Experiment No.6

Date:11.03.25

Aim: Implementation of real world problem based on servlet concept

CO Mapping – CO 2

Objective:

1. To develop the servlet based application.
2. To use servlets as the primary component to process client requests and generate dynamic responses.
3. To handle HTTP request and response objects (e.g., GET, POST, PUT, DELETE).
4. To utilize RequestDispatcher for forwarding or including resources (e.g., JSP pages).
5. To use HTTP sessions to maintain state for users interacting with the application.

Code:

**pom.xml**

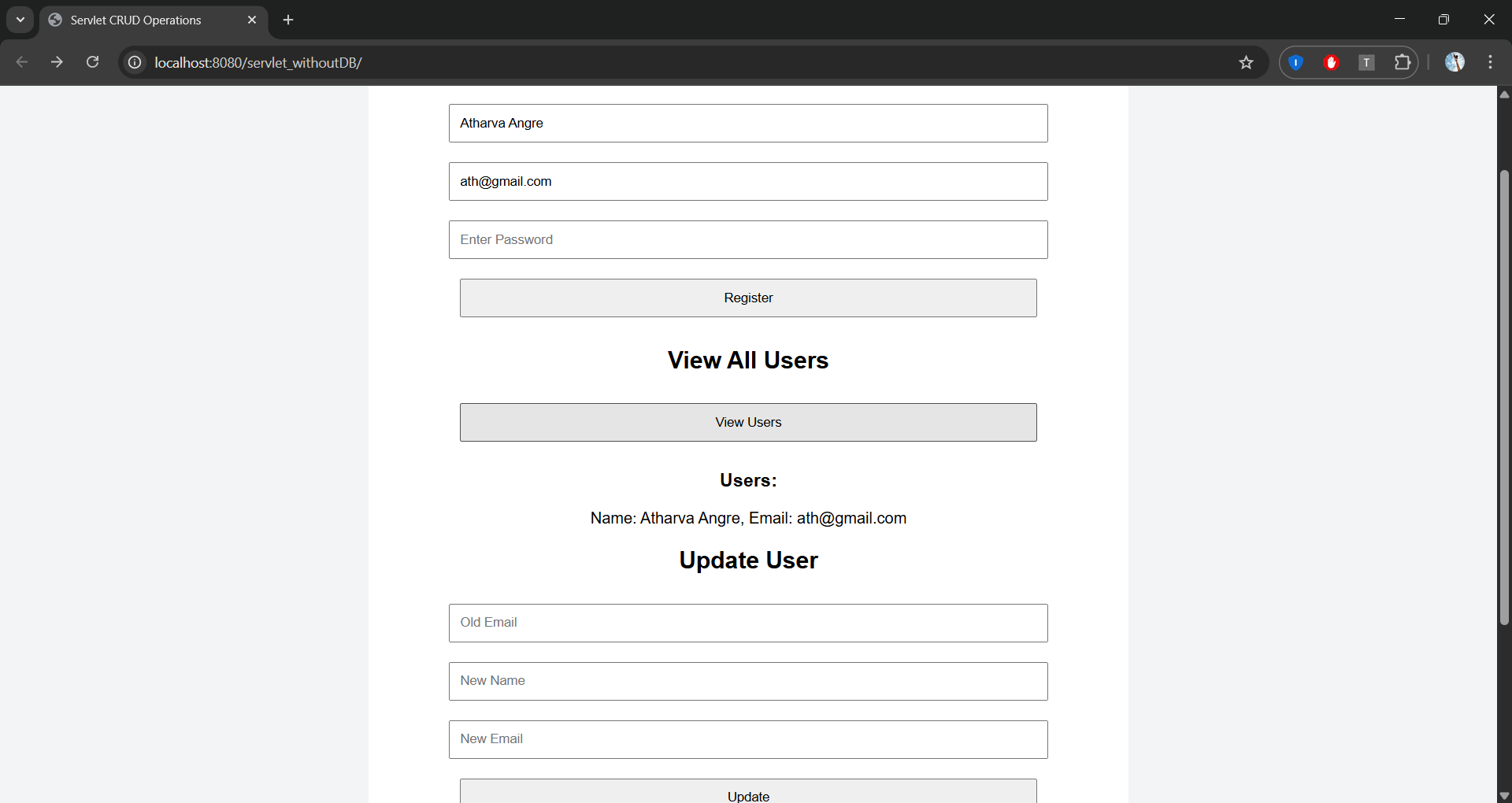
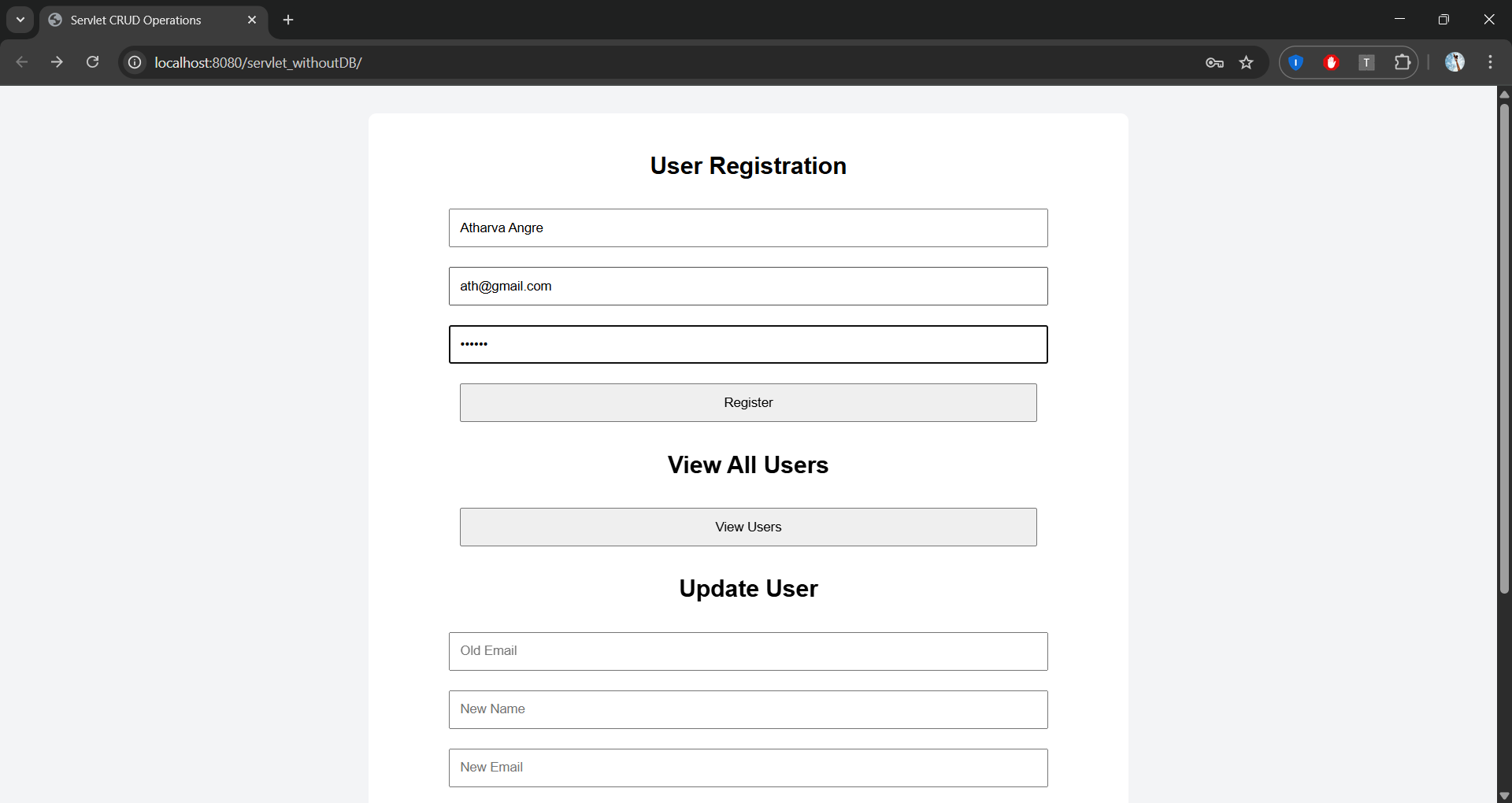
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>org.example</groupId>  
 <artifactId>servlet\_withoutDB</artifactId>  
 <packaging>war</packaging>  
 <version>1.0-SNAPSHOT</version>  
 <name>servlet\_withoutDB Maven Webapp</name>  
 <url>http://maven.apache.org</url>  
 <dependencies>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>3.8.1</version>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>javax.servlet-api</artifactId>  
 <version>4.0.1</version>  
 <scope>provided</scope>  
 </dependency>  
 </dependencies>  
 <build>  
 <finalName>servlet\_withoutDB</finalName>  
 </build>  
</project>

