + "; max-age=" + (24 \* 60 \* 60);

}

// Function to retrieve cookie value by name

function getCookie(name) {

var cookies = document.cookie.split(';');

for (var i = 0; i < cookies.length; i++) {

var cookie = cookies[i].trim();

if (cookie.indexOf(name + "=") === 0) {

return decodeURIComponent(cookie.substring(name.length + 1));

}

}

return null;

}

function navigateToURL() {

var currentURL = window.location.href;

window.location.href = currentURL;

}

function addIngredient() {

var container = document.getElementById('ingredientsSection');

var newIngredient = document.createElement('div');

newIngredient.className = 'form-outline mb-4 ingredient-container';

newIngredient.innerHTML = `

<div class="d-flex justify-content-around">

<div>

<input type="text" name="ingredientName[]" class="form-control" placeholder="Name" required/>

</div>

<div>

<input type="text" name="ingredientQuantity[]" class="form-control" placeholder="Quantity" required/>

</div>

<div>

<button type="button" class="btn btn-outline-danger" onclick="removeIngredient(this)"><i class="fa-solid fa-trash"></i></button>

</div>

</div>

`;

container.appendChild(newIngredient);

}

document.getElementById('searchButton').addEventListener('click', function (event) {

event.preventDefault(); // Prevent the default form submission behavior

var searchInput = document.getElementById('searchInput').value;

var xhr = new XMLHttpRequest();

xhr.open('GET', 'SearchServlet?query=' + encodeURIComponent(searchInput), true);

xhr.onreadystatechange = function () {

if (xhr.readyState === XMLHttpRequest.DONE) {

if (xhr.status === 200) {

var response = xhr.responseText;

// Update the DOM with the search results

document.getElementById('searchResults').innerHTML = response;

} else {

console.error('Error:', xhr.status);

}

}

};

xhr.send();

});

# home.jsp :

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<%@ page import="myClass.Recipe" %>

<%@ page import="java.util.List" %>

<%@ page import="myClass.Recipe" %>

<%@ page import="myClass.Ingredient" %>

<%@ page import="java.net.URLDecoder" %>

<!DOCTYPE html>

<html>

<head>

<title>Simple Recipe App</title>

<!--FONT AWESOME CDN-->

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css" integrity="sha512-DTOQO9RWCH3ppGqcWaEA1BIZOC6xxalwEsw9c2QQeAIftl+Vegovlnee1c9QX4TctnWMn13TZye+giMm8e2LwA==" crossorigin="anonymous" referrerpolicy="no-referrer" />

<!-- BOOTSTRAP CDNs -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL" crossorigin="anonymous"></script>

<!-- LOCAL CSS LINK -->

<link rel="stylesheet" href="./home\_style.css" />

</head>

<body>

<%

if (session.getAttribute("userEmail") != null) {

%>

<div class="d-flex justify-content-around mb-4">

<div class="navbar-text mr-3">

<strong><%= session.getAttribute("userEmail")%></strong>

</div>

<div class="d-flex w-50">

<input class="form-control me-2" id="searchInput" type="search" placeholder="Recipe Name to Search">

<button class="btn btn-outline-success" id="searchButton" type="submit">Search</button>

</div>

<div class="row">

<span class="col">

<button type="button" class="btn btn-outline-primary" data-bs-toggle="modal" data-bs-target="#recentRecipesModal">

<i class="fa-solid fa-clock-rotate-left"></i>

</button>

</span>

<span class="col">

<form action="LogoutServlet" method="get">

<button class="btn btn-outline-danger"><i class="fa-solid fa-power-off"></i></button>

