<u>HNDIT4232 - Enterprise Architecture</u>

Sri Lanka Institute of Advanced Technological Education (SLIATE)

Department of Information Technology

Submitted By: GAM/IT/2022/F/0034- K.T.T.R. Kodithuwakku

Submitted To: Ms. M.V.M. Jayathilaka

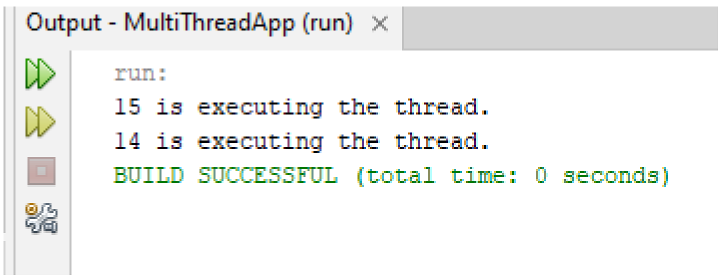
Date of submission: 19.05.2025

Java Thread

01. Create a Simple Thread Class

```
package multithreadapp;

public class SimpleThread extends Thread {
   public void run () {
System.out.println(Thread.currentThread(). getId () + " is executing the thread.");
}}

public class MultiThreadApp {

    public static void main(String[] args) {

       SimpleThread thread1 = new SimpleThread();
       SimpleThread thread2 = new SimpleThread();
       thread1.start(); // Starts thread1
       thread2.start(); // Starts thread2
   }
  }
```

Output:



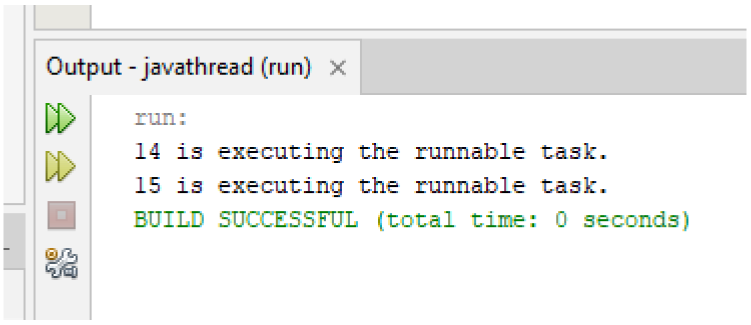
```
run:
15 is executing the thread.
14 is executing the thread.
BUILD SUCCESSFUL (total time: 0 seconds)
```

02. creating runnable class

```
public class RunnableTask implements Runnable {
   @Override
   public void run () {
System.out.println(Thread.currentThread().getId() + " is executing the runnable task.");
}}
```

```
public class Javathread {
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
   }
  }
```

Output:

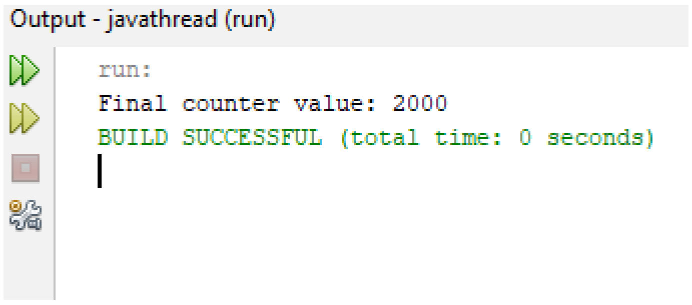


```
run:
14 is executing the runnable task.
15 is executing the runnable task.
BUILD SUCCESSFUL (total time: 0 seconds)
```

03. synchronizing shared resources

```
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();  }}
}

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
   try {
      thread1.join();
   } catch (InterruptedException ex) {
        Logger.getLogger(Javathread.class.getName()).log(Level.SEVERE, null, ex);
    }
thread2.join();
System.out.println("Final counter value: " + counter.getCount());
    }
   }
```

Output:



Output - javathread (run)

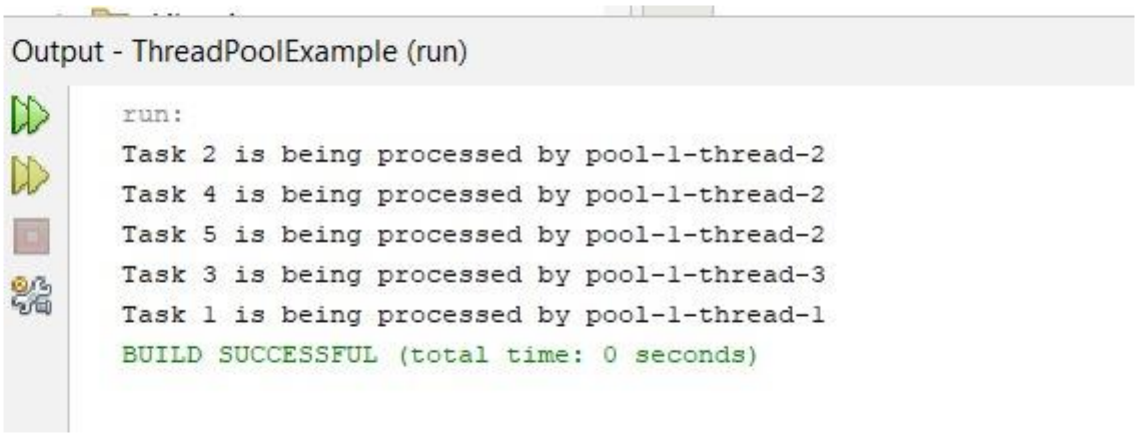
```
run:
Final counter value: 2000
BUILD SUCCESSFUL (total time: 0 seconds)
```

04. Using Executor Service for Thread Pooling

```
package threadpoolexample;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

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();

    }

package threadpoolexample;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

public class task implements Runnnable {
    private int taskId;
public task(int taskId){
    this.taskId=taskId;
}
public void run() {
```

```
System.out.println("Task " + taskId + " is being processed by "
+Thread.currentThread().getName());


}
```

Output:



