Government Engineering College, Rajkot Computer Engineering Department

Advance Java Programming

(Subject Code –3160707)

# LAB

# MANUAL

**6th Semester Computer Engineering**

*YASH A. LATHIYA*

*CE\_K2*

*200200107095*

**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr.**  **No.** | **List of Practical** | **Pag e**  **No.** | **Date of Submission** | **Signature** |
| **1.1** | Write java program(s) which accepts names of website from user and prints IP Address of it. The program should be stopped when user enters “exit” as an input. | **5** | **31/01/2023** |  |
| **1.2** | Write a java program (client) which accepts integer from user, send it to the server, server sorts all the numbers and returns the sorted numbers to the clients. Example: Client enters and sends following number => 10, 25, 4, 29, and 15, exit. Output on client program should be => 4, 10, 15, 25,29. | **6** | **31/01/2023** |  |
| **1.3** | Write a menu based java program which performs following.  1. Reverse of the given string  2. Converts string into upper case  3. Converts string into lower case  4. Counts string length | **9** | **07/02/2023** |  |
| **1.4** | Write a java program(s) to implement simple chat application. | **11** | **07/02/2023** |  |
| **1.5** | Write java program to send hello message to the server using UDP. | **14** | **14/02/2023** |  |
| **2.1** | Write a java program to insert a student record in Student Table. | **16** | **14/02/2023** |  |
| **2.2** | Write a java program to insert a student record in Student Table using Prepared Statement. | **18** | **21/02/2023** |  |
| **2.3** | Write a java program to call a stored procedure using Callable Statement Interface. | **20** | **21/02/2023** |  |
| **3.1** | Write a servlet application to print the current date and time. | **22** | **28/02/2023** |  |
| **3.2** | Write a Servlet application to count the total number of visits on your website. | **24** | **28/02/2023** |  |
| **3.3** | Write a Servlet application to authenticate user. | **25** | **28/02/2023** |  |
| **3.4** | Write a Servlet application to perform sign up on website. Fields : username, password, Name, Email and Mobile number | **27** | **07/03/2023** |  |
| **3.5** | Write a Servlet application to fetch Mark sheet of a given enrollment number. | **29** | **07/03/2023** |  |
| **3.6** | To develop a web application using servlet event handling and filters. | **31** | **14/03/2023** |  |
| **3.7** | Develop a web application that demonstrates the use of session level events.  Implement the servlet that adds two numbers. Use the two filters in a chain mapped to servlet such that first filter checks the number for valid format and second checks them for a range. If both conditions are fulfilled then only servlet perform the operation and displays the result, otherwise appropriate message from respective filter. | **32** | **14/03/2023** |  |
| **4.1** | To develop a web application using JSP and JDBC Write two HTML pages named as register.html and login.html.  Where, “register.html‟ has one UI-form, which collects user’s details like full name, age, mobile number, email address, password and confirm password. This register.html page will submit data to servlet named as “RegisterServlet.java”. RegisterServlet will insert user’s details to database and on successful insertion, it will redirect user tologin.html with appropriate message otherwise show appropriate error message.  “login.html” asks either mobile number OR email address and password details for login.  For valid user, redirect him/her to welcome.jsp page and also start new session.  In welcome.jsp,  1. Check whether user is eligible for Vote or not using session data and display appropriate message.  2. Display logout button, so that once user will click on logout, it will close the user's session and redirect to login.html.  For invalid user, display appropriate message like at login page  1. Invalid mobile Number/email address or password. OR  2. Mobile Number/email address doesn't exist. | **35** | **21/03/2023** |  |
| **4.2** | Write a JSP page to demonstrate various tag of SQL tag library | **42** | **21/03/2023** |  |
| **5.1** | Write a program to demonstrate the use of JSF Convertor Tag and Validation Tag. | **44** | **28/03/2023** |  |
| **5.2** | Write a program that demonstrates the use of JSF Event Handling and Database Access. | **46** | **04/04/2023** |  |
| **6.1** | Write a hibernate application to save object of Employee class into database.  Employee class contains following properties  • EID  • ENAME  • ESalary  Write SQL query for table as well. | **48** | **11/04/2023** |  |
| **6.2** | Write a menu based program to perform following operations using HQL (consider above class and table)  a. Insert new record  b. Update Existing Record  c. Delete Record | **50** | **02/05/2023** |  |
| **7.1** | Create a MVC Spring Application which demonstrates Dependency Injection using constructor Consider an Employee(E-id, E\_name) class. Write all necessary classes/files. | **54** | **09/05/2023** |  |