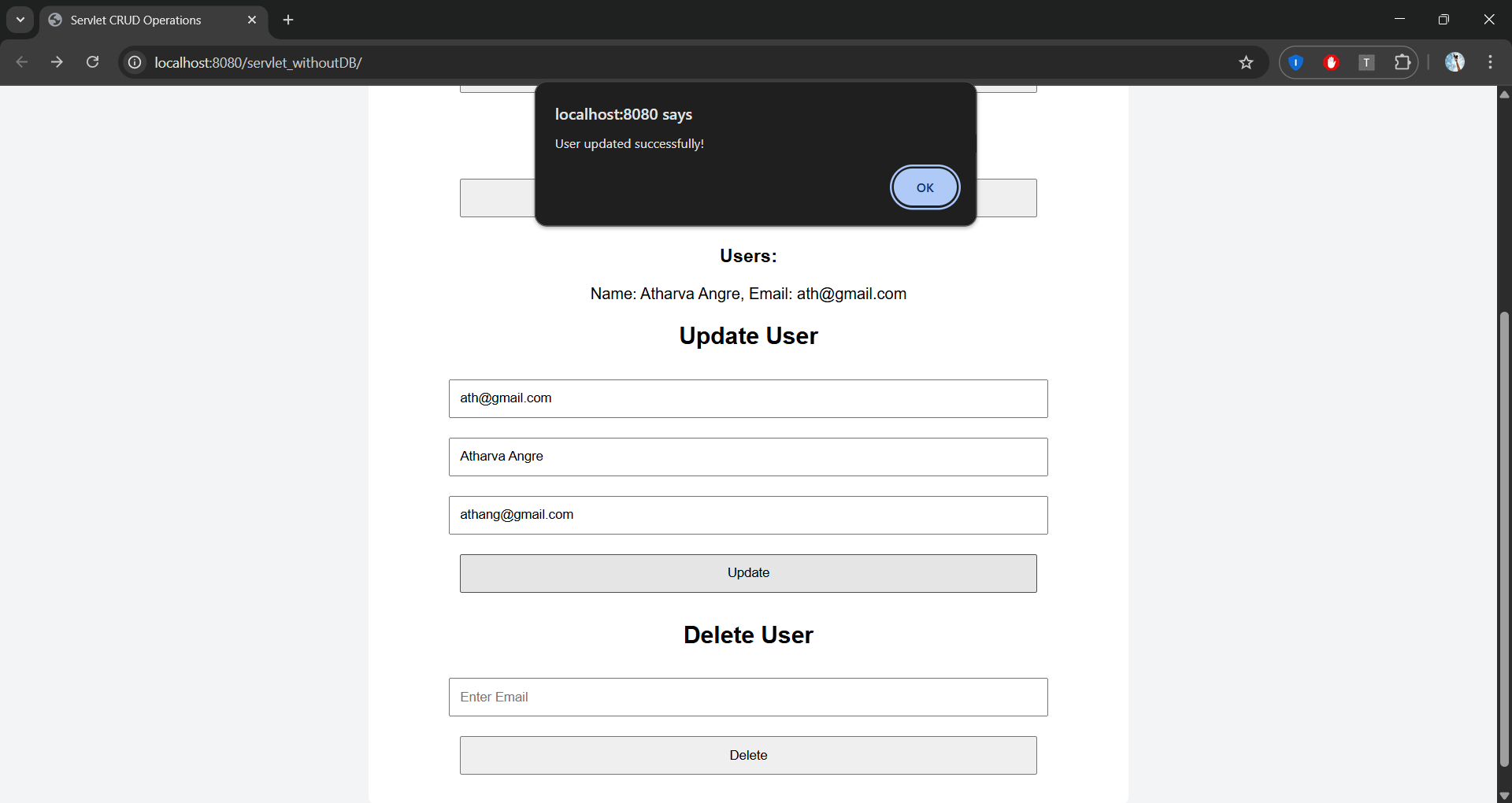
**RegisterServlet.java**  
package com.example;  
  
