**Lab Sheet 01(Java Thread)**

Task 01

public class SimpleThread extends Thread {

@Override

public void run() {

System.out.println(Thread.currentThread().getId() + " is executing the thread.");

}

public static void main(String[] args) {

SimpleThread thread1 = new SimpleThread();

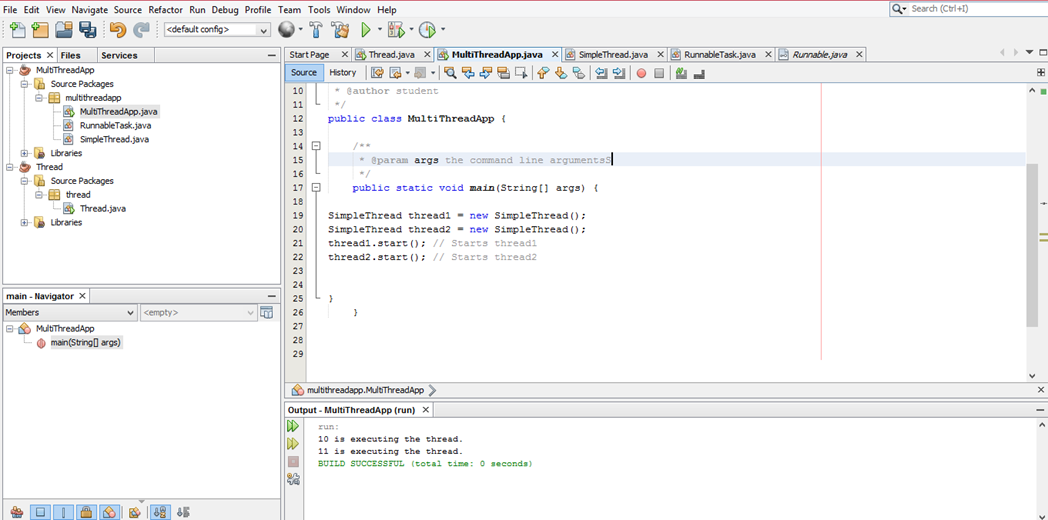
SimpleThread thread2 = new SimpleThread();

thread1.start(); // Starts thread1

thread2.start(); // Starts thread2

}

}



Task 02

public class RunnableTask implements Runnable {

@Override

public void run() {

System.out.println(Thread.currentThread().getId() + " is executingthe runnable task.");

}

}

public static void main(String[] args) {

RunnableTask task1 = new RunnableTask();

RunnableTask task2 = new RunnableTask();

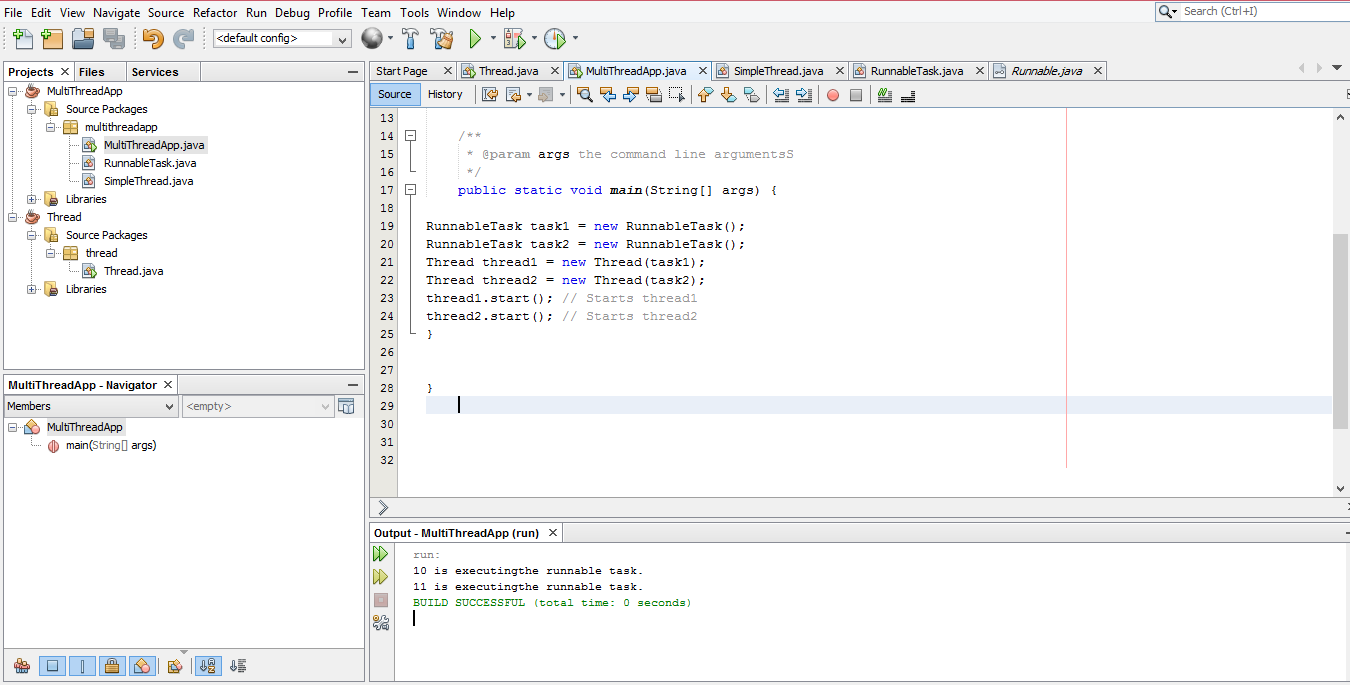
Thread thread1 = new Thread(task1);

Thread thread2 = new Thread(task2);

thread1.start(); // Starts thread1

thread2.start(); // Starts thread2

}

}

Task 03

public class counter {

private int count = 0;

// Synchronized method to ensure thread-safe access to the counter

public synchronized void increment() {

count++;

}

public int getCount() {

return count;

}

}

public class SynchronizedExample extends Thread {

private counter counter;

public SynchronizedExample(counter counter) {

this.counter = counter;