***Unit 1 :Java Networking***

**Practical 1.1:**

**Write java program(s) which accepts names of website from user and prints IP Address of it. The program should be stopped when user enters “exit” as an input.**

import java.util.Scanner;

import java.net.\*;

public class **Practical1** {

/\*\*

\* @param args the command line arguments

\* @param {str} URL of Website

\*/

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

String str = "";

while(str != "exit"){

str = sc.nextLine();

try{

InetAddress ip = InetAddress.getByName(new URL(str).getHost());

System.out.println(ip);

}

catch(Exception e){

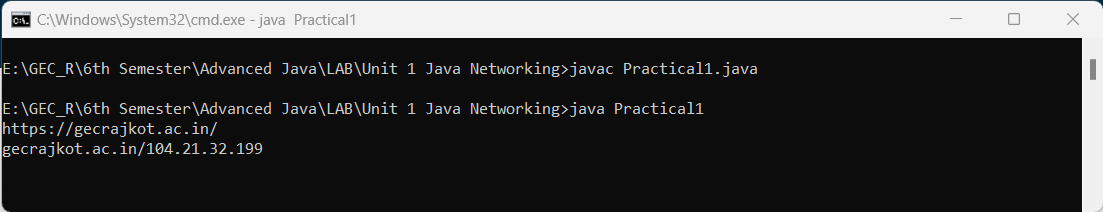
System.out.println("Please enter correct URL");

}

}

}

}



**Practical 1.2**

**Write a java program (client) which accepts integer from user, send it to the server, server sorts all the numbers and returns the sorted numbers to the clients. Example: Client enters and sends following number =>10, 25, 4, 29, and 15, exit. Output on client program should be => 4, 10, 15, 25,29.**

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class **Practical2Client** {

/\*\*

\* @param args the command line arguments

\* @param {numbers} List of num

\* @param {num} a number which is entered by client at command line

\*/

public static void main(String[] args) {

try {

Socket socket = new Socket("localhost", 3000); // connect to server

System.out.println("Connected to server.");

// create input/output streams

InputStream is = socket.getInputStream();

DataInputStream dis = new DataInputStream(is);

OutputStream os = socket.getOutputStream();

DataOutputStream dos = new DataOutputStream(os);

Scanner sc = new Scanner(System.in);

List<Integer> numbers = new ArrayList<Integer>();

System.out.println("Enter numbers to be sorted (enter -1 to stop):");

// read numbers from user input and send to server

while (true) {

int num = sc.nextInt();

if (num == -1) {

break;

}

numbers.add(num);

dos.writeInt(num);

}

dos.writeInt(-1); // signal end of input

// read sorted numbers from server and print

System.out.println("Sorted numbers:");

while (true) {

try {

int num = dis.readInt();

if (num == -1) {

break;

}

System.out.print(num + " ");

} catch (Exception e) {

e.printStackTrace();

break;

}

}

socket.close(); // close socket

} catch (Exception e) {

e.printStackTrace();

}

}

}

public class **Practical2Server** {

/\*\*

\* @param args the command line arguments

\* @param {numbers} List of num

\* @param {num} a number which is entered by client at command line

\*/

public static void main(String[] args) {

try {

ServerSocket serverSocket = new ServerSocket(3000); // create server socket

System.out.println("Server started. Waiting for client...");

Socket socket = serverSocket.accept(); // accept client connection

System.out.println("Client connected.");

// create input/output streams

InputStream is = socket.getInputStream();

DataInputStream dis = new DataInputStream(is);

OutputStream os = socket.getOutputStream();

DataOutputStream dos = new DataOutputStream(os);

List<Integer> numbers = new ArrayList<Integer>();

while (true) {

try {

int num = dis.readInt(); // read number sent by client

if (num == -1) { // client signals end of input

break;