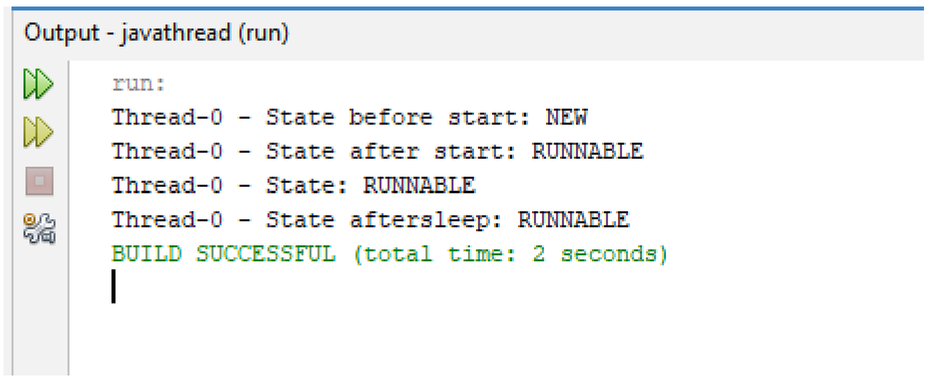
Output - ThreadPoolExample (run)

```
run:
Task 2 is being processed by pool-1-thread-2
Task 4 is being processed by pool-1-thread-2
Task 5 is being processed by pool-1-thread-2
Task 3 is being processed by pool-1-thread-3
Task 1 is being processed by pool-1-thread-1
BUILD SUCCESSFUL (total time: 0 seconds)
```

05. Thread Lifecycle Example

```
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 aftersleep: " +
Thread.currentThread().getState());
}
}

public class Javathread {
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());
}}
```

Output:


Output - javathread (run)

```
run:
Thread-0 - State before start: NEW
Thread-0 - State after start: RUNNABLE
Thread-0 - State: RUNNABLE
Thread-0 - State aftersleep: RUNNABLE
BUILD SUCCESSFUL (total time: 2 seconds)
```

JDBC

Main.java

```
package jdbcexamplea;
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

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);
}
}
```

employeeDAO.java

```
package jdbcexamplea;
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

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 = databaseconnec.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();
}
}
public static List <Employee> getAllEmployees() {
List<Employee> employees = new ArrayList<>();
String sql =("SELECT * FROM employees") ;
try (Connection conn = databaseconnec.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 = ?";
// Corrected SQL
    try (Connection conn = databaseconnec.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 = databaseconnec.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();
}
}
}
```

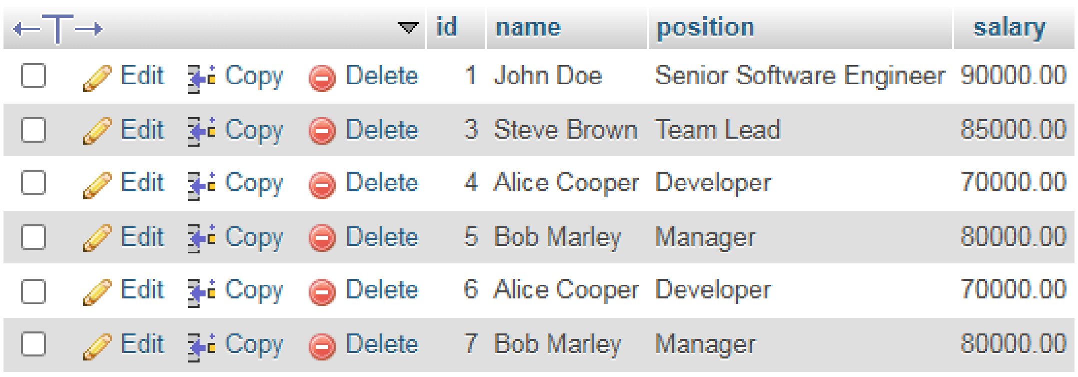
Databaseconnec.java

```
package jdbcexamplea;

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


public class databaseconnec {
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 {
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");
}
}
}
```

Employee.java

```
package jdbcexamplea;

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 + '}';

}

}
```

<u>Output:</u>

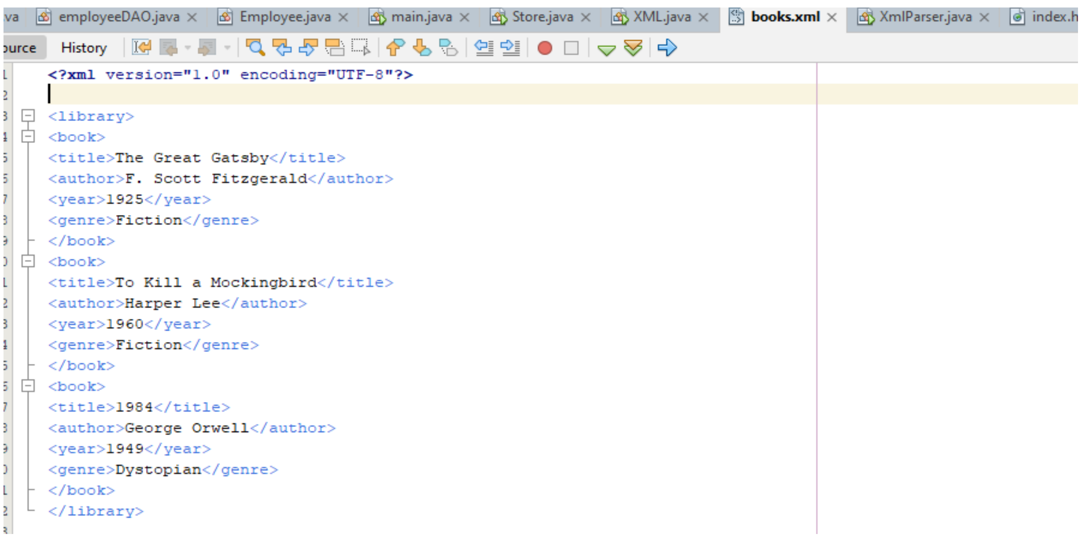


				id	name	position	salary
☐	Edit	Copy	Delete	1	John Doe	Senior Software Engineer	90000.00
☐	Edit	Copy	Delete	3	Steve Brown	Team Lead	85000.00
☐	Edit	Copy	Delete	4	Alice Cooper	Developer	70000.00
☐	Edit	Copy	Delete	5	Bob Marley	Manager	80000.00
☐	Edit	Copy	Delete	6	Alice Cooper	Developer	70000.00
☐	Edit	Copy	Delete	7	Bob Marley	Manager	80000.00