}

@Override

public void run() {

for (int i = 0; i < 1000; i++) {

counter.increment();

}

}

public static void main(String[] args) throws InterruptedException {

counter counter = new counter();

// Create and start multiple threads

Thread thread1 = new SynchronizedExample(counter);

Thread thread2 = new SynchronizedExample(counter);

thread1.start();

thread2.start();

// Wait for threads to finish

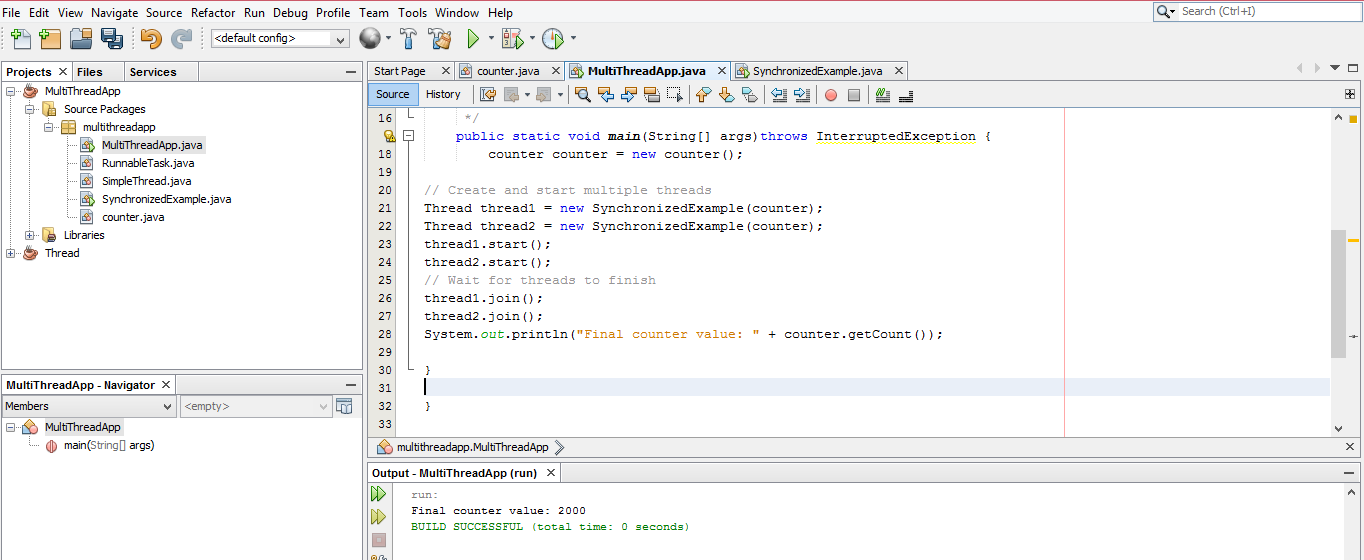
thread1.join();

thread2.join();

System.out.println("Final counter value: " + counter.getCount());

}

}



Task 04

import java.util.concurrent.ExecutorService;

import java.util.concurrent.Executors;

class Task implements Runnable {

private int taskId;

public Task(int taskId) {

this.taskId = taskId;

}

@Override

public void run() {

System.out.println("Task " + taskId + " is being processed by " +

Thread.currentThread().getName());

}

}

public class ThreadPoolExample {

public static void main(String[] args) {

// Create a thread pool with 3 threads

ExecutorService executorService = Executors.newFixedThreadPool(3);

// Submit tasks to the pool

for (int i = 1; i <= 5; i++) {

executorService.submit(new Task(i));

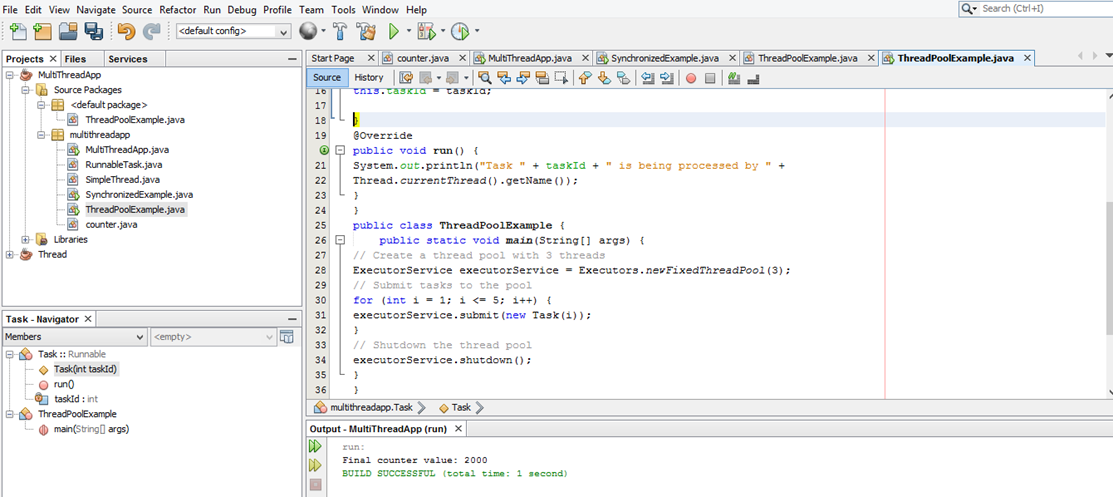
}

// Shutdown the thread pool

executorService.shutdown();

}

}



Task 05

public class ThreadLifecycleExample extends Thread {

@Override

public void run() {

System.out.println(Thread.currentThread().getName() + " - State: " +

Thread.currentThread().getState());

try {

Thread.sleep(2000); // Simulate waiting state

} catch (InterruptedException e) {

e.printStackTrace();

}

System.out.println(Thread.currentThread().getName() + " - State after

sleep: " + Thread.currentThread().getState());

}

public static void main(String[] args) {

ThreadLifecycleExample thread = new ThreadLifecycleExample();

System.out.println(thread.getName() + " - State before start: " +

thread.getState());