}

numbers.add(num);

} catch (Exception e) {

e.printStackTrace();

break;

}

}

Collections.sort(numbers); // sort the numbers

// send sorted numbers back to client

for (int num : numbers) {

dos.writeInt(num);

}

dos.writeInt(-1); // signal end of output

System.out.println("Numbers sorted and sent back to client.");

serverSocket.close(); // close server socket

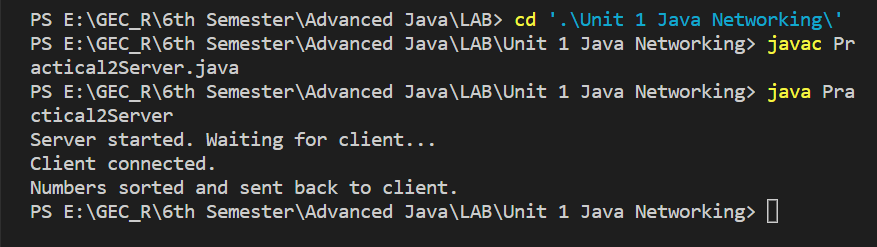
} catch (Exception e) {

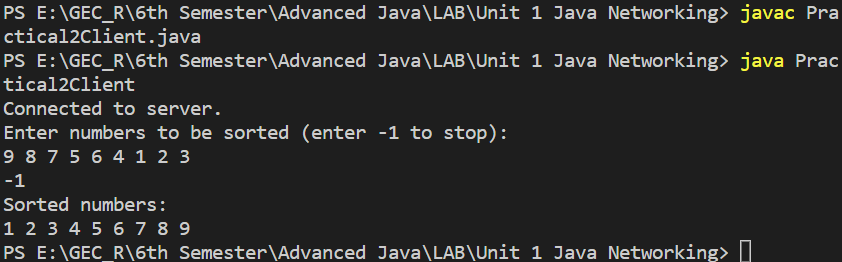
e.printStackTrace();

}

}

}





**Practical 1.3**

**Write a menu based java program which performs following.**

**1. Reverse of the given string**

**2. Converts string into upper case**

**3. Converts string into lower case**

**4. Counts string length**

import java.util.Scanner;

public class **Practical3** {

/\*\*

\* @param args the command line arguments

\* @param {str} string on which operations to be performed

\* @param {operation} operation number {1 , 2, 3, 4} details mentioned on line 30- 33

\*/

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

while(true){

System.out.println("Enter String -- enter exit to terminate program");

String str = sc.nextLine();

System.out.println("enter 1 for reverse the "+str);

System.out.println("enter 2 for converting the "+str+ " into upper case");

System.out.println("enter 3 for converting the "+str+ " into lower case");

System.out.println("enter 4 for count length of the "+str);

int operation = sc.nextInt();

sc.nextLine();

switch(operation){

case 1: StringBuilder sb = new StringBuilder(); //reverse the string

sb.append(str);

sb.reverse();

System.out.println(sb);

break;

case 2: System.out.println(str.toUpperCase()); // to upper case

break;

case 3: System.out.println(str.toLowerCase()); //to lower case

break;

case 4: System.out.println("length of string is "+str.length()); //length of string