XML

books.xml

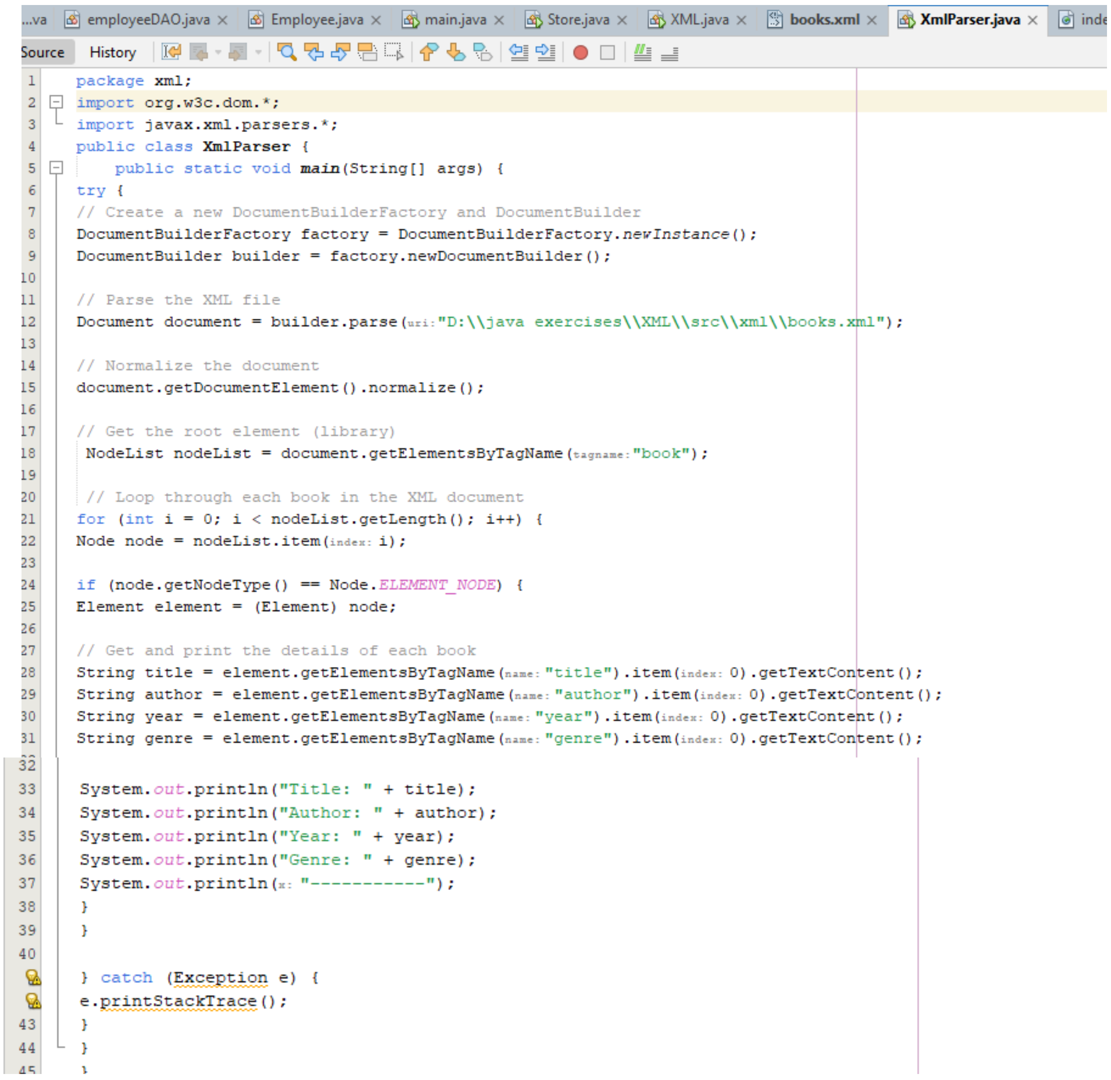
```xml
<?xml version="1.0" encoding="UTF-8"?>

<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>
```



```
<?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>
```

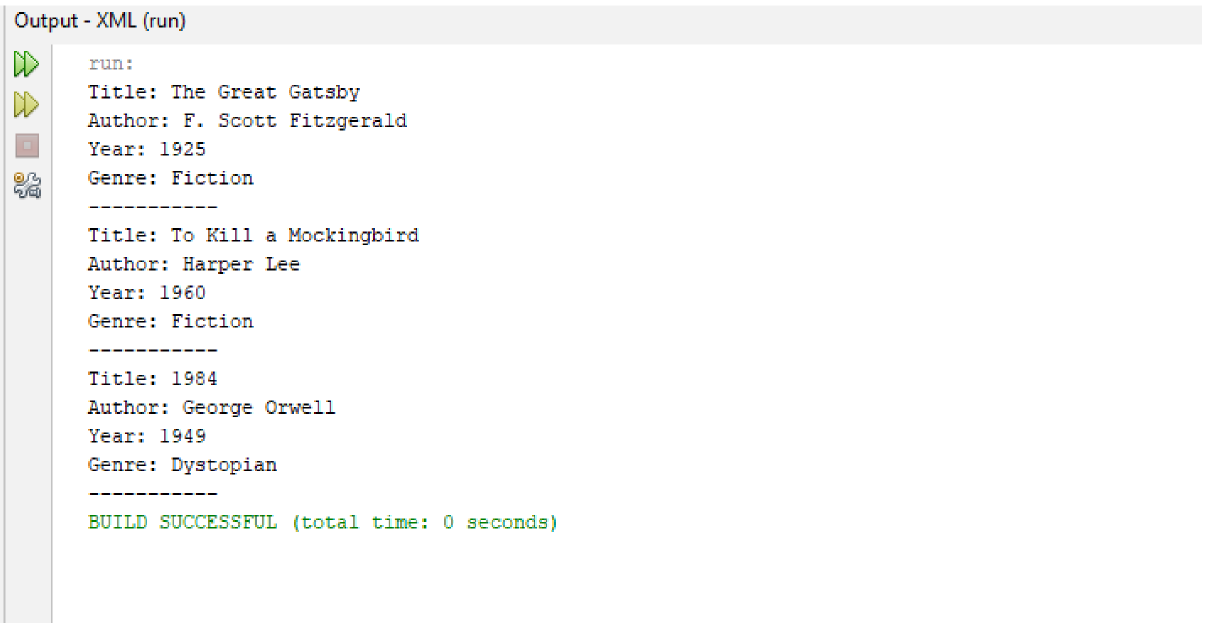
XmlParser.java

```
package xml;
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("D:\\java exercises\\XML\\src\\xml\\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();
}
}
}
```



```java
package xml;
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(uri:"D:\\java exercises\\XML\\src\\xml\\books.xml");

// Normalize the document
document.getDocumentElement().normalize();

// Get the root element (library)
 NodeList nodeList = document.getElementsByTagName(tagname:"book");

 // Loop through each book in the XML document
for (int i = 0; i < nodeList.getLength(); i++) {
Node node = nodeList.item(index: i);

if (node.getNodeType() == Node.ELEMENT_NODE) {
Element element = (Element) node;

// Get and print the details of each book
String title = element.getElementsByTagName(name: "title").item(index: 0).getTextContent();
String author = element.getElementsByTagName(name: "author").item(index: 0).getTextContent();
String year = element.getElementsByTagName(name: "year").item(index: 0).getTextContent();
String genre = element.getElementsByTagName(name: "genre").item(index: 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(x: "-----------");
}
}

} catch (Exception e) {
e.printStackTrace();
}
}
}
```