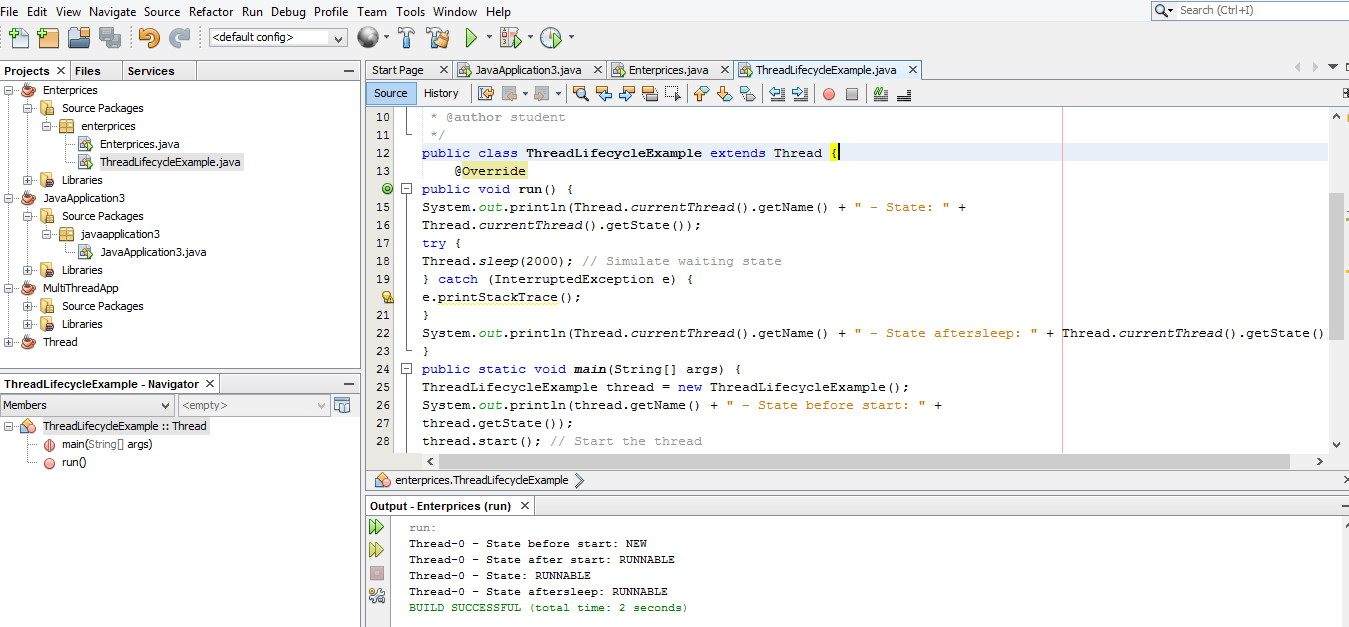
thread.start(); // Start the thread

System.out.println(thread.getName() + " - State after start: " +

thread.getState());

}

}



**Lab Sheet 02(JDBC)**

CREATE DATABASE employee\_db;

USE employee\_db;

CREATE TABLE employees (

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100),

position VARCHAR(100),

salary DECIMAL(10, 2)

);

INSERT INTO employees (name, position, salary) VALUES (” John Doe”,” Software

Engineer”, 75000);

INSERT INTO employees (name, position, salary) VALUES (“Jane Smith”,

“Manager”, 65000);

INSERT INTO employees (name, position, salary) VALUES (’’Steve Brown’’, ‘’Team

Lead’’, 85000);

package jdbcexample;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DatabaseConnection {

private static final String URL =

"jdbc:mysql://localhost:3306/employee\_db"; // Database URL

private static final String USER = "root"; // Your MySQL username

private static final String PASSWORD = ""; // Your MySQL password public static Connection getConnection() throws SQLException {

public static Connection getConnection() throws SQLException {

try {

// Load the JDBC driver

Class.forName("com.mysql.cj.jdbc.Driver");

// Return the database connection

return DriverManager.getConnection(URL, USER, PASSWORD);

} catch (ClassNotFoundException | SQLException e) {

System.out.println("Connection failed: " + e.getMessage()); throw new SQLException("Failed to establish connection.");}

}

}public class Employee {

private int id;

private String name;

private String position;

private double salary;

public Employee(int id, String name, String position, double salary) { this.id = id;

this.name = name;

this.position = position;

this.salary = salary;

}

// Getters and setters

public int getId() { return id; }

public void setId(int id) { this.id = id; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

public String getPosition() { return position; }

public void setPosition(String position) { this.position = position; } public double getSalary() { return salary; }

public void setSalary(double salary) { this.salary = salary; }

@Override

public String toString() {

return "Employee{id=" + id + ", name='" + name + "', position='" + position + "', salary=" + salary + '}';

}

}

}package jdbcexample;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

/\*\*

\*

\* @author student

\*/

public class EmployeeDAO {

// Create an employee

public static void addEmployee(String name, String position, double salary) {

String sql = "INSERT INTO employees (name, position, salary) VALUES (?, ?, ?)";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setString(1, name);

stmt.setString(2, position);

stmt.setDouble(3, salary);

int rowsAffected = stmt.executeUpdate();

System.out.println("Employee added successfully. Rows affected: " + rowsAffected);

} catch (SQLException e) {

e.printStackTrace();

}

}

// Read all employees

public static List<Employee> getAllEmployees() {

List<Employee> employees = new ArrayList<>();

String sql = "SELECT \* FROM employees";

try (Connection conn = DatabaseConnection.getConnection();

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql)) {

while (rs.next()) {

Employee employee = new Employee(rs.getInt("id"),

rs.getString("name"),

rs.getString("position"),

rs.getDouble("salary")

);

employees.add(employee);

}

} catch (SQLException e) {

e.printStackTrace();

}

return employees;

}

// Update an employee's information

public static void updateEmployee(int id, String name, String position, double salary) {

String sql = "UPDATE employees SET name = ?, position = ?, salary = ? WHERE id = ?";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setString(1, name);

stmt.setString(2, position);

stmt.setDouble(3, salary);

stmt.setInt(4, id);

int rowsAffected = stmt.executeUpdate();