</form>

</span>

</div>

</div>

<center><div id="searchResults"></div></center>

<%

String exceptionMsg = (String) session.getAttribute("exception");

if (exceptionMsg != null && !exceptionMsg.isEmpty()) {

%>

<p class="alert alert-danger"><%= exceptionMsg%></p>

<%

}

%>

<br>

<hr class="border border-primary border-2">

<div class="container">

<div class="row align-items-center">

<div class="col-md-10 col-sm-8 mx-auto">

<div class="card shadow align-items-center">

<div class="card-body d-flex flex-column col-md-9 col-sm-6 mb-4 mt-4">

<form action="AddRecipeServlet" method="post">

<%

String error = (String) request.getAttribute("error");

if (error != null && !error.isEmpty()) {

%>

<p class="alert alert-danger"><%= error%></p>

<%

}

%>

<div class="mb-4">

<h2>Create New Recipe</h2>

</div>

<div class="form-outline mb-4">

<label for="title" class="form-label">Recipe Title:</label>

<input type="text" id="title" name="title" class="form-control" required/>

</div>

<label class="form-label">Ingredients List:</label>

<div class="card-body">

<div id="ingredientsSection">

<div class="form-outline mb-4 ingredient-container">

<div class="d-flex justify-content-around">

<div>

<input type="text" name="ingredientName[]" class="form-control" placeholder="Name" required/>

</div>

<div>

<input type="text" name="ingredientQuantity[]" class="form-control" placeholder="Quantity" required/>

</div>

<div>

<button type="button" class="btn btn-outline-dark" onclick="removeIngredient(this)" disabled><i class="fa-solid fa-trash"></i></button>

</div>

</div>

</div>

<!--SECTION WHERE INGREDIENTS WILL BE ADDED-->

</div>

<div><button type="button" class="btn btn-primary" onclick="addIngredient()"><i class="fa-solid fa-plus"></i> Add Ingredient</button></div>

</div>

<div class="form-outline mb-4">

<label class="form-label" for="steps">Steps:</label>

<textarea id="steps" name="steps" class="form-control" required></textarea>

</div>

<div class="card-body d-flex justify-content-center">

<div class="form-check">

<input class="form-check-input" type="radio" name="veg" value="true" id="veg" checked>

<label class="form-check-label" for="veg">

Veg

</label>

</div>

<div class="form-check ms-4">

<input class="form-check-input" type="radio" name="veg" value="false" id="nonveg">

<label class="form-check-label" for="nonveg">

Non-Veg

</label>

</div>

</div>

<center><button type="submit" id="submitButton" class="btn btn-outline-primary btn-block mb-2 sendbtn">Send</button></center>

</form>

</div>

</div>

</div>

</div>

</div>

<section id="my\_card">

<div class="row row-cols-1 row-cols-md-4 g-4">

<%

List<Recipe> recipeList = (List<Recipe>) request.getAttribute("recipeList");

if (recipeList != null && !recipeList.isEmpty()) {

int modalCount = 1;

for (Recipe r : recipeList) {

String modalId = "recipeModel" + modalCount;

%>

<div class="col">

<div class="card" style="width: 18rem;">

<div class="card-body">

<div class="d-flex justify-content-between">

<h5 class="card-title"><%= r.getTitle()%></h5>

<%

String iconName = r.getIsVeg() ? "fa-leaf" : "fa-drumstick-bite";

String circleColor = r.getIsVeg() ? "green" : "red";

%>

<i class="fa-solid <%= iconName %> align-self-center" style="color: <%= circleColor %>; font-size: smaller;"></i>

</div>

<p class="card-text">CHEF : <%= r.getAuth()%></p>

<button type="button" class="btn btn-outline-primary" onclick="setRecentRecipeCookie('<%= r.getTitle() %>')" data-bs-toggle="modal" data-bs-target="#<%= modalId %>">Read</button>

<!--MODEL1 for recipe View-->

<div class="modal fade" id="<%= modalId %>" tabindex="-1" aria-labelledby="<%= modalId %>Label" aria-hidden="true">

<div class="modal-dialog modal-dialog-centered modal-dialog-scrollable">

<div class="modal-content">

<div class="modal-header">

<div class="d-flex">

<i class="fa-solid <%= iconName %> align-self-center" style="color: <%= circleColor %>; font-size: smaller;"></i>