Output:



```
run:
Title: The Great Gatsby
Author: F. Scott Fitzgerald
Year: 1925
Genre: Fiction
-----------
Title: To Kill a Mockingbird
Author: Harper Lee
Year: 1960
Genre: Fiction
-----------
Title: 1984
Author: George Orwell
Year: 1949
Genre: Dystopian
-----------
BUILD SUCCESSFUL (total time: 0 seconds)
```

Servlet

GetUserInputServlet to handle the form submission and display the user's name.

HTML Form (index.html):

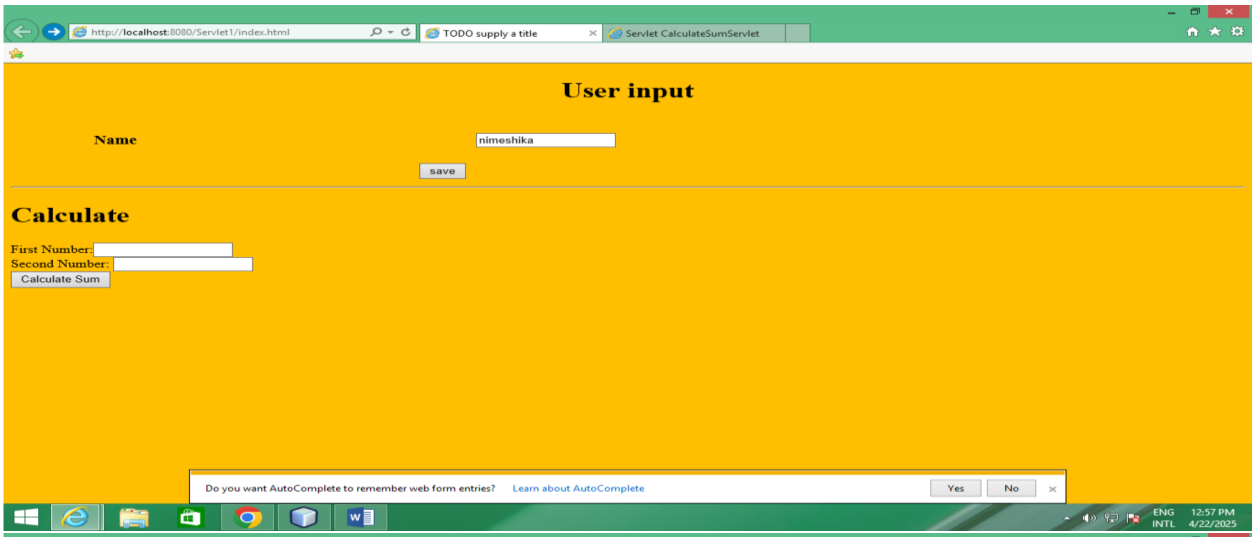
```
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body bgcolor ="#ffbf00">

        <div><h1><center>User input</center></h1></div>

            <form action="getUserInput12" method="POST">
                <table style = "width:70%">
                    <tr> <th> <h3> Name </h3></th><th><input type="text" name="un"</tr>
                <tr> <td colspan="2" align="center"><input type ="submit"
value="save"</td></tr>
            </form>
                </table>
                <tr>
                <hr>
                <h1>Calculate</h1>
                <form action="CalculateSumServlet"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>
                      </tr>
```

```
</form>


    </body>
</html>
```