System.out.println("Employee updated successfully. Rows affected: " + rowsAffected);

} catch (SQLException e) {

e.printStackTrace();

}

}

// Delete an employee

public static void deleteEmployee(int id) {

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

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setInt(1, id);

int rowsAffected = stmt.executeUpdate();

System.out.println("Employee deleted successfully. Rows affected: " + rowsAffected);

} catch (SQLException e) {

e.printStackTrace();

}}

}import java.util.List;

/\*\*

\*

\* @author student

\*/

public class JDBCExample {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

EmployeeDAO.addEmployee("Alice Cooper", "Developer", 70000);

EmployeeDAO.addEmployee("Bob Marley", "Manager", 80000);

// Update employee

EmployeeDAO.updateEmployee(1, "John Doe", "Senior Software Engineer", 90000);

// Get all employees

List<Employee> employees = EmployeeDAO.getAllEmployees();

employees.forEach(System.out::println);

// Delete employee

EmployeeDAO.deleteEmployee(2);

}}import java.util.List;

/\*\*

\*

\* @author student

\*/

public class Main {

public static void main(String[] args) {

// Add employees

EmployeeDAO.addEmployee("Alice Cooper", "Developer", 70000);

EmployeeDAO.addEmployee("Bob Marley", "Manager", 80000);

// Update employee

EmployeeDAO.updateEmployee(1, "John Doe", "Senior Software Engineer", 90000);

// Get all employees

List<Employee> employees = EmployeeDAO.getAllEmployees();

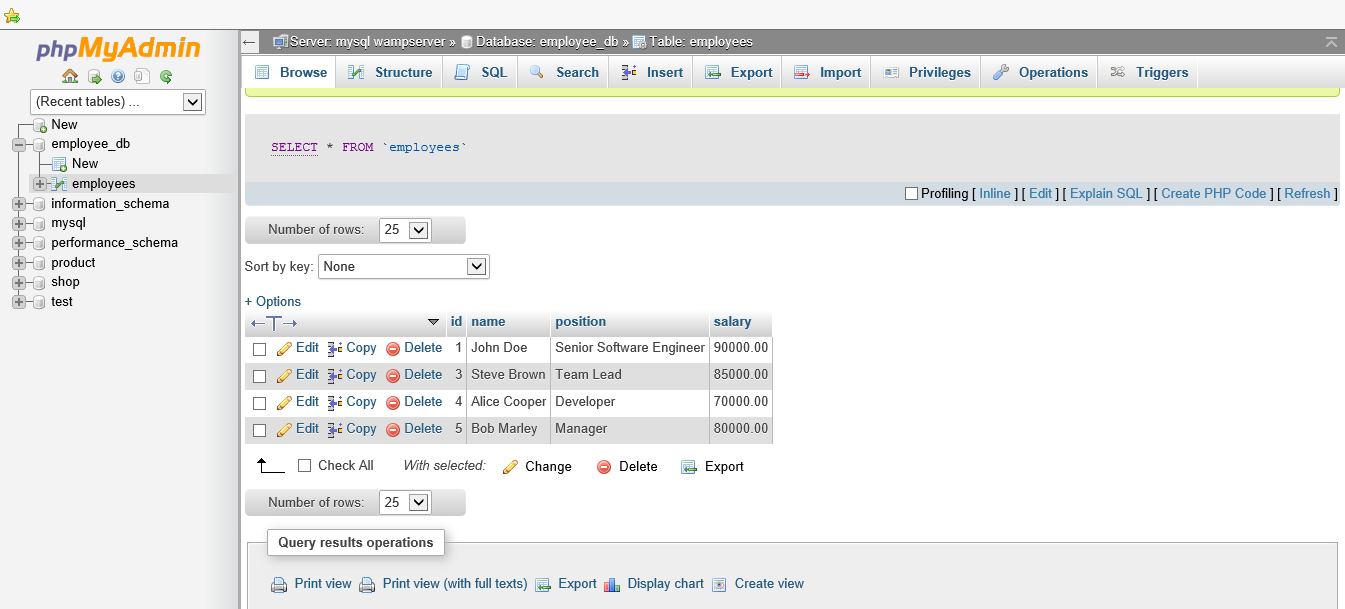
employees.forEach(System.out::println);