<h5 class="modal-title ms-4" id="<%= modalId %>Label"><%= r.getTitle()%> by <%= r.getAuth()%></h5>

</div>

</div>

<div class="modal-body">

<h3>Ingredients :</h3>

<ol>

<%

List<Ingredient> ingredientsList = r.getIngredients();

if (ingredientsList != null && !ingredientsList.isEmpty()) {

for (Ingredient i : ingredientsList) {

%>

<li><%= i.getName() %> - <%= i.getQuantity() %></li>

<%

}

} else {

%>

<p>No ingredients found</p>

<%

}

%>

</ol>

<hr>

<h3>Steps :</h3>

<p><%= r.getSteps()%></p>

</div>

<div class="modal-footer">

<button type="button" onclick="navigateToURL()" class="btn btn-primary" data-bs-dismiss="modal">Close</button>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

<%

modalCount++;

}

} else {

%>

<div class="card mt-4 messages\_card">

<div class="card-body">

<div class="row">

<h5 class="card-title col">No Recipes Made Yet</h5>

</div>

<p class="card-text">Try to Add New Recipes!</p>

</div>

</div>

<%

}

%>

</div>

</section>

<% } else { %>

<hr class="border border-primary border-2">

<div class="container">

<div class="row align-items-center">

<div class="col-md-6 col-sm-6 mx-auto">

<div class="card shadow align-items-center">

<div class="card-body d-flex flex-column col-md-9 col-sm-6 mb-4 mt-4">

<p><center>Please log in to access the chat page.</center></p>

<a class="btn btn-outline-primary btn-block" href="login.jsp">Login</a>

</div>

</div>

</div>

</div>

</div>

<% }%>

<hr class="border border-primary border-2">

<!-- Modal2 for displaying cookies -->

<div id="contentToRefresh">

<div class="modal fade" id="recentRecipesModal" data-bs-backdrop="static" data-bs-keyboard="false" tabindex="-1" aria-labelledby="recentRecipesModalLabel" aria-hidden="true">

<div class="modal-dialog modal-dialog-centered modal-dialog-scrollable">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="recentRecipesModalLabel">Recently Viewed Recipes</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body">

<%

Cookie[] cookies = request.getCookies();

boolean foundRecipe = false;

if (cookies != null) {

for (Cookie cookie : cookies) {

if (cookie.getName().equals("recentRecipe")) {

String recentRecipe = URLDecoder.decode(cookie.getValue(), "UTF-8");

if (recentRecipe != null && !recentRecipe.isEmpty()) {

%>

<p><%= response.encodeURL(recentRecipe) %></p>

<%

foundRecipe = true;

}

}

}

}

if (!foundRecipe) {

// No recently viewed recipes

%>

<p>No recently viewed recipes</p>

<%

}

%>

</div>

</div>

</div>

</div>

</div>

<!--Local Java Script-->

<script src="script.js"></script>

</body>

</html>

### SignupServlet.java

import jakarta.servlet.RequestDispatcher;

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

@WebServlet("/SignupServlet")

public class SignupServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

@Override

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

String jdbcUri = getServletContext().getInitParameter("jdbcUri");

String dbUri = getServletContext().getInitParameter("dbUri");

String dbId = getServletContext().getInitParameter("dbId");

String dbPass = getServletContext().getInitParameter("dbPass");

response.setContentType("text/html");

PrintWriter pw = response.getWriter();

String email = request.getParameter("mail");

String pass1 = request.getParameter("pass1");

try {

Class.forName(jdbcUri);

try (Connection con = DriverManager.getConnection(dbUri, dbId, dbPass)) {

PreparedStatement ps1 = con.prepareStatement("SELECT \* FROM chatusers WHERE mail=?");

ps1.setString(1, email);

ResultSet rs = ps1.executeQuery();

if (rs.next()) {

String error = "User already registered with <br>" + email;

request.setAttribute("error", error);