(getUserInput12.java)

```
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;

/**
 *
 * @author student
 */
@WebServlet(urlPatterns = {"/getUserInput12"})
public class getUserInput12 extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
     * methods.
     *
     * @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 {
```

```
        String name = request.getParameter("un");
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample code. */
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet getUserInput12</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1> Input name " + name + "</h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on
the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    /**
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```


http://localhost:8080/Servlet1/index.html
TODO supply a title
Servlet CalculateSumServlet

User input

Name nimeshika

save

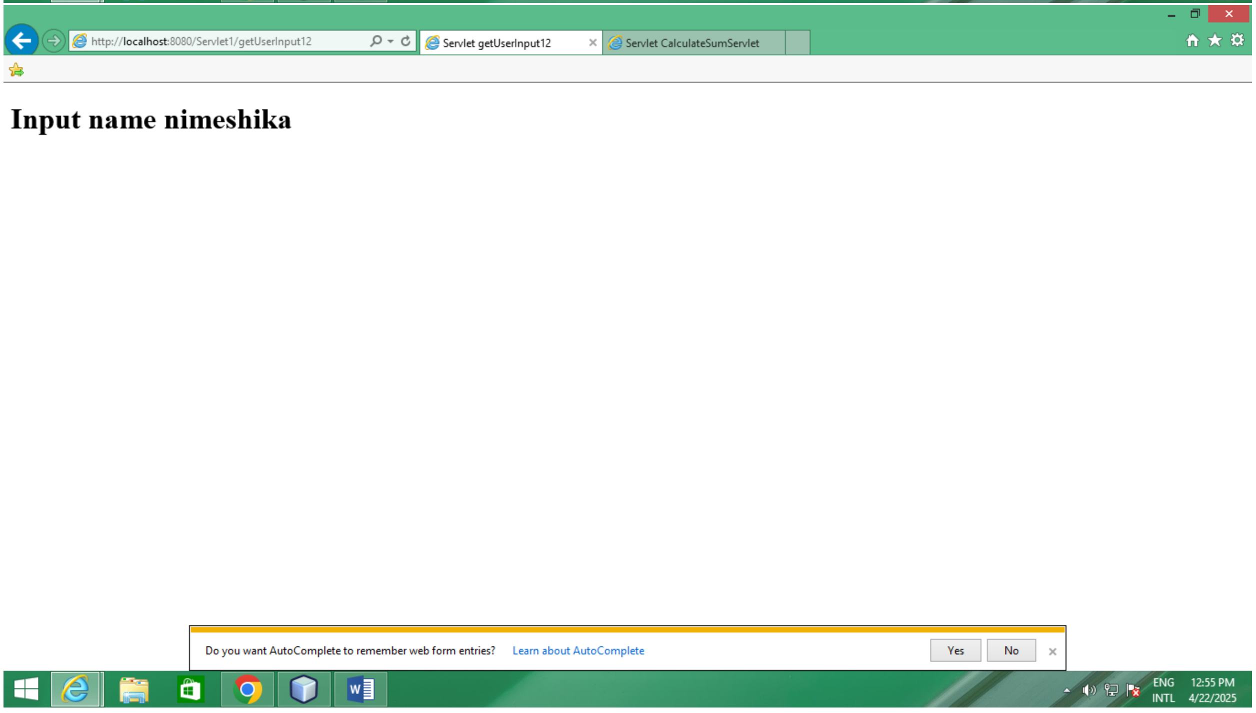
Calculate

First Number:

Second Number:

Calculate Sum

Do you want AutoComplete to remember web form entries? Learn about AutoComplete Yes No

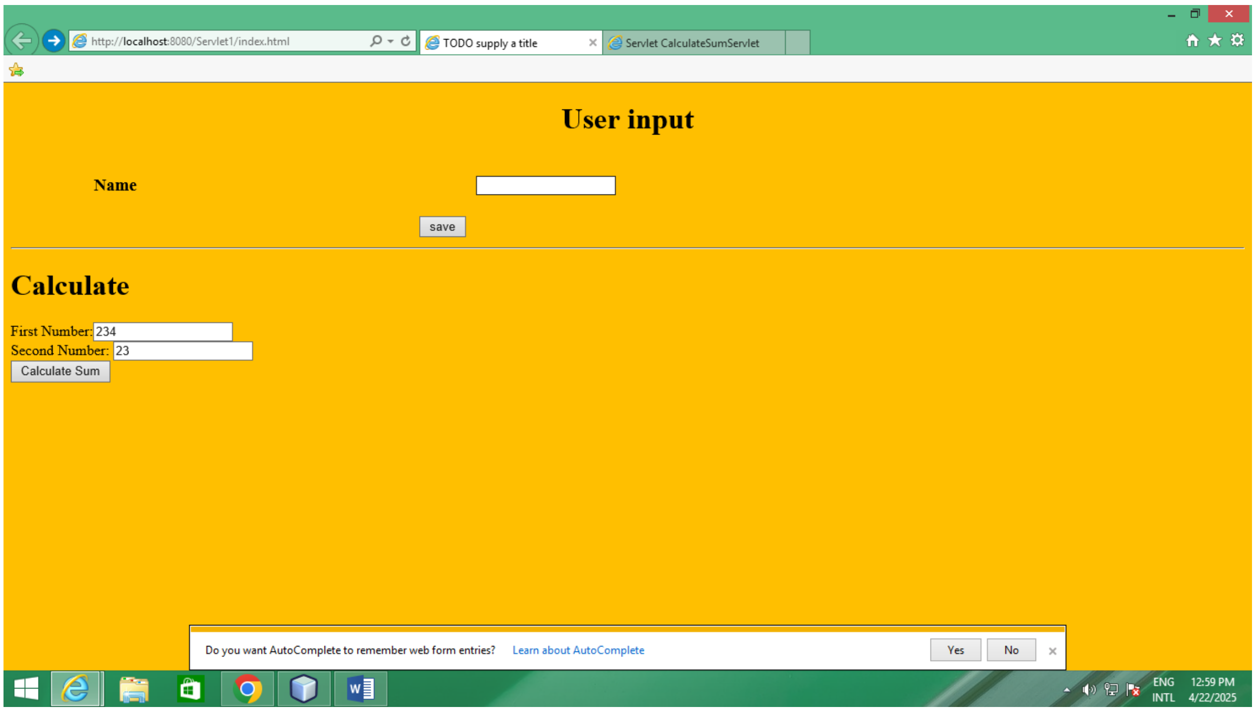

http://localhost:8080/Servlet1/getUserInput12
Servlet getUserInput12
Servlet CalculateSumServlet

Input name nimeshika

Do you want AutoComplete to remember web form entries? Learn about AutoComplete Yes No

```
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;

/**
 *
 * @author student
 */
@WebServlet(urlPatterns = {"/CalculateSumServlet"})
public class CalculateSumServlet extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
     * methods.
     *
     * @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()) {
            /* TODO output your page here. You may use following sample code. */
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet CalculateSumServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1> Calculate Sum</h1>");
            out.println("<h3> First user input:"+num1+"</h1>");
            out.println("<h3> Second  user input:"+num2+"</h1>");
            out.println("<h1> Answer:"+sum+"</h1>");
            //out.println("<h1>The sum of " + num1 + " and " + num2 + " is: " + sum + "</h1>");
            //out.println("<h1>Servlet CalculateSumServlet at " + request.getContextPath() +
"</h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on
the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
```