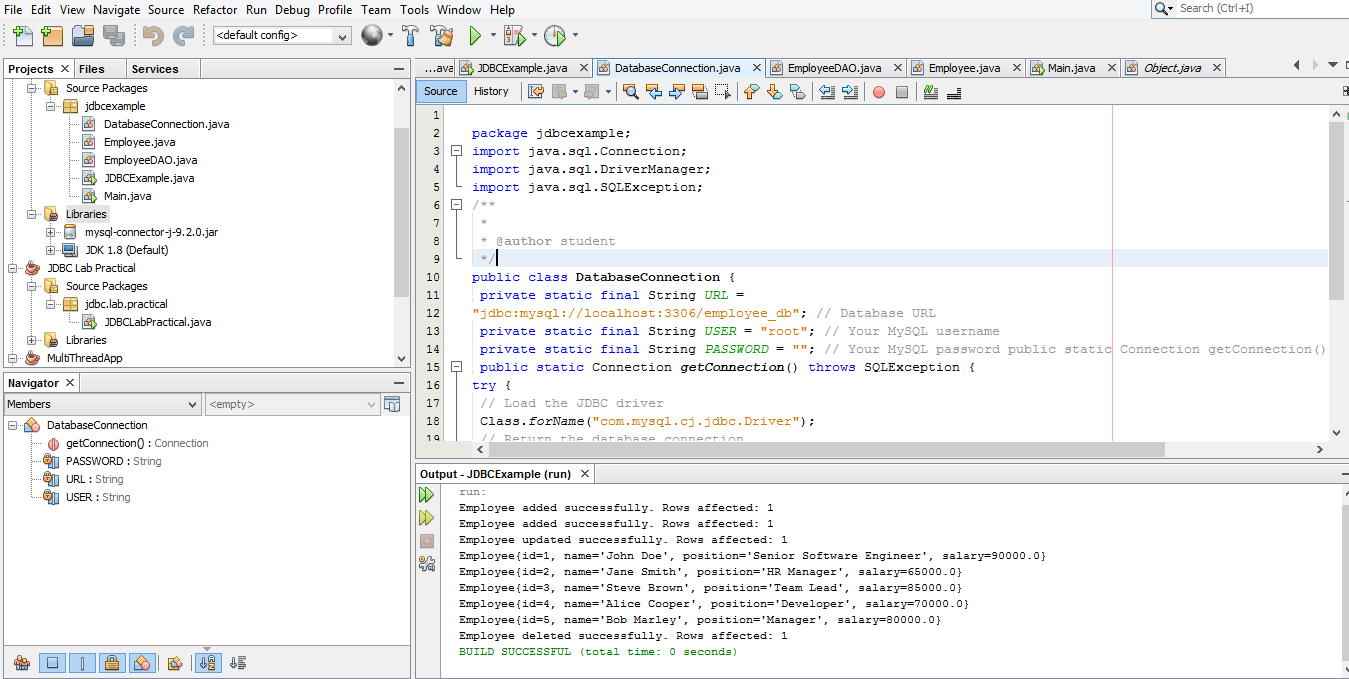
// Delete employee

EmployeeDAO.deleteEmployee(2);

}

}





**Lab Sheet 03(XML)**

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

<library>

<book>

<title>The Great Gatsby</title>

<author>F. Scott Fitzgerald</author>

<year>1925</year>

<genre>Fiction</genre>

</book>

<book>

<title>To Kill a Mockingbird</title>

<author>Harper Lee</author>

<year>1960</year>

<genre>Fiction</genre>

</book>

<book>

<title>1984</title>

<author>George Orwell</author>

<year>1949</year>

<genre>Dystopian</genre>

</book>

</library>

package javaapplication8;

import org.w3c.dom.\*;

import javax.xml.parsers.\*;

public class xmlparser {

public static void main(String[] args) {

try {

// Create a new DocumentBuilderFactory and DocumentBuilder

DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

DocumentBuilder builder = factory.newDocumentBuilder();

// Parse the XML file

Document document = builder.parse("C:\\Users\\student\\Desktop\\0041\\JavaApplication8\\src\\javaapplication8\\books.xml");

// Normalize the document

document.getDocumentElement().normalize();

// Get the root element (library)

NodeList nodeList = document.getElementsByTagName("book");

// Loop through each book in the XML document

for (int i = 0; i < nodeList.getLength(); i++) {

Node node = nodeList.item(i);

if (node.getNodeType() == Node.ELEMENT\_NODE) {

Element element = (Element) node;

// Get and print the details of each book

String title =

element.getElementsByTagName("title").item(0).getTextContent();

String author =

element.getElementsByTagName("author").item(0).getTextContent();

String year =

element.getElementsByTagName("year").item(0).getTextContent();

String genre =

element.getElementsByTagName("genre").item(0).getTextContent();

System.out.println("Title: " + title);

System.out.println("Author: " + author);

System.out.println("Year: " + year);

System.out.println("Genre: " + genre);

System.out.println("-----------");

}

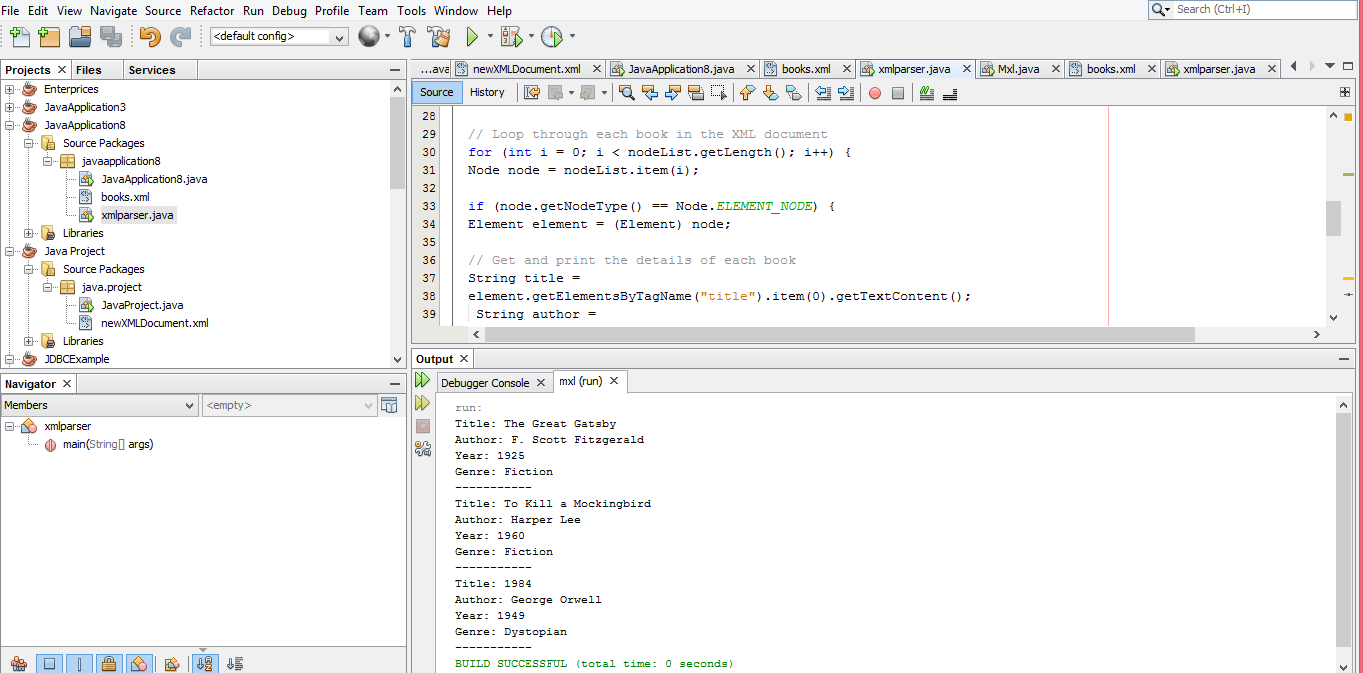
}

} catch (Exception e) {

e.printStackTrace();