import java.io.BufferedReader;  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.util.ArrayList;  
import java.util.Iterator;  
import javax.servlet.ServletException;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
public class RegisterServlet extends HttpServlet {  
  
 private static ArrayList<User> *users* = new ArrayList<>();  
  
 protected void doPost(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
 response.setContentType("application/json");  
 PrintWriter out = response.getWriter();  
  
 String name = request.getParameter("name");  
 String email = request.getParameter("email");  
 String password = request.getParameter("password");  
  
 if (name == null || email == null || password == null) {  
 out.println("{\"message\": \"Invalid input!\"}");  
 return;  
 }  
  
 *users*.add(new User(name, email, password));  
 out.println("{\"message\": \"User registered successfully!\"}");  
 }  
  
 protected void doGet(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
 response.setContentType("application/json");  
 PrintWriter out = response.getWriter();  
  
 out.print("[");  
 for (int i = 0; i < *users*.size(); i++) {  
 User user = *users*.get(i);  
 out.print("{\"name\":\"" + user.getName() + "\", \"email\":\"" + user.getEmail() + "\"}");  
 if (i < *users*.size() - 1) {  
 out.print(",");  
 }  
 }  
 out.print("]");  
 }  
  
 protected void doPut(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
 response.setContentType("application/json");  
 PrintWriter out = response.getWriter();  
  
 String requestData = readRequestBody(request);  
 if (requestData.isEmpty()) {  
 out.println("{\"message\": \"Invalid input!\"}");  
 return;  
 }  
  
 String oldEmail = extractJsonValue(requestData, "oldEmail");  
 String newName = extractJsonValue(requestData, "name");  
 String newEmail = extractJsonValue(requestData, "email");  
  
 boolean updated = false;  
 for (User user : *users*) {  
 if (user.getEmail().equals(oldEmail)) {  
 user.setName(newName);  
 user.setEmail(newEmail);  
 updated = true;  
 break;  
 }  
 }  
  
 out.println(updated ? "{\"message\": \"User updated successfully!\"}" :  
 "{\"message\": \"User not found!\"}");  
 }  
  
 protected void doDelete(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
 response.setContentType("application/json");  
 PrintWriter out = response.getWriter();  
  
 String requestData = readRequestBody(request);  
 if (requestData.isEmpty()) {  
 out.println("{\"message\": \"Invalid input!\"}");  
 return;  
 }  
  
 String email = extractJsonValue(requestData, "email");  
  
 Iterator<User> iterator = *users*.iterator();  
 boolean deleted = false;  
 while (iterator.hasNext()) {  
 User user = iterator.next();  
 if (user.getEmail().equals(email)) {  
 iterator.remove();  
 deleted = true;  
 break;  
 }  
 }  
  
 out.println(deleted ? "{\"message\": \"User deleted successfully!\"}" :  
 "{\"message\": \"User not found!\"}");  
 }  
  
 private String readRequestBody(HttpServletRequest request) throws IOException {  
 BufferedReader reader = request.getReader();  
 StringBuilder sb = new StringBuilder();  
 String line;  
 while ((line = reader.readLine()) != null) {  
 sb.append(line);  
 }  
 return sb.toString().trim();  
 }  
  
 private String extractJsonValue(String json, String key) {  
 String searchKey = "\"" + key + "\":\"";  
 int startIndex = json.indexOf(searchKey);  
 if (startIndex == -1) {  
 return "";  
 }  
 startIndex += searchKey.length();  
 int endIndex = json.indexOf("\"", startIndex);  
 return endIndex == -1 ? "" : json.substring(startIndex, endIndex);  
 }  
}