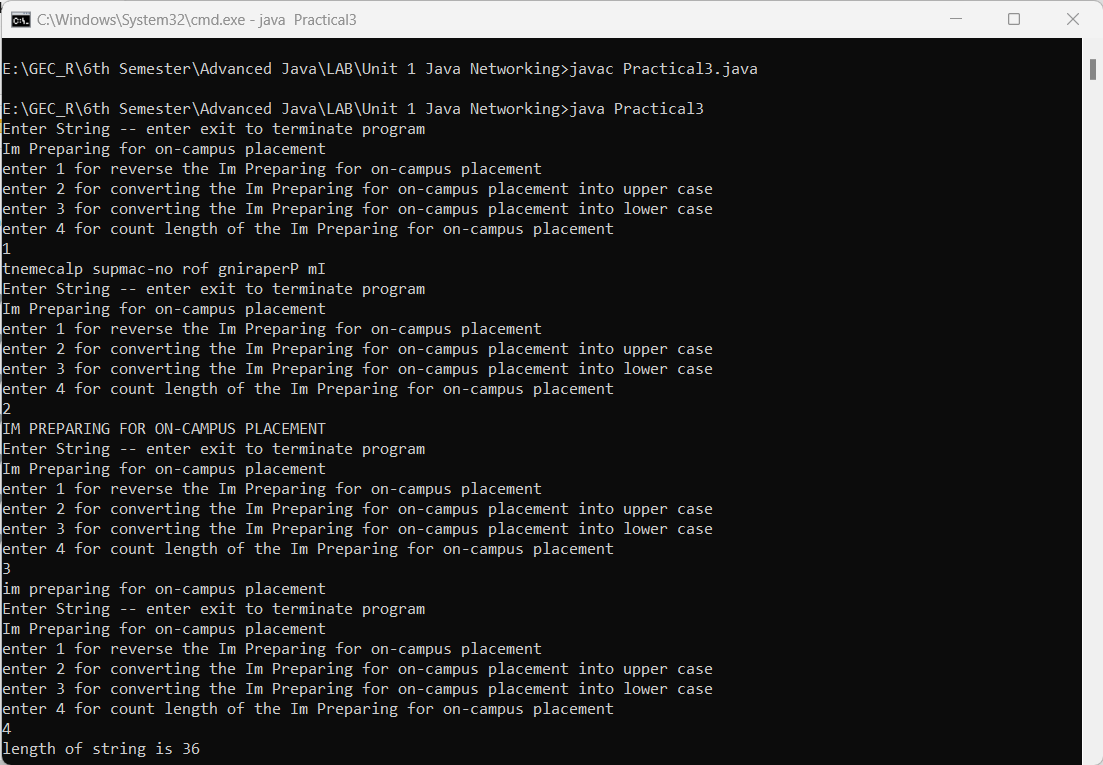
break;

}

}

}

}



**Practical 1.4**

**Write a java program(s) to implement simple chat application.**

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class **Practical4Client** {

/\*\*

\* @param args the command line arguments

\* @param {str} message which will be send to server

\* @param {msg} message which is send by server

\*/

public static void main(String[] args) {

try {

Socket socket = new Socket("localhost", 3000); // connect to server

System.out.println("Connected to server.");

// create input/output streams

InputStream is = socket.getInputStream();

DataInputStream dis = new DataInputStream(is);

OutputStream os = socket.getOutputStream();

DataOutputStream dos = new DataOutputStream(os);

Scanner sc = new Scanner(System.in);

System.out.println("Enter Message : ");

// read numbers from user input and send to server

while (true) {

String str = sc.nextLine();

dos.writeUTF(str);

try{

String msg = dis.readUTF();

}

catch(Exception e){

System.out.println("");

}

}

} catch (Exception e) {

e.printStackTrace();

}

}

}

public class **Practical4Server** {

/\*\*

\* @param args the command line arguments

\* @param {str} message which will be send to client

\* @param {msg} message which is send by client

\*/

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

try {

ServerSocket serverSocket = new ServerSocket(3000); // create server socket

System.out.println("Server started. Waiting for client...");

Socket socket = serverSocket.accept(); // accept client connection

System.out.println("Client connected.");

// create input/output streams

InputStream is = socket.getInputStream();

DataInputStream dis = new DataInputStream(is);

OutputStream os = socket.getOutputStream();

DataOutputStream dos = new DataOutputStream(os);

while (true) {

try {

String str = dis.readUTF();

System.out.println(str);

try{

String msg = sc.nextLine();

dos.writeUTF(msg);

}

catch(Exception e){

System.out.println("");

}

} catch (Exception e) {

e.printStackTrace();

break;