```
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```



http://localhost:8080/Servlet1/index.html

TODO supply a title | Servlet CalculateSumServlet

User input

Name

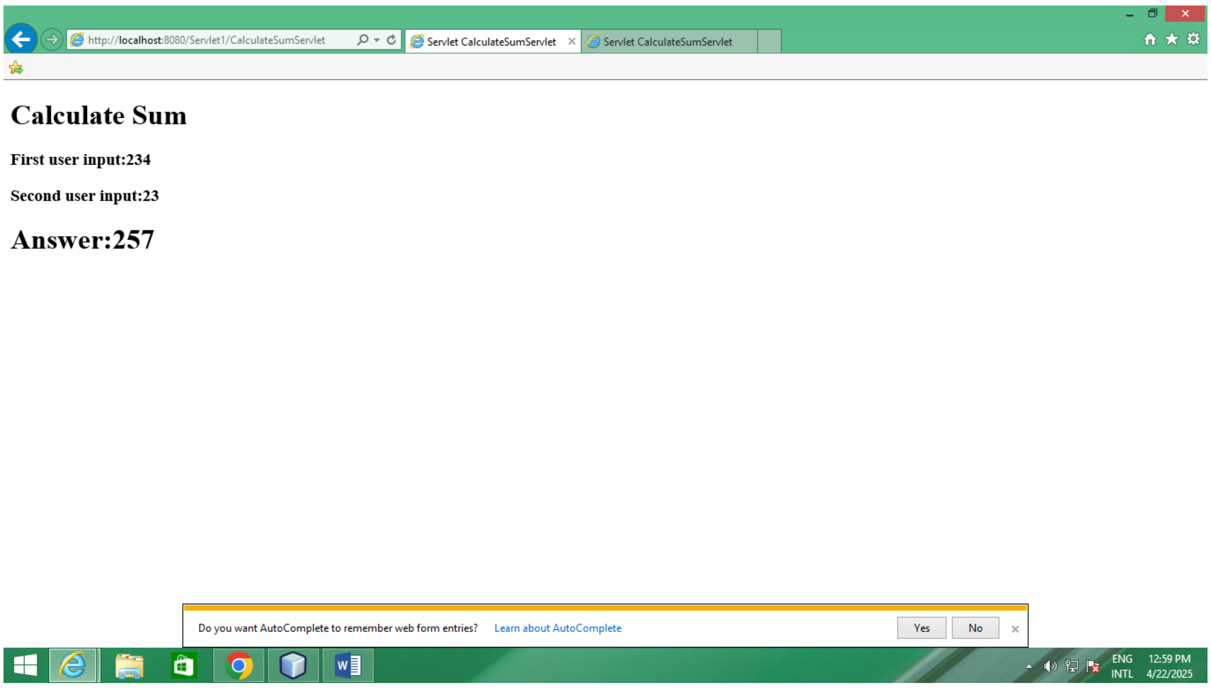
save

Calculate

First Number: 234
Second Number: 23

Calculate Sum

Do you want AutoComplete to remember web form entries? Learn about AutoComplete Yes No



http://localhost:8080/Servlet1/CalculateSumServlet

Servlet CalculateSumServlet | Servlet CalculateSumServlet

Calculate Sum

First user input:234

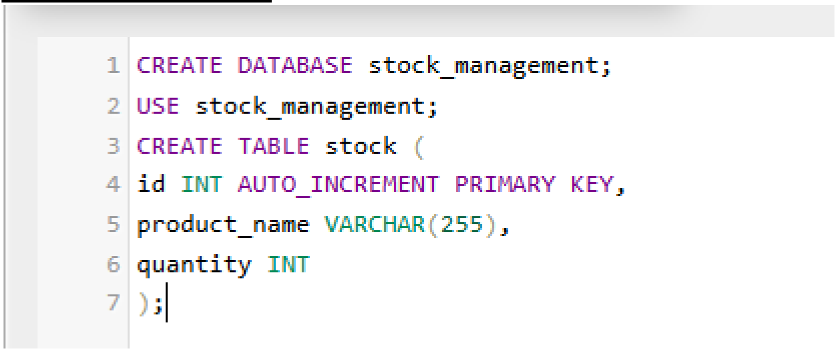
Second user input:23

Answer:257

Do you want AutoComplete to remember web form entries? Learn about AutoComplete Yes No

Lab 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
);
```

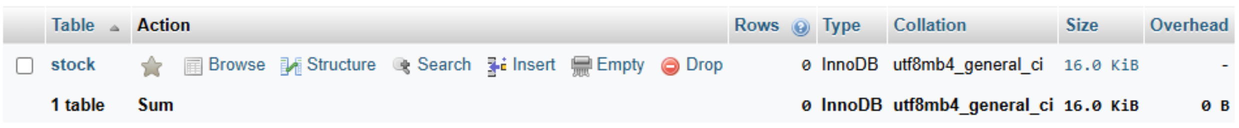
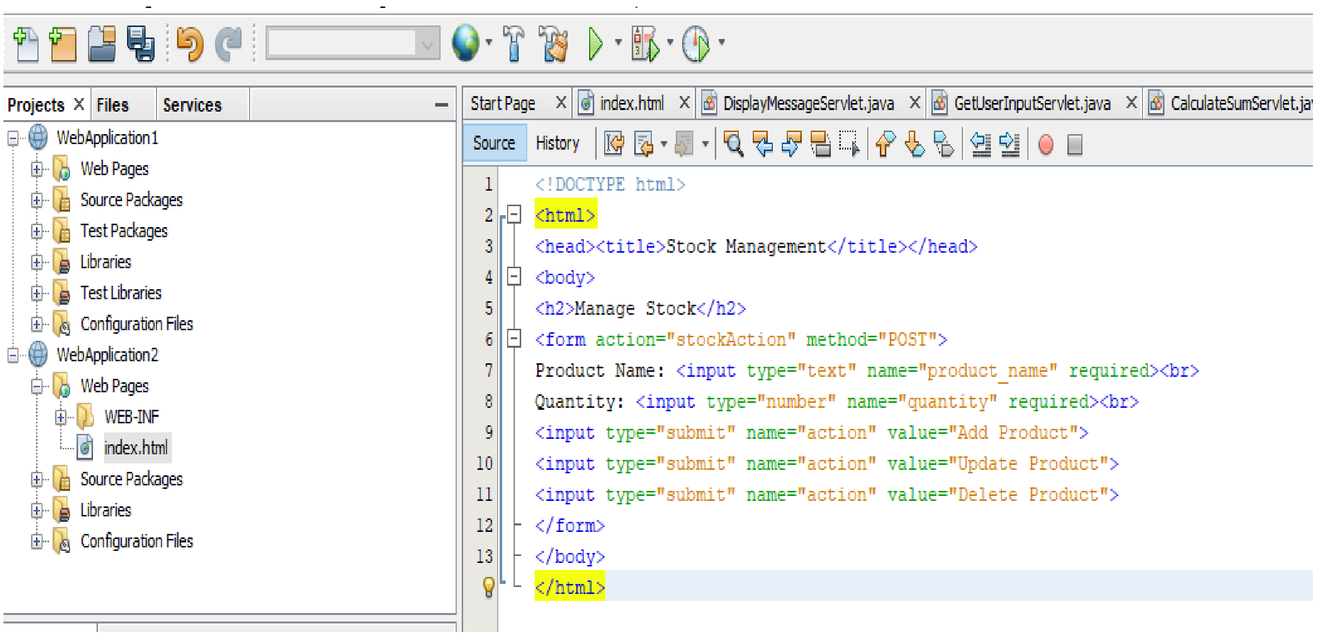


Table	Action	Rows	Type	Collation	Size	Overhead
stock	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
1 table	Sum	0	InnoDB	utf8mb4_general_ci	16.0 KiB	0 B

HTML Form (stockForm.html)



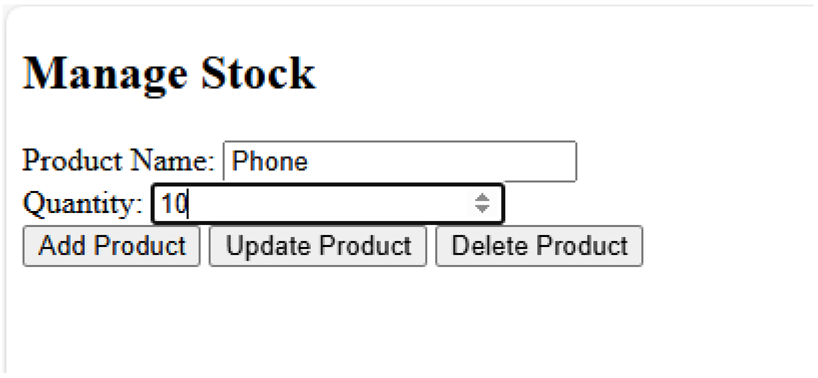
```
<!DOCTYPE html>
<html>
<head><title>Stock Management</title></head>
<body>
<h2>Manage Stock</h2>
<form action="stockAction" method="POST">
Product Name: <input type="text" name="product_name" required><br>
Quantity: <input type="number" name="quantity" required><br>
<input type="submit" name="action" 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):