}

}

}

**Update Task**

package mxl;

import java.io.File;

import org.w3c.dom.\*;

import javax.xml.parsers.\*;

import javax.xml.transform.Transformer;

import javax.xml.transform.TransformerFactory;

import javax.xml.transform.dom.DOMSource;

import javax.xml.transform.stream.StreamResult;

/\*\*

\*

\* @author student

\*/

public class xmlparser {

public static void main(String[] args) {

try {

// Create a new DocumentBuilderFactory and DocumentBuilder

DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

DocumentBuilder builder = factory.newDocumentBuilder();

// Parse the XML file

Document document = builder.parse("C:\\Users\\student\\Desktop\\0041\\mxl\\src\\mxl\\books.xml");

// Normalize the document

document.getDocumentElement().normalize();

// Get the root element (library)

NodeList nodeList = document.getElementsByTagName("book");

// Loop through each book in the XML document

for (int i = 0; i < nodeList.getLength(); i++) {

Node node = nodeList.item(i);

if (node.getNodeType() == Node.ELEMENT\_NODE) {

Element element = (Element) node;

// Get and print the details of each book

String title =

element.getElementsByTagName("title").item(0).getTextContent();

String author =

element.getElementsByTagName("author").item(0).getTextContent();

String year =

element.getElementsByTagName("year").item(0).getTextContent();

String genre =

element.getElementsByTagName("genre").item(0).getTextContent();

System.out.println("Title: " + title);

System.out.println("Author: " + author);

System.out.println("Year: " + year);

System.out.println("Genre: " + genre);

System.out.println("-----------");

}

}

Element firstBook = (Element) nodeList.item(0);

firstBook.getElementsByTagName("year").item(0).setTextContent("2023");

// Save the modified document

TransformerFactory transformerFactory =

TransformerFactory.newInstance();

Transformer transformer = transformerFactory.newTransformer();

DOMSource source = new DOMSource(document);

StreamResult result = new StreamResult(new File("C:\\Users\\student\\Desktop\\0041\\mxl\\src\\mxl\\updated\_books.xml"));

transformer.transform(source, result);

} catch (Exception e) {

e.printStackTrace();

}

}

}

<?xml version="1.0" encoding="UTF-8" standalone="no"?><library>

<book>

<title>The Great Gatsby</title>

<author>F. Scott Fitzgerald</author>

<year>2023</year>

<genre>Fiction</genre>

</book>

<book>

<title>To Kill a Mockingbird</title>

<author>Harper Lee</author>

<year>1960</year>

<genre>Fiction</genre>

</book>

<book>

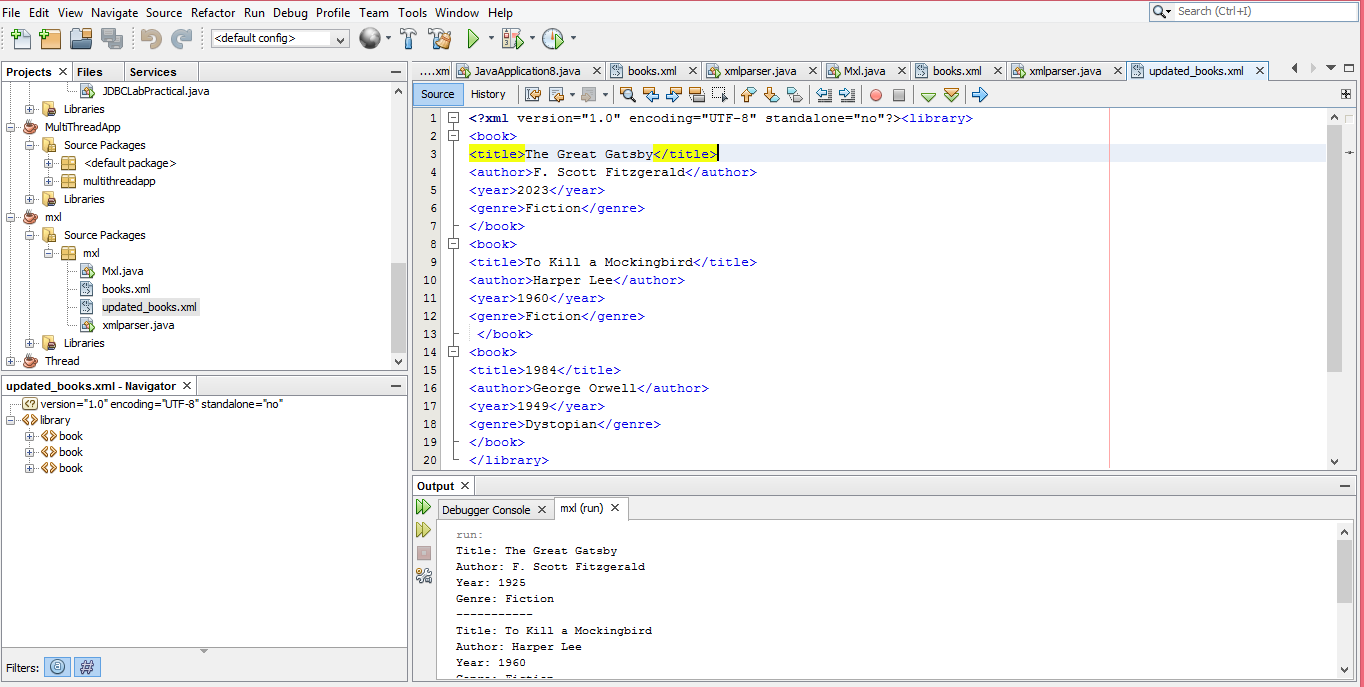
<title>1984</title>

<author>George Orwell</author>

<year>1949</year>

<genre>Dystopian</genre>

</book>

</library>

**Lab Sheet 4: Simple Servlet**

Task 1

package com.example;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

@WebServlet("/displayMessage")

public class DisplayMessageServlet extends HttpServlet {

protected void doGet(HttpServletRequest request, HttpServletResponse

response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html><body>");

out.println("<h1>Welcome to the Java Servlet Lab!</h1>");

out.println("</body></html>");