}

}

serverSocket.close(); // close server socket

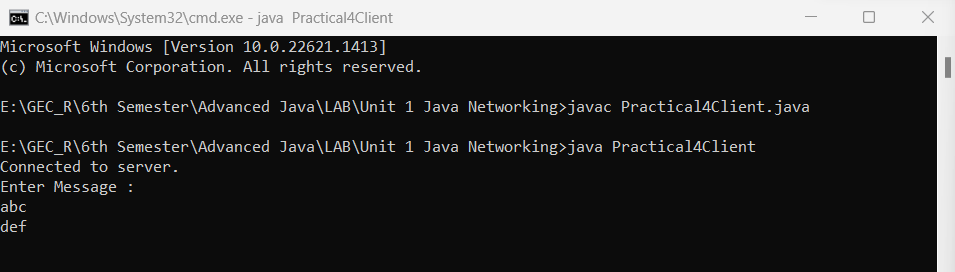
} catch (Exception e) {

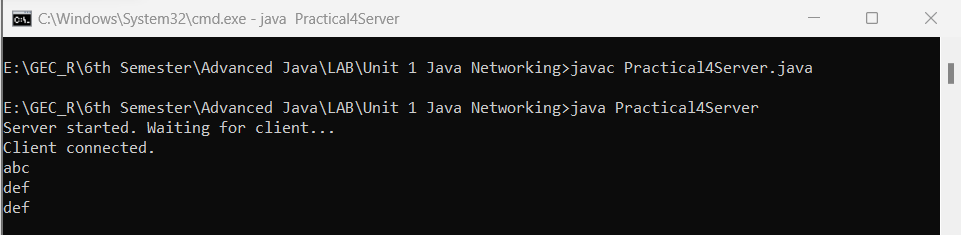
e.printStackTrace();

}

}

}





**Practical 1.5**

**Write java program to send hello message to the server using UDP.**

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class **Practical5Client** {

public static void main(String[] args) {

String message = "Hello";

byte[] buffer = message.getBytes();

int serverPort = 12345;

try (DatagramSocket socket = new DatagramSocket()) {

InetAddress serverAddress = InetAddress.getByName("localhost");

DatagramPacket packet = new DatagramPacket(buffer, buffer.length, serverAddress, serverPort);

socket.send(packet);

System.out.println("Sent message: " + message);

}

catch (IOException e) {

System.err.println("Error sending message: " + e.getMessage());

}

}

}

public class **Practical5Server** {

public static void main(String[] args) {

int serverPort = 12345;

byte[] buffer = new byte[1024];

try (DatagramSocket socket = new DatagramSocket(serverPort)) {

System.out.println("Server started and listening on port " + serverPort);

while (true) {

DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

socket.receive(packet);

String message = new String(packet.getData(), 0, packet.getLength());

System.out.println("Received message: " + message);

}

}

catch (SocketException e) {

System.err.println("Error creating socket: " + e.getMessage());

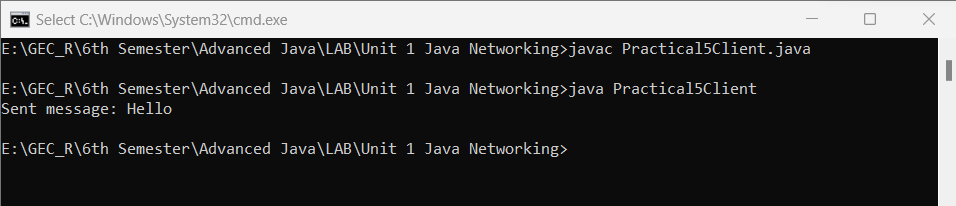
} catch (IOException e) {

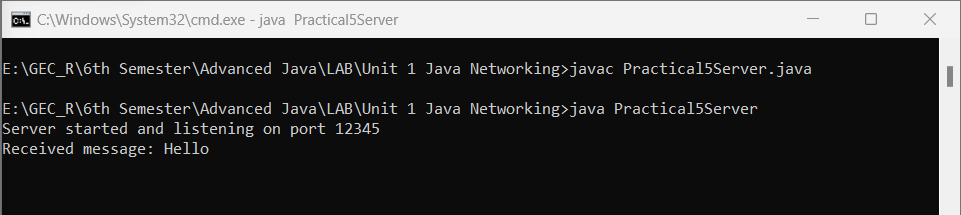
System.err.println("Error receiving message: " + e.getMessage());