RequestDispatcher rd = request.getRequestDispatcher("index.jsp");

rd.include(request, response);

} else {

PreparedStatement ps2 = con.prepareStatement("INSERT INTO chatusers(mail, pass) values(?,?)");

ps2.setString(1, email);

ps2.setString(2, pass1);

int i = ps2.executeUpdate();

response.sendRedirect("./login.jsp");

}

}

} catch (SQLException | ClassNotFoundException ex) {

Logger.getLogger(SignupServlet.class.getName()).log(Level.SEVERE, null, ex);

request.setAttribute("exception", ex.getMessage());

RequestDispatcher rd = request.getRequestDispatcher("index.jsp");

rd.include(request, response);

}

}

}

### LoginServlet.java

import jakarta.servlet.RequestDispatcher;

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import jakarta.servlet.http.HttpSession;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

@WebServlet("/LoginServlet")

public class LoginServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

@Override

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

String jdbcUri = getServletContext().getInitParameter("jdbcUri");

String dbUri = getServletContext().getInitParameter("dbUri");

String dbId = getServletContext().getInitParameter("dbId");

String dbPass = getServletContext().getInitParameter("dbPass");

response.setContentType("text/html");

PrintWriter pw = response.getWriter();

String email = request.getParameter("mail");

String pass = request.getParameter("pass");

try {

Class.forName(jdbcUri);

try (Connection con = DriverManager.getConnection(dbUri, dbId, dbPass)) {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM chatusers WHERE mail=? AND pass=?");

ps.setString(1, email);

ps.setString(2, pass);

ResultSet rs = ps.executeQuery();

if (rs.next()) {

HttpSession session = request.getSession();

session.setAttribute("userEmail", email);

response.sendRedirect("NewNote");

} else {

String error = "Invalid Login Credentials!";

request.setAttribute("error", error);

RequestDispatcher rd = request.getRequestDispatcher("login.jsp");

rd.include(request, response);

}

}

} catch (SQLException | ClassNotFoundException ex) {

Logger.getLogger(LoginServlet.class.getName()).log(Level.SEVERE, null, ex);

request.setAttribute("exception", ex.getMessage());

RequestDispatcher rd = request.getRequestDispatcher("login.jsp");

rd.include(request, response);

}

}

}

### LogoutServlet.java

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import jakarta.servlet.http.HttpSession;

import java.io.IOException;

@WebServlet("/LogoutServlet")

public class LogoutServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

@Override

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

// Invalidate the session

HttpSession session = request.getSession(false);

if (session != null) {

session.invalidate();

}

// Redirect to the login page after logout

response.sendRedirect("login.jsp");

}

}

### AddRecipeServlet.java

import jakarta.servlet.RequestDispatcher;

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import jakarta.servlet.http.HttpSession;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Arrays;

import java.util.List;

@WebServlet("/AddRecipeServlet")

public class AddRecipeServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

@Override

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

String jdbcUri = getServletContext().getInitParameter("jdbcUri");

String dbUri = getServletContext().getInitParameter("dbUri");

String dbId = getServletContext().getInitParameter("dbId");

String dbPass = getServletContext().getInitParameter("dbPass");

response.setContentType("text/html");

PrintWriter pw = response.getWriter();

HttpSession session = request.getSession();

String auth = (String) session.getAttribute("userEmail");

String title = request.getParameter("title");

String[] ingredientNames = request.getParameterValues("ingredientName[]");

String[] ingredientQuantities = request.getParameterValues("ingredientQuantity[]");

String steps = request.getParameter("steps");

List<String> ingredientNamesList = Arrays.asList(ingredientNames);

List<String> ingredientQuantitiesList = Arrays.asList(ingredientQuantities);

String vegParam = request.getParameter("veg");

boolean isVeg = Boolean.parseBoolean(vegParam);

try {

Class.forName(jdbcUri);

try (Connection con = DriverManager.getConnection(dbUri, dbId, dbPass)) {

PreparedStatement ps1;