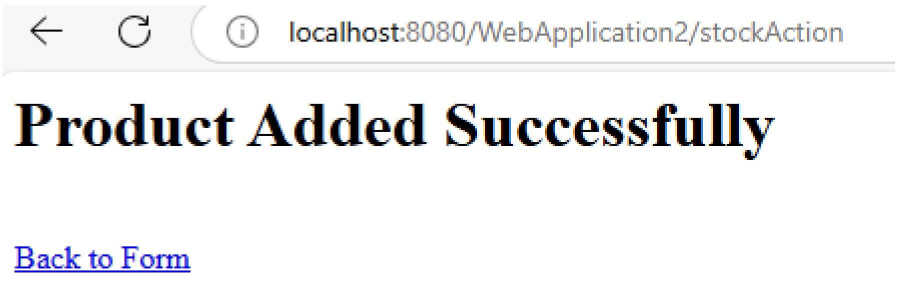
package com.example;

```
import java.io.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;

@WebServlet("/stockAction")
public class StockManagementServlet extends HttpServlet {

    // Initialize driver when servlet loads
    @Override
    public void init() throws ServletException {
        try {
Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            throw new ServletException("MySQL JDBC Driver not found", e);
        }
    }

    private Connection getConnection() throws SQLException {
        String url =
"jdbc:mysql://localhost:3306/stock_management?useSSL=false&serverTimezone=UTC";
        String username = "root";
        String password = "316830059";
        return DriverManager.getConnection(url, username, password);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
response.setContentType("text/html");
PrintWriter out = response.getWriter();

        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":
addProduct(conn, productName, quantity, out);
                    break;
                case "Update Product":
updateProduct(conn, productName, quantity, out);
                    break;
                case "Delete Product":
deleteProduct(conn, productName, out);
                    break;
                default:
out.println("<h1>Invalid Action</h1>");
            }
        } catch (SQLException e) {
out.println("<h1>Database Error: " + e.getMessage() + "</h1>");
e.printStackTrace();
        }

out.println("<br><a href='stockForm.html'>Back to Form</a>");
    }

    private void addProduct(Connection conn, String name, int quantity, PrintWriter out)
            throws SQLException {
        String sql = "INSERT INTO stock (product_name, quantity) VALUES (?, ?)";
        try (PreparedStatementstmt = conn.prepareStatement(sql)) {
stmt.setString(1, name);
stmt.setInt(2, quantity);
stmt.executeUpdate();
out.println("<h1>Product Added Successfully</h1>");
        }
    }

    private void updateProduct(Connection conn, String name, int quantity, PrintWriter out)
            throws SQLException {
        String sql = "UPDATE stock SET quantity = ? WHERE product_name= ?";
        try (PreparedStatementstmt = conn.prepareStatement(sql)) {
stmt.setInt(1, quantity);
stmt.setString(2, name);
            int rows = stmt.executeUpdate();
            if (rows > 0) {
out.println("<h1>Product Updated Successfully</h1>");
            } else {
out.println("<h1>Product Not Found</h1>");
            }
        }
    }

    private void deleteProduct(Connection conn, String name, PrintWriter out)
            throws SQLException {
        String sql = "DELETE FROM stock WHERE product_name= ?";
        try (PreparedStatementstmt = conn.prepareStatement(sql)) {
stmt.setString(1, name);
            int rows = stmt.executeUpdate();
            if (rows > 0) {
out.println("<h1>Product Deleted Successfully</h1>");
            } else {
out.println("<h1>Product Not Found</h1>");
            }
        }
    } }
```

Output

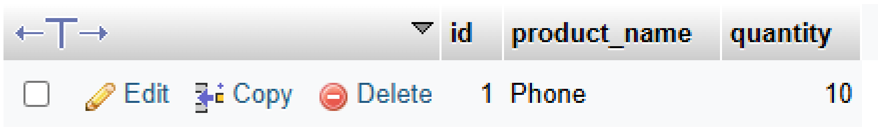

Manage Stock

Product Name: Phone
Quantity: 10


localhost:8080/WebApplication2/stockAction

Product Added Successfully

Back to Form



id	product_name	quantity
1	Phone	10

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

Servlet Code (DisplayProductsServlet.java):

```
package com.example;


import java.io.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
@WebServlet("/displayProducts")
public class DisplayProductsServlet extends HttpServlet {
   // Reuse your existing connection method
   private Connection getConnection() throws SQLException {
      String url =
"jdbc:mysql://localhost:3306/stock_management?useSSL=false&serverTimezone=UTC";
      String username = "root";
      String password = "316830059";
```