}

}

}





***Unit 2 : Java Database Connectivity***

**Practical 2.1**

**Write a java program to insert a student record in Student Table.**

package unit2labpractical;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.Statement;

/\*\*

\* @author Yash

\*/

public class Unit2LABPractical {

/\*\*

\* @param args the command line arguments

\* @param {con} object of Connection class

\* @param {stmt} object of statement class

\* @param {sq} Insert SQL Query

\*/

public static void main(String[] args) {

try {

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

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/advancejavapractical","root","Yash@2233");

Statement stmt = con.createStatement();

String sq = "INSERT INTO student VALUES ('yash', 'ya', 'yash@gmail.com', 'surat',7046985354, 1)";

stmt.executeUpdate(sq);

} catch (ClassNotFoundException ex) {

Logger.getLogger(Unit2LABPractical.class.getName()).log(Level.SEVERE, null, ex);

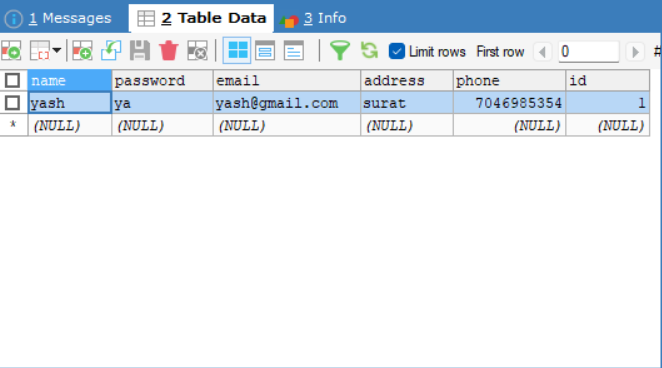
} catch (SQLException ex) {

Logger.getLogger(Unit2LABPractical.class.getName()).log(Level.SEVERE, null, ex);

}

}

}



**Practical 2.2**

**Write a java program to insert a student record in Student Table using Prepared Statement.**

package unit2practical2;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.PreparedStatement;

/\*\*

\* @author Yash

\*/

public class Unit2Practical2 {

/\*\*

\* @param args the command line arguments

\* @param {con} object of Connection class

\* @param {stmt} object of PreparedStatement class

\* @param {sq} Insert SQL Query

\*/

public static void main(String[] args) {

try {

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

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/advancejavapractical","root","Yash@2233");

String sq = "INSERT INTO student (name, password, email, address, phone, id) VALUES (?,?,?,?,?,?)";

PreparedStatement stmt = con.prepareStatement(sq);

stmt.setString(1, "Nishant");

stmt.setString(2, "nishu");

stmt.setString(3, "n@gmail.com");

stmt.setString(4, "Ahemdabad");

stmt.setLong(5, 8401055354L);

stmt.setInt(6, 2);

stmt.executeUpdate();

} catch (ClassNotFoundException ex) {

Logger.getLogger(Unit2Practical2.class.getName()).log(Level.SEVERE, null, ex);

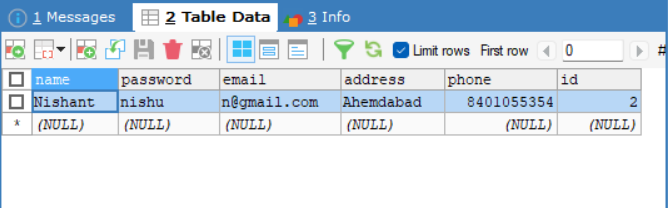
} catch (SQLException ex) {

Logger.getLogger(Unit2Practical2.class.getName()).log(Level.SEVERE, null, ex);

}

}

}



**Practical 2.3**

**Write a java program to call a stored procedure using Callable Statement Interface.**

package unit2practical3;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.CallableStatement;

/\*\*

\* @author Yash

\*/

public class Unit2Practical3 {

/\*\*

\* @param args the command line arguments

\* @param {con} object of Connection class

\* @param {stmt} object of CallableStatement class

\* @param {sql} Calls stored procedure add\_studentlocation

\*/

public static void main(String[] args) {

try {

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

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/advancejavapractical","root","Yash@2233");