PreparedStatement ps2;

ps1 = con.prepareStatement("INSERT INTO recipe(auth, title, steps, isVeg) VALUES(?,?,?,?)", new String[]{"id"});

ps1.setString(1, auth);

ps1.setString(2, title);

ps1.setString(3, steps);

ps1.setBoolean(4, isVeg);

int affectedRows = ps1.executeUpdate();

if (affectedRows == 0) {

throw new SQLException("Creating recipe failed, no rows affected.");

}

try (ResultSet generatedKeys = ps1.getGeneratedKeys()) {

if (generatedKeys.next()) {

ps2 = con.prepareStatement("INSERT INTO ingredients(recipeId, name, quantity) VALUES(?,?,?)");

int recipeId = generatedKeys.getInt(1);

// Batching for ingredient insertions

for (int i = 0; i < ingredientNamesList.size(); i++) {

ps2.setInt(1, recipeId);

ps2.setString(2, ingredientNamesList.get(i));

ps2.setString(3, ingredientQuantitiesList.get(i));

ps2.addBatch();

}

ps2.executeBatch();

response.sendRedirect("HomeServlet");

} else {

throw new SQLException("Creating recipe failed, no ID obtained.");

}

}

}

} catch (SQLException | ClassNotFoundException ex) {

ex.printStackTrace();

request.setAttribute("exception", ex.getMessage());

RequestDispatcher rd = request.getRequestDispatcher("home.jsp");

rd.include(request, response);

}

}

}

### HomeServlet.java

import jakarta.servlet.RequestDispatcher;

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import jakarta.servlet.http.HttpSession;

import java.io.IOException;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

import myClass.Ingredient;

import myClass.Recipe;

@WebServlet("/HomeServlet")

public class HomeServlet extends HttpServlet {

@Override

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

String jdbcUri = getServletContext().getInitParameter("jdbcUri");

String dbUri = getServletContext().getInitParameter("dbUri");

String dbId = getServletContext().getInitParameter("dbId");

String dbPass = getServletContext().getInitParameter("dbPass");

response.setContentType("text/html");

HttpSession session = request.getSession();

List<Recipe> recipeList = new ArrayList<>();

try {

Class.forName(jdbcUri);

try (Connection con = DriverManager.getConnection(dbUri, dbId, dbPass)) {

PreparedStatement ps;

ps = con.prepareStatement("SELECT recipe.id, recipe.auth, recipe.title, recipe.steps, recipe.isVeg, ingredients.name AS ingNames, ingredients.quantity AS ingQuantity FROM recipe LEFT JOIN ingredients ON recipe.id = ingredients.recipeId GROUP BY recipe.id, recipe.auth, recipe.title, recipe.steps, recipe.isVeg, ingredients.name, ingredients.quantity");

try (ResultSet rs = ps.executeQuery()) {

int currentRecipeId = -1; // Initialize with an invalid ID

Recipe recipe = null;

while (rs.next()) {

int recipeId = rs.getInt("id");

if (recipeId != currentRecipeId) {

// If the recipe ID changes, create a new Recipe object

if (recipe != null) {

// Add the previously created recipe to the list

recipeList.add(recipe);

}

// Create a new Recipe object

String auth = rs.getString("auth");

String title = rs.getString("title");

String steps = rs.getString("steps");

boolean isVeg = rs.getBoolean("isVeg");

recipe = new Recipe(recipeId, auth, title, new ArrayList<>(), steps, isVeg);

currentRecipeId = recipeId;

}

// Retrieve ingredient details and add them to the recipe

String ingredientName = rs.getString("ingNames");

String ingredientQuantity = rs.getString("ingQuantity");

if (ingredientName != null && ingredientQuantity != null) {

Ingredient ingredient = new Ingredient(ingredientName, ingredientQuantity);

recipe.getIngredients().add(ingredient);

}

}

// Add the last recipe to the list (if any)

if (recipe != null) {

recipeList.add(recipe);