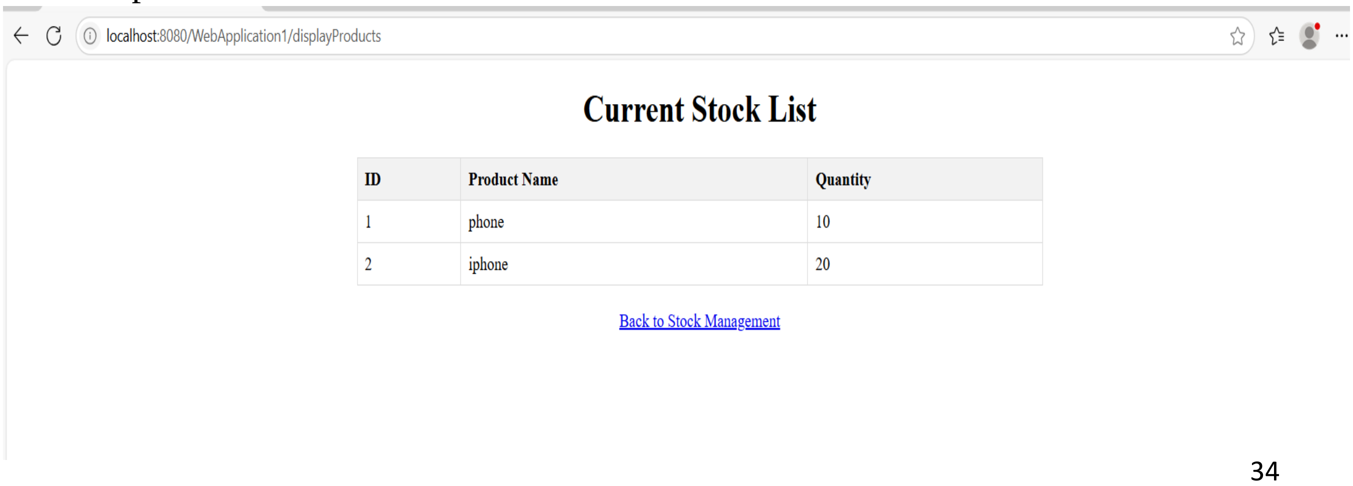
```
        return DriverManager.getConnection(url, username, password);
    }
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
response.setContentType("text/html");
PrintWriter out = response.getWriter();
out.println("<!DOCTYPE html>");
out.println("<html>");
out.println("<head>");
out.println("<title>Stock List</title>");
out.println("<style>");
out.println("table { border-collapse: collapse; width: 50%; margin: 20px auto; }");
out.println("th, td { border: 1px solid #ddd; padding: 8px; text-align: left; }");
out.println("th { background-color: #f2f2f2; }");
out.println("</style>");
out.println("</head>");
out.println("<body>");
out.println("<h1 style='text-align: center;'>Current Stock List</h1>");

try (Connection conn = getConnection();
             Statement stmt = conn.createStatement();
ResultSetrs = stmt.executeQuery("SELECT * FROM stock")) {
out.println("<table>");
out.println("<tr><th>ID</th><th>Product Name</th><th>Quantity</th></tr>");

            while (rs.next()) {
out.println("<tr>");
out.println("<td>" + rs.getInt("id") + "</td>");
out.println("<td>" + rs.getString("product_name") + "</td>");
out.println("<td>" + rs.getInt("quantity") + "</td>");
out.println("</tr>");
            }

out.println("</table>");
        } catch (SQLException e) {
out.println("<h2 style='color: red; text-align: center;'>Error retrieving stock: "
                    + e.getMessage() + "</h2>");
e.printStackTrace();      }

out.println("<div style='text-align: center; margin-top: 20px;'>");
out.println("<a href='stockForm.html'>Back to Stock Management</a>");
out.println("</div>");
out.println("</body>");
out.println("</html>");
```

```
    }
}
```

<u>Updated stockForm.html</u>

```
<!DOCTYPE html>
<html>
<head><title>Stock Management</title>
<style>    body { font-family: Arial, sans-serif; margin: 20px; }
           form { max-width: 500px; margin: 0 auto; padding: 20px; border: 1px solid #ddd;
border-radius: 5px; }
           input[type="text"], input[type="number"] { width: 100%; padding: 8px; margin:
5px 0 15px; }
            input[type="submit"] { padding: 8px 15px; margin-right: 10px; }
.view-link { display: block; text-align: center; margin-top: 20px; }
</style>
</head>
<body><h2 style="text-align: center;">   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>
<div class="view-link">
<a href="displayProducts"> View All Products   </a>
</div>
</body>
</html>
```

Output:



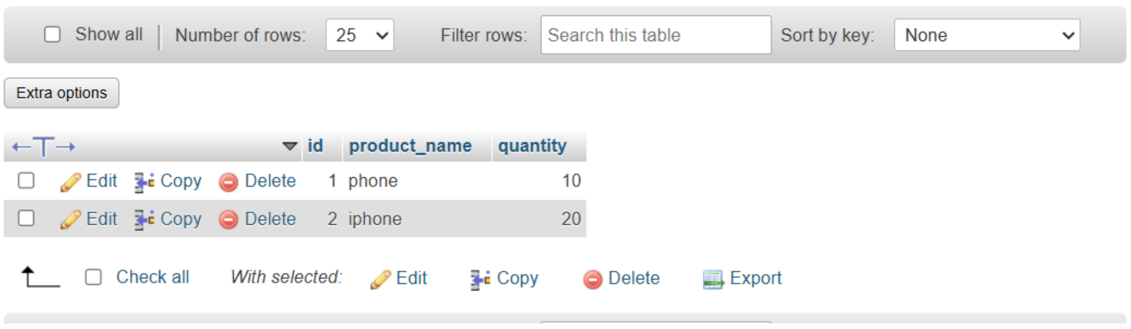
localhost:8080/WebApplication1/displayProducts

Current Stock List

ID	Product Name	Quantity
1	phone	10
2	iphone	20

Back to Stock Management

Database



	id	product_name	quantity
Edit Copy Delete	1	phone	10
Edit Copy Delete	2	iphone	20