Name : Mohil Mokaria

Class : 6CE

Division : E

Subject : Java Experiment 5

# Q – Create a web application where users can securely store and manage personal notes. Users need to log in to access their notes, and the application should manage user sessions.

## **User Registration:**

## Allow users to register with a unique username and password.

## Store user information securely.

## **User Login:**

## Provide a login page for users to enter their credentials.

## Authenticate users against stored credentials.

## **Session Management:**

## Create a session upon successful login.

## Use the session to maintain user authentication across requests.

## **Note Management:**

## After login, users can create, view, edit, and delete their personal notes.

## Notes should be associated with the logged-in user.

## **Logout:**

## Allow users to log out, ending their session.

### Note.java

package myClass;

import java.sql.Timestamp;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Note {

// Instance variables for the post title, body, and timestamp

private int id;

private String title;

private String body;

private Timestamp note\_time;

// Default constructor

public Note() {

}

// Parameterized constructor to initialize the Note object with title, body, and timestamp

public Note(int id, String title, String body, Timestamp note\_time) {

this.id = id;

this.title = title;

this.body = body;

this.note\_time = note\_time;

}

// Method to get the formatted message time in "HH:mm | dd/MM/yyyy" format

public String getFormattedMsgTime() {

if (note\_time != null) {

SimpleDateFormat dateFormat = new SimpleDateFormat("HH:mm | dd/MM/yyyy");

return dateFormat.format(new Date(note\_time.getTime()));

} else {

return null; // Return null if the timestamp is not set

}

}

// Setter method for the post id

public void setId(int id) {

this.id = id;

}

// Setter method for the post title

public void setTitle(String title) {

this.title = title;

}

// Setter method for the post body

public void setBody(String body) {

this.body = body;

}

// Setter method for the post timestamp

public void setTime(Timestamp note\_time) {

this.note\_time = note\_time;

}

// Getter method for the post id

public int getId() {

return this.id;

}

// Getter method for the post title

public String getTitle() {

return this.title;

}

// Getter method for the post body

public String getBody() {

return this.body;

}

// Getter method for the post timestamp

public Timestamp getTime() {

return this.note\_time;

}

}

# index.jsx :

<%@ 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.jsx :

<%@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>

# home.jsx :

<%@page import="java.sql.Timestamp"%>

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

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

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

<!DOCTYPE html>

<head>

<title>Simple Chat App</title>

<!-- Google Icon CDN -->

<link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outlined:opsz,wght,FILL,GRAD@24,400,0,0" />

<!-- 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-between">

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

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

</span>

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

<button class="btn btn-outline-danger">Logout</button>

</form>

</div>

<%

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

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

%>

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

<%

}

%>

<br>

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

<section id="NewForm">

<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="NewNote" 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>New Note</h2>

</div>

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

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

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

</div>

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

<label class="form-label" for="body">Note:</label>

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

</div>

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

</form>

</div>

</div>

</div>

</div>

</div>

</section>

<section id="UpdateForm">

<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="EditNote" method="post">

<div class="mb-4">

<h2>Edit Note</h2>

</div>

<input id="id" style="display: none" type="text" name="id" class="form-control"/>

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

<label for="newTitle" class="form-label">Updated Title:</label>

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

</div>

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

<label class="form-label" for="newBody">Updated Note:</label>

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

</div>

<center><button type="submit" class="btn btn-primary btn-block mb-2 sendbtn">Update</button></center>

<center><button onclick="cancelEdit()" class="btn btn-outline-danger btn-block mb-2 sendbtn">Cancel</button></center>

</form>

</div>

</div>

</div>

</div>

</div>

</section>

<section>

<div class="row align-items-center mt-4 mb-4">

<div class="col-md-9 col-sm-8 mx-auto">

<div class="card-body d-flex flex-column">

<%

List<Note> noteList = (List<Note>) request.getAttribute("noteList");

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

for (Note n : noteList) {

%>

<div class="card mb-2">

<div class="card-header d-flex align-items-center justify-content-between">

<%= n.getTitle()%>

<div class="d-flex align-items-center w-50 justify-content-end">

<a onclick="showEditForm('<%= n.getId() %>', '<%= n.getTitle() %>', '<%= n.getBody() %>')" class="btn btn-primary delete">

<span class="material-symbols-outlined">

contract\_edit

</span>

</a>

<a href="DeleteNote?id=<%= n.getId() %>" class="btn btn-danger delete">

<span class="material-symbols-outlined">

delete\_sweep

</span>

</a>

</div>

</div>

<div class="card-body">

<blockquote class="blockquote mb-0">

<p><%= n.getBody()%></p>

<footer class="blockquote-footer text-muted"><small><%= n.getFormattedMsgTime()%></small></footer>

</blockquote>

</div>

</div>

<%

}

} else {

%>

<div class="card messages\_card">

<div class="card-body">

<div class="row">

<h5 class="card-title col">No Notes are Added Yet!</h5>

</div>

<p class="card-text">Try Creating some Notes!</p>

</div>

</div>

<%

}

%>

</div>

</div>

</div>

</section>

<script>

var newFormSection = document.getElementById('NewForm');

var formSection = document.getElementById('UpdateForm');

function showEditForm(id, title, body) {

newFormSection.style.display = 'none'

formSection.style.display = 'block';

// Scroll to the top of the page

window.scrollTo(0, 0);

// Set default values for the modal input fields

document.getElementById('id').value = id;

document.getElementById('newTitle').value = title;

document.getElementById('newBody').value = body;