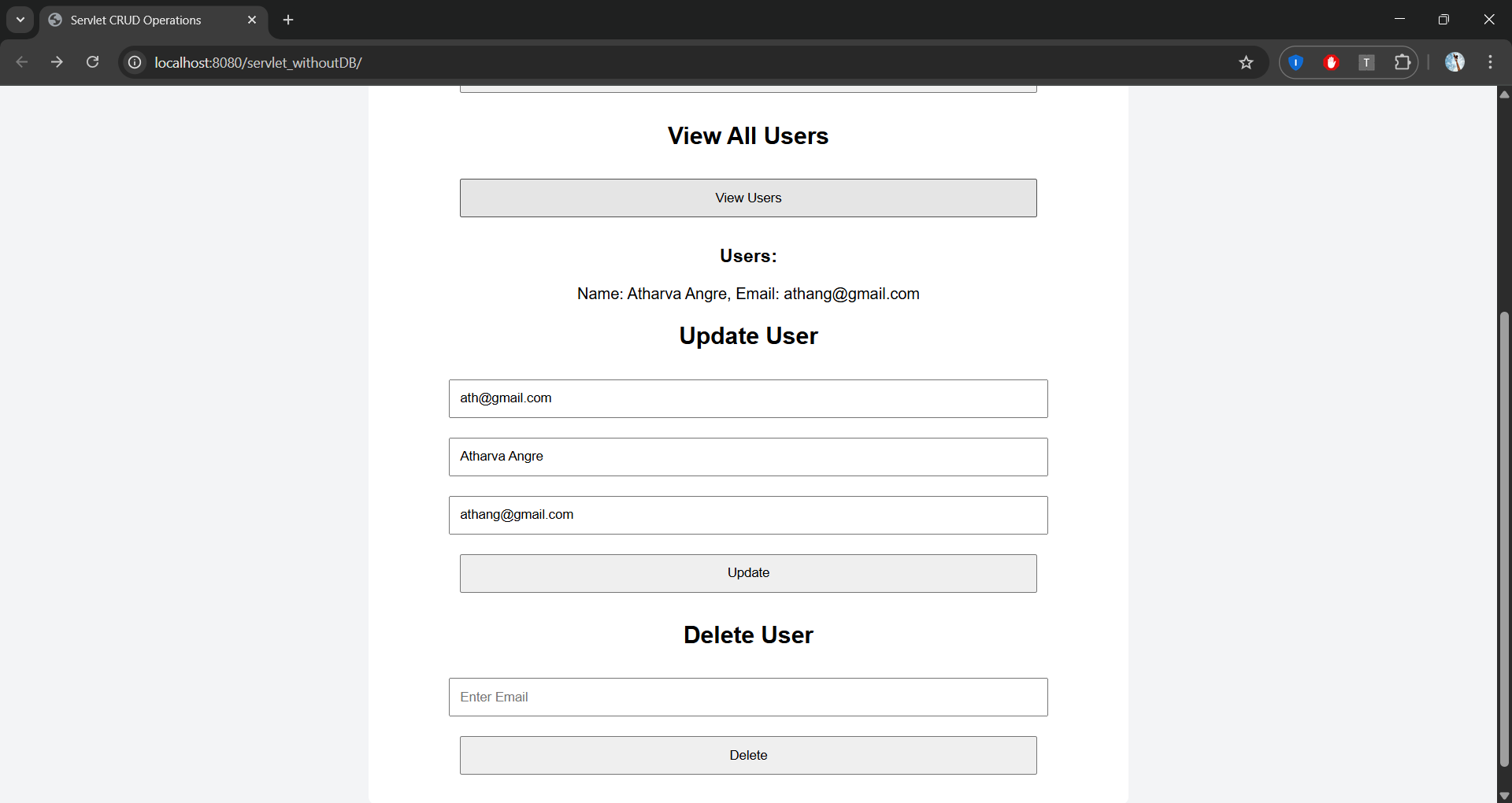
**web.xml**<!DOCTYPE web-app PUBLIC  
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"  
 "http://java.sun.com/dtd/web-app\_2\_3.dtd" >  
  