}

}

<web-app xmlns="http://java.sun.com/xml/ns/javaee"

version="3.0">

<servlet>

<servlet-name>DisplayMessageServlet</servlet-name>

<servlet-class>com.example.DisplayMessageServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>DisplayMessageServlet</servlet-name>

<url-pattern>/displayMessage</url-pattern>

</servlet-mapping>

</web-app>

# Task 02

# <html>

# <head>

# <title>user Input</title>

# <meta charset="UTF-8">

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

# </head>

# <body bgcolor="gray">

# <div><h1>User Input</h1></div>

# <form action="getUserInput" method="POST">

# <center><table style="width:70%">

# <tr><th><h3>Name</h3></th><th><input type="text" name="unname"></th>

# <tr><td colspan="2" align="center"><input type="submit" value="save"></td></tr>

# </table></center>

# </form>

</body>

</html>

Servlet Code (getUserInput.java):

package com.example;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

@WebServlet("/getUserInput")

public class GetUserInputServlet extends HttpServlet {

protected void doPost(HttpServletRequest request, HttpServletResponse

response)

throws ServletException, IOException {

String name = request.getParameter("unname");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

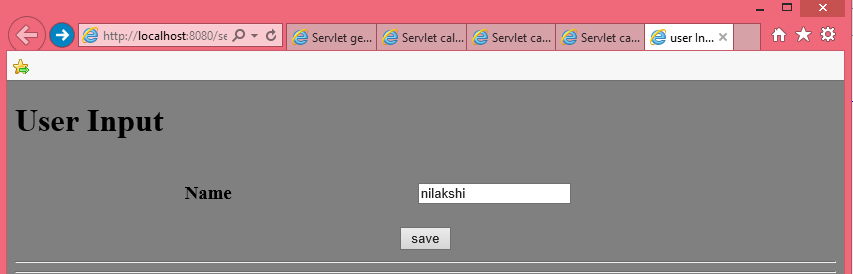
out.println("<html><body>");

out.println("<h1>Input name " + name + "!</h1>");

out.println("</body></html>");

}

}





Task 3

# <html>

# <head>

# <title>user Input</title>

# <meta charset="UTF-8">

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

# </head>

# <h1> calculator</h1>

# <form action="calculateSum" method="POST">

# First Number: <input type="number" name="num1" required><br>

# Second Number: <input type="number" name="num2" required><br>

# <input type="submit" value="Calculate Sum">

# </form>

# </body>

# </html>

# import java.io.IOException;

# import java.io.PrintWriter;

# import javax.servlet.ServletException;

# import javax.servlet.annotation.WebServlet;

# import javax.servlet.http.HttpServlet;

# import javax.servlet.http.HttpServletRequest;

# import javax.servlet.http.HttpServletResponse;

# @WebServlet(urlPatterns = {"/calculateSum"})

# public class calculateSum extends HttpServlet {

# \* Processes requests for both HTTP <code>GET</code> and <code>POST</code>

# \* @param request servlet request

# \* @param response servlet response

# \* @throws ServletException if a servlet-specific error occurs

# \* @throws IOException if an I/O error occurs

# \*/

# protected void processRequest(HttpServletRequest request, HttpServletResponse response)

# throws ServletException, IOException {

# int num1 = Integer.parseInt(request.getParameter("num1"));

# int num2 = Integer.parseInt(request.getParameter("num2"));

# int sum = num1 + num2;

# response.setContentType("text/html;charset=UTF-8");

# try (PrintWriter out = response.getWriter()) {

# out.println("<!DOCTYPE html>");

# out.println("<html>");

# out.println("<head>");

# out.println("<title>Servlet calculateSum</title>");

# out.println("</head>");

# out.println("<body>");

# out.println("<h1>Calculate Sum</h1>");

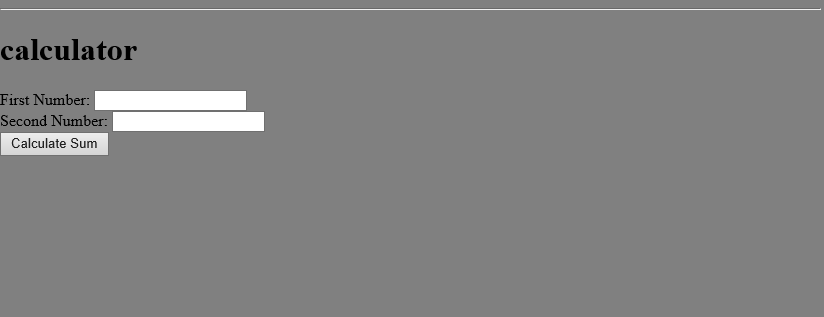
# out.println("<h3>First Number : " + num1 + "</h3>");

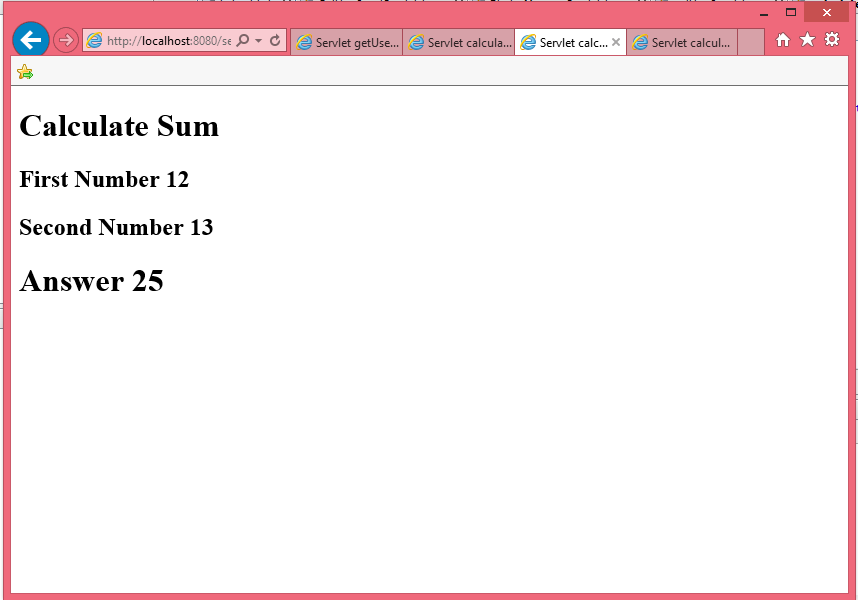
# out.println("<h3>Second Number :" + num2 + "</h3>");