}

}

session.setAttribute("recipeList", recipeList);

RequestDispatcher rd = request.getRequestDispatcher("home.jsp");

rd.forward(request, response);

}

} catch (ClassNotFoundException | SQLException ex) {

Logger.getLogger(HomeServlet.class.getName()).log(Level.SEVERE, null, ex);

request.setAttribute("exception", ex.getMessage());

RequestDispatcher rd = request.getRequestDispatcher("home.jsp");

rd.include(request, response);

}

}

}

### SearchServlet.java

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import jakarta.servlet.http.HttpSession;

import java.io.IOException;

import java.util.List;

import myClass.Recipe;

@WebServlet("/SearchServlet")

public class SearchServlet extends HttpServlet {

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

String query = request.getParameter("query");

String searchResults = "";

HttpSession session = request.getSession();

List<Recipe> recipeList = (List<Recipe>) session.getAttribute("recipeList");

if (recipeList != null && !recipeList.isEmpty()) {

StringBuilder resultBuilder = new StringBuilder();

boolean found = false;

for (Recipe r : recipeList) {

if (r.getTitle().toLowerCase().contains(query.toLowerCase())) {

resultBuilder.append("<p>").append(r.getTitle()).append("</p>");

found = true;

}

}

if (found) {

searchResults = "<html><body>" + resultBuilder.toString() + "</body></html>";

} else {

searchResults = "<p>No Recipes Found</p>";

}

} else {

searchResults = "<p>No recipe added to be searched</p>";

}

response.setContentType("text/html");

response.getWriter().write(searchResults);

}

}

### web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app version="6.0" xmlns="https://jakarta.ee/xml/ns/jakartaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee https://jakarta.ee/xml/ns/jakartaee/web-app\_6\_0.xsd">

<context-param>

<param-name>jdbcUri</param-name>

<param-value>com.mysql.cj.jdbc.Driver</param-value>

</context-param>

<context-param>

<param-name>dbUri</param-name>

<param-value>jdbc:mysql://localhost:3306/mysql</param-value>

</context-param>

<context-param>

<param-name>dbId</param-name>

<param-value>root</param-value>

</context-param>

<context-param>

<param-name>dbPass</param-name>

<param-value/>

</context-param>

<servlet>

<servlet-name>SignupServlet</servlet-name>

<servlet-class>SignupServlet</servlet-class>

</servlet>

<servlet>

<servlet-name>LoginServlet</servlet-name>

<servlet-class>LoginServlet</servlet-class>

</servlet>

<servlet>

<servlet-name>HomeServlet</servlet-name>

<servlet-class>HomeServlet</servlet-class>

</servlet>

<servlet>

<servlet-name>AddRecipeServlet</servlet-name>

<servlet-class>AddRecipeServlet</servlet-class>

</servlet>

<servlet>

<servlet-name>LogoutServlet</servlet-name>

<servlet-class>LogoutServlet</servlet-class>

</servlet>

<servlet>

<servlet-name>SessionCheck</servlet-name>

<servlet-class>SessionCheck</servlet-class>

</servlet>

<servlet>

<servlet-name>SearchServlet</servlet-name>

<servlet-class>SearchServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>SignupServlet</servlet-name>

<url-pattern>/SignupServlet</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>LoginServlet</servlet-name>

<url-pattern>/LoginServlet</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>HomeServlet</servlet-name>

<url-pattern>/HomeServlet</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>AddRecipeServlet</servlet-name>

<url-pattern>/AddRecipeServlet</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>LogoutServlet</servlet-name>

<url-pattern>/LogoutServlet</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>SessionCheck</servlet-name>

<url-pattern>/SessionCheck</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>SearchServlet</servlet-name>

<url-pattern>/SearchServlet</url-pattern>

</servlet-mapping>

<welcome-file-list>

<welcome-file>SessionCheck</welcome-file>

</welcome-file-list>

</web-app>

## Screenshots:

## 

## 

## 