<web-app>  
 <display-name>Archetype Created Web Application</display-name>  
 <servlet>  
 <servlet-name>RegisterServlet</servlet-name>  
 <servlet-class>com.example.RegisterServlet</servlet-class>  
 </servlet>  
  
 <servlet-mapping>  
 <servlet-name>RegisterServlet</servlet-name>  
 <url-pattern>/register</url-pattern>  
 </servlet-mapping>  
  
 <welcome-file-list>  
 <welcome-file>index.jsp</welcome-file>  
 </welcome-file-list>  
  
</web-app>

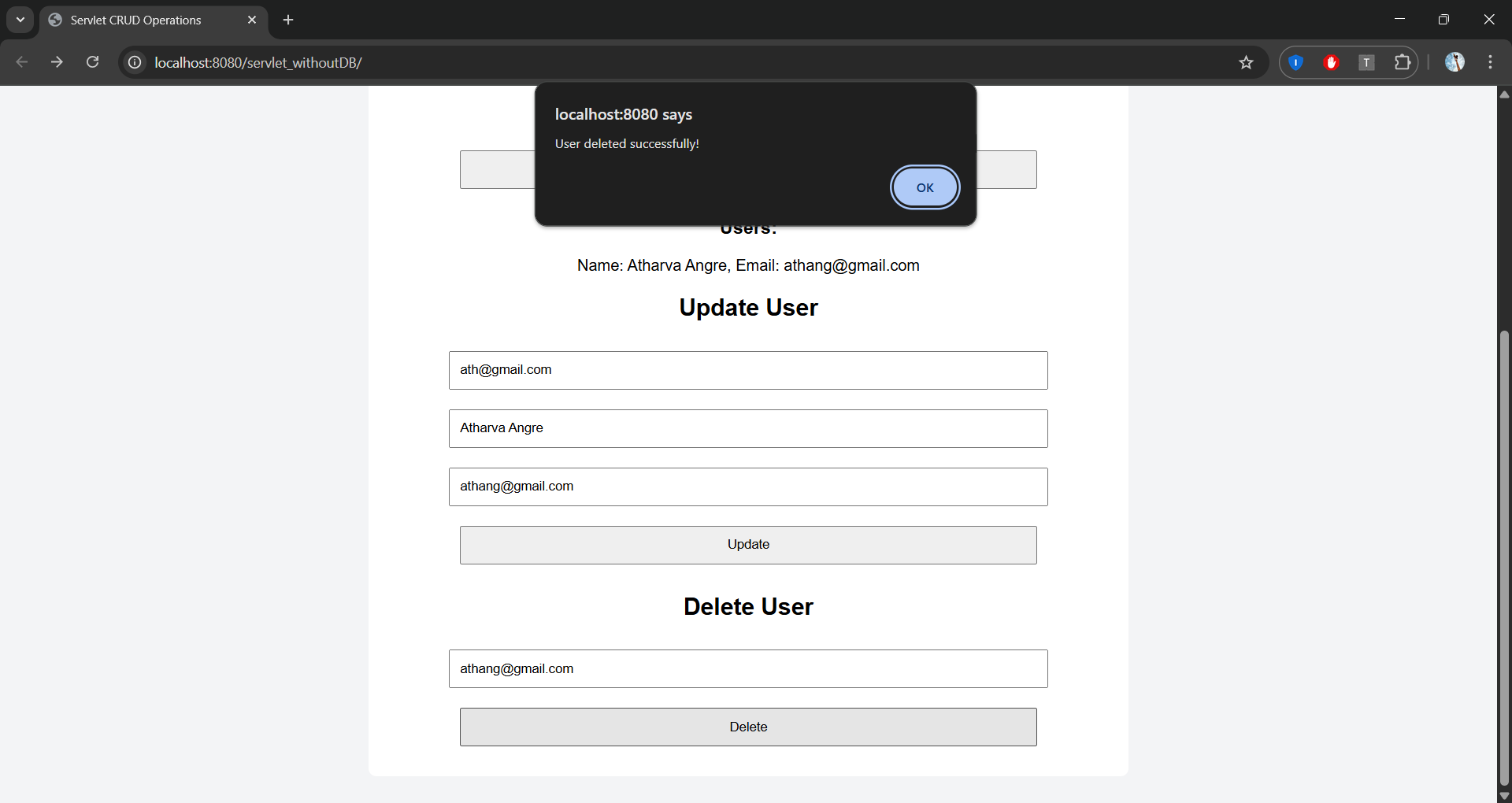
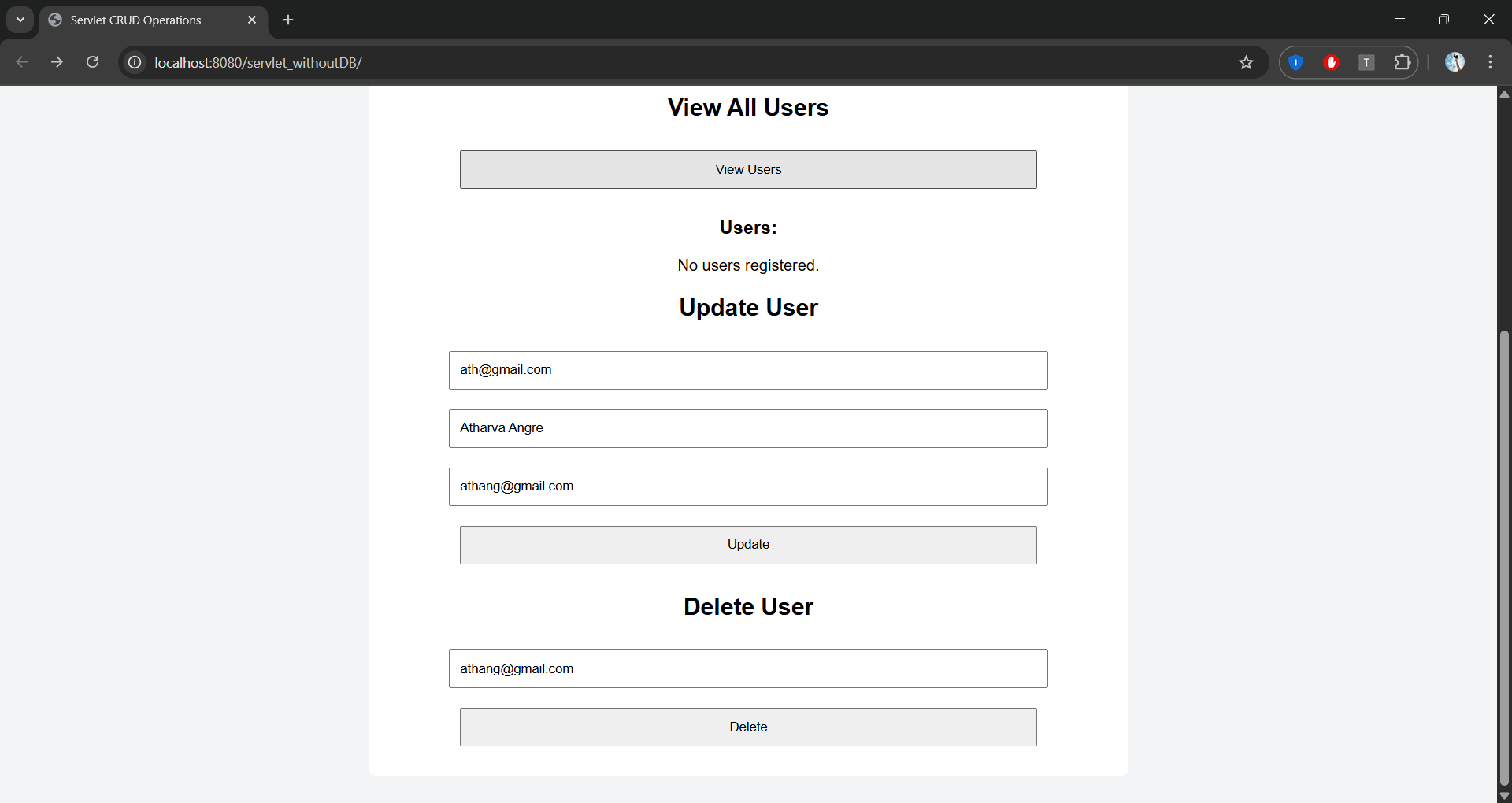
**index.jsp**  
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>  
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Servlet CRUD Operations</title>  
 <style>  
 body { font-family: Arial, sans-serif; background: #f3f4f6; text-align: center; padding: 20px; }  
 .container { width: 50%; margin: auto; padding: 20px; background: white; border-radius: 8px; }  
 input, button { width: 80%; padding: 10px; margin: 10px; }  
 </style>  
</head>  
<body>  
 <div class="container">  
 <h2>User Registration</h2>  
 <form action="register" method="post">  
 <input type="text" name="name" placeholder="Enter Name" required><br>  
 <input type="email" name="email" placeholder="Enter Email" required><br>  
 <input type="password" name="password" placeholder="Enter Password" required><br>  
 <button type="submit">Register</button>  
 </form>  
  