# out.println("<h1>Answer " + sum + "</h1>");

# out.println("</body>");

# out.println("</html>");}}



  
Task 4: Java Servlet with Database CRUD Operations

**Database Setup**

CREATE DATABASE stock\_management;

USE stock\_management;

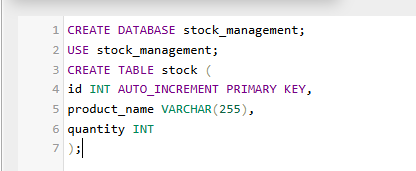
CREATE TABLE stock (

id INT AUTO\_INCREMENT PRIMARY KEY,

product\_name VARCHAR(255),

quantity INT

);



**HTML Form (stockForm.html):**

<!DOCTYPE html>

<html>

<head><title>Stock Management</title></head>

<body>

<h2>Manage Stock</h2>

<form action="stockAction" method="POST">

Product Name: <input type="text" name="product\_name" required><br>

Quantity: <input type="number" name="quantity" required><br>

<input type="submit" name="action" value="Add Product">

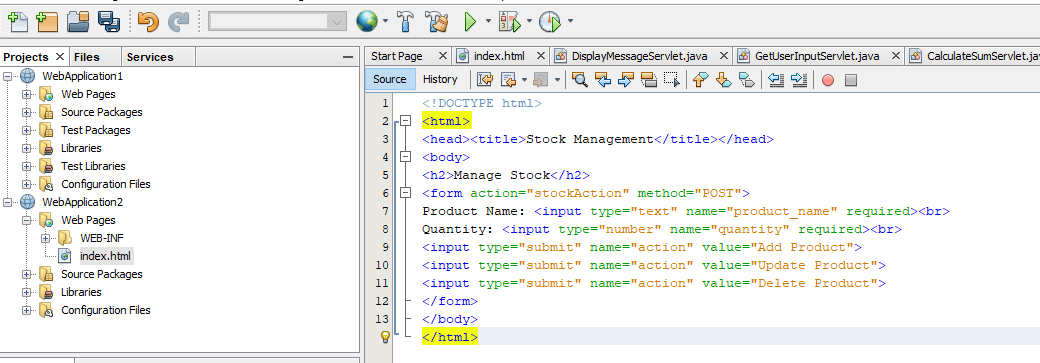
<input type="submit" name="action" value="Update Product">

<input type="submit" name="action" value="Delete Product">

</form>

</body>

</html>



**Servlet Code (StockManagementServlet.java):**

package com.example;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.\*;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

SLIATE HNDIT ENTERPRISE ARCHITECTURE 2 YEAR -2 SEMESTR

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

@WebServlet("/stockAction")

public class StockManagementServlet extends HttpServlet {

private Connection getConnection() throws SQLException {

String url = "jdbc:mysql://localhost:3306/stock\_management";

String username = "root";

String password = "root"; // replace with your database password

return DriverManager.getConnection(url, username, password);

}

protected void doPost(HttpServletRequest request, HttpServletResponse

response)

throws ServletException, IOException {

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

String productName = request.getParameter("product\_name");

int quantity = Integer.parseInt(request.getParameter("quantity"));

try (Connection conn = getConnection()) {

switch(action) {

case "Add Product":

try (PreparedStatement stmt = conn.prepareStatement(

"INSERT INTO stock (product\_name, quantity)

VALUES (?, ?)")) {

stmt.setString(1, productName);

stmt.setInt(2, quantity);

stmt.executeUpdate();

response.getWriter().write("<h1>Product Added

Successfully</h1>");

}

break;

case "Update Product":

try (PreparedStatement stmt = conn.prepareStatement(

"UPDATE stock SET quantity = ? WHERE product\_name

= ?")) {

stmt.setInt(1, quantity);

stmt.setString(2, productName);

stmt.executeUpdate();

response.getWriter().write("<h1>Product Updated

Successfully</h1>");

}

break;

case "Delete Product":

try (PreparedStatement stmt = conn.prepareStatement(

"DELETE FROM stock WHERE product\_name = ?")) {

stmt.setString(1, productName);

stmt.executeUpdate();

response.getWriter().write("<h1>Product Deleted

Successfully</h1>");

}

break;

default:

response.getWriter().write("<h1>Invalid Action</h1>");

}

} catch (SQLException e) {

e.printStackTrace();

response.getWriter().write("<h1>Database Error: " +

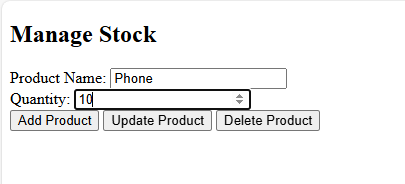
e.getMessage() + "</h1>");

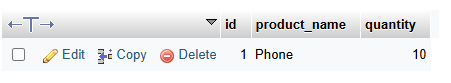
}

}

}

}





Lab Task 5: Display Data from Database on Another Web Page

**Servlet Code (DisplayProductsServlet.java):**

@WebServlet("/displayProducts")

public class DisplayProductsServlet extends HttpServlet {

protected void doGet(HttpServletRequest request, HttpServletResponse

response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

try (Connection conn = getConnection()) {

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM stock");

out.println("<html><body><h1>Stock List</h1>");

while (rs.next()) {

out.println("<p>" + rs.getString("product\_name") + ": " +

rs.getInt("quantity") + "</p>");

}

out.println("</body></html>");

} catch (SQLException e) {

e.printStackTrace();

out.println("<h1>Database Error</h1>");

}

}

}