// Set focus on the new Title input field

document.getElementById('newTitle').focus();

// You may also need to set the ID as a hidden input value if you need it for your update operation

document.getElementById('noteId').value = id;

}

function cancelEdit() {

newFormSection.style.display = 'block'

formSection.style.display = 'none';

}

</script>

<% } else { %>

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

<a href="login.jsp">Login</a>

<% }%>

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

</body>

### 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");

}

}

### NewNote.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.sql.Timestamp;

import java.util.ArrayList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

import myClass.Note;

@WebServlet("/NewNote")

public class NewNote extends HttpServlet {

private static final long serialVersionUID = 1L;

@Override

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

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

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

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

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

HttpSession session = request.getSession();

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

List<Note> noteList = new ArrayList<>();

try {

Class.forName(jdbcUri);

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

String sql = "SELECT id, title, body, note\_time FROM notes WHERE who = ? ORDER BY note\_time DESC";

try (PreparedStatement statement = connection.prepareStatement(sql)) {

statement.setString(1, who);

try (ResultSet resultSet = statement.executeQuery()) {

while (resultSet.next()) {

Note n = new Note();

n.setId(resultSet.getInt("id"));

n.setTitle(resultSet.getString("title"));

n.setBody(resultSet.getString("body"));

n.setTime(resultSet.getTimestamp("note\_time"));

noteList.add(n);

}

}

}

request.setAttribute("noteList", noteList);

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

rd.forward(request, response);

}

} catch (ClassNotFoundException | SQLException ex) {

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

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

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

rd.forward(request, response);

}

}

@Override

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

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

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

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

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

response.setContentType("text/html");

HttpSession session = request.getSession();

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

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

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

Timestamp noteTime = new Timestamp(System.currentTimeMillis());

try {

Class.forName(jdbcUri);

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

try (PreparedStatement ps = con.prepareStatement("INSERT INTO notes(who, title, body, note\_time) values(?,?,?,?)")) {

ps.setString(1, who);

ps.setString(2, title);

ps.setString(3, body);

ps.setTimestamp(4, noteTime);

int i = ps.executeUpdate();

response.sendRedirect("NewNote");

}

}

} catch (SQLException | ClassNotFoundException ex) {

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

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

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

rd.forward(request, response);

}

}

}

### DeleteNote.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.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

@WebServlet("/DeleteNote")

public class DeleteNote extends HttpServlet {

private static final long serialVersionUID = 1L;

@Override

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

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

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

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

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

String idParameter = request.getParameter("id");

if (idParameter != null) {

try {

int noteId = Integer.parseInt(idParameter);

Class.forName(jdbcUri);

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

String sql = "DELETE FROM notes WHERE id=?";

try (PreparedStatement statement = connection.prepareStatement(sql)) {

statement.setInt(1, noteId);

int rowsDeleted = statement.executeUpdate();

response.sendRedirect("NewNote");

// if (rowsDeleted > 0) {

// // Note successfully deleted

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

// } else {

// // Note with the specified ID not found

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

// }

}

}

} catch (NumberFormatException | ClassNotFoundException | SQLException ex) {

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

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

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

rd.forward(request, response);

}

} else {

// Handle the case where idParameter is null

response.sendRedirect("InvalidIdPage.jsp");

}

}

}

### EditNote.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.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

import java.sql.Timestamp;

import java.util.logging.Level;

import java.util.logging.Logger;

@WebServlet("/EditNote")

public class EditNote extends HttpServlet {

private static final long serialVersionUID = 1L;

@Override

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

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

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

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

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

response.setContentType("text/html");

String idParameter = request.getParameter("id");

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

String body = request.getParameter("newBody");

Timestamp noteTime = new Timestamp(System.currentTimeMillis());

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

try {

int id = Integer.parseInt(idParameter);

Class.forName(jdbcUri);

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

try (PreparedStatement ps = con.prepareStatement("UPDATE notes SET title=?, body=?, note\_time=? WHERE id=?")) {

ps.setString(1, title);

ps.setString(2, body);

ps.setTimestamp(3, noteTime);

ps.setInt(4, id);

int i = ps.executeUpdate();

response.sendRedirect("NewNote");

}

}

} catch (SQLException | ClassNotFoundException ex) {

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

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

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

rd.forward(request, response);

}

} else {

response.sendRedirect("NewNote");

}

}

}

### 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>loginUser</servlet-name>

<servlet-class>loginUser</servlet-class>

</servlet>

<servlet>

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

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

</servlet>

<servlet>

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

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

</servlet>

<servlet>

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

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

</servlet>

<servlet>

<servlet-name>NewNote</servlet-name>

<servlet-class>NewNote</servlet-class>

</servlet>

<servlet>

<servlet-name>EditNote</servlet-name>

<servlet-class>EditNote</servlet-class>

</servlet>

<servlet>

<servlet-name>DeleteNote</servlet-name>

<servlet-class>DeleteNote</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>loginUser</servlet-name>

<url-pattern>/loginUser</url-pattern>

</servlet-mapping>

<servlet-mapping>

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

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

</servlet-mapping>

<servlet-mapping>

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

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

</servlet-mapping>

<servlet-mapping>

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

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

</servlet-mapping>

<servlet-mapping>

<servlet-name>NewNote</servlet-name>

<url-pattern>/NewNote</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>EditNote</servlet-name>

<url-pattern>/EditNote</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>DeleteNote</servlet-name>

<url-pattern>/DeleteNote</url-pattern>

</servlet-mapping>

</web-app>

## Screenshots:

## 

## 

## 

## 

## 

## 