 <h2>View All Users</h2>  
 <button onclick="viewUsers()">View Users</button>  
 <div id="usersList"></div>  
  
 <h2>Update User</h2>  
 <form id="updateForm">  
 <input type="email" id="oldEmail" placeholder="Old Email" required><br>  
 <input type="text" id="newName" placeholder="New Name" required><br>  
 <input type="email" id="newEmail" placeholder="New Email" required><br>  
 <button type="submit">Update</button>  
 </form>  
  
 <h2>Delete User</h2>  
 <form id="deleteForm">  
 <input type="email" id="deleteEmail" placeholder="Enter Email" required><br>  
 <button type="submit">Delete</button>  
 </form>  
 </div>  
  
 <script>  
 function viewUsers() {  
 fetch('register', { method: 'GET' })  
 .then(response => response.json())  
 .then(data => {  
 let usersList = document.getElementById('usersList');  
 usersList.innerHTML = '<h3>Users:</h3>';  
 if (data.length === 0) {  
 usersList.innerHTML += '<p>No users registered.</p>';  
 } else {  
 data.forEach(user => {  
 usersList.innerHTML += `<p>Name: ${user.name}, Email: ${user.email}</p>`;  
 });  
 }  
 });  
 }  
  
 document.getElementById('updateForm').addEventListener('submit', function(event) {  
 event.preventDefault();  
 let oldEmail = document.getElementById('oldEmail').value;  
 let newName = document.getElementById('newName').value;  
 let newEmail = document.getElementById('newEmail').value;  
  
 fetch('register', {  
 method: 'PUT',  
 headers: { 'Content-Type': 'application/json' },  
 body: JSON.stringify({ oldEmail, name: newName, email: newEmail })  
 })  
 .then(response => response.json())  
 .then(data => alert(data.message));  
 });  
  
  
 document.getElementById('deleteForm').addEventListener('submit', function(event) {  
 event.preventDefault();  
 let email = document.getElementById('deleteEmail').value;  
  
 fetch('register', {  
 method: 'DELETE',  
 headers: { 'Content-Type': 'application/json' },  
 body: JSON.stringify({ email })  
 })  
 .then(response => response.json())  
 .then(data => alert(data.message));  
 });  
  
 </script>  
</body>  
</html>

**User.java**  
package com.example;  
  
public class User {  
 private String name;  
 private String email;  
 private String password;  
  
 public User(String name, String email, String password) {  
 this.name = name;  
 this.email = email;  
 this.password = password;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getEmail() {  
 return email;  
 }  
  
 public void setEmail(String email) {  
 this.email = email;  
 }  
  
 public String getPassword() {  
 return password;  
 }  
}

Output: 





Observation:

Servlets handle client requests and responses dynamically, enabling user registration without a database. JSP simplifies web page creation by combining Java with HTML. Proper request handling ensures smooth interaction and a better user experience.