String sql = "{CALL add\_studentlocation(?,?)}";

CallableStatement stmt = con.prepareCall(sql);

stmt.setString(1, "Satyam");

stmt.setString(2, "Bhavnagar");

stmt.executeUpdate();

}

catch (ClassNotFoundException ex) {

Logger.getLogger(Unit2Practical3.class.getName()).log(Level.SEVERE, null, ex);

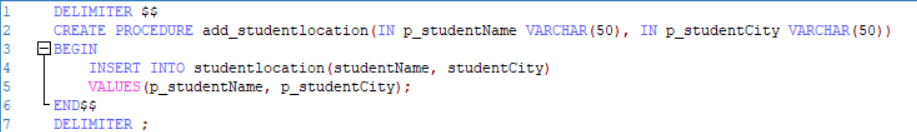
} catch (SQLException ex) {

Logger.getLogger(Unit2Practical3.class.getName()).log(Level.SEVERE, null, ex);

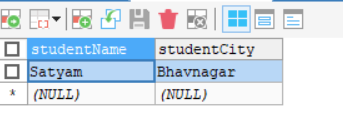
}

}

}



[stored procedure – add\_studentlocation]



***Unit 3 : Servlet***

**Practical 3.1**

**Write a servlet application to print the current date and time.**

import jakarta.servlet.ServletException;

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.util.Date;

/\*\*

\* @author Yash

\*/

public class **Unit3Practical1Servlet** extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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>Currant Date and Time</title>");

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

out.println("<body>");

out.println("<h1>Currant Date and Time</h1>");

out.println("<p>" + new Date() + "</p>");

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

\*/

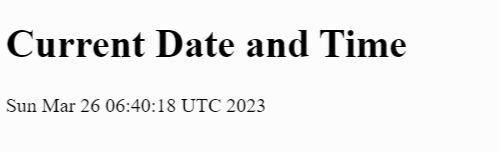
@Override

public String getServletInfo() {

return "Short description";

}// </editor-fold>

}

****

**Practical 3.2**

**Write a Servlet application to count the total number of visits on your website.**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

/\*\*

\* @author Yash

\*/

public class **Unit3Practical2Servlet** extends HttpServlet {

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

HttpSession session = request.getSession(true);

// Get the current visit count from the session

Integer count = (Integer)session.getAttribute("count");

if (count == null) {

count = new Integer(1);

} else {

count = new Integer(count.intValue() + 1);

}

// Store the updated count back in the session

session.setAttribute("count", count);

// Set the response content type

response.setContentType("text/html");

// Write the response

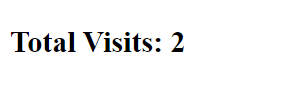
PrintWriter out = response.getWriter();

out.println("<html><head><title>Visit Counter Servlet</title></head>");

out.println("<body><h2>Total Visits: " + count + "</h2></body></html>");

}

}

****

**Practical 3.3**

**Write a Servlet application to authenticate user.**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

/\*\*

\* @author Yash

\*/

public class **Unit3Practical3Servlet** extends HttpServlet {

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

// Get the user's name and password from the request

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

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

// Check if the user is authenticated

if (isAuthenticated(name, password)) {

// If the user is authenticated, create a new session

HttpSession session = request.getSession(true);

// Set the authenticated user's name in the session

session.setAttribute("user", name);

// Redirect the user to a secure page

response.sendRedirect("secure.jsp");

}

else {

// If the user is not authenticated, show an error message

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html><head><title>Login Error</title></head>");

out.println("<body><h2>Login Error</h2>");

out.println("<p>The username or password you entered is incorrect. Please try again.</p>");

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

}

}

private boolean isAuthenticated(String name, String password) {

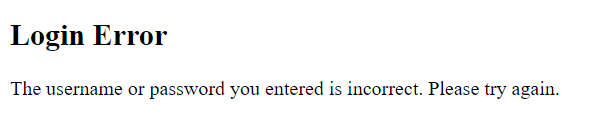
// This method should be implemented to check if the user is authenticated

// For example, you could check a database to see if the user's name and password are valid

return name.equals("myuser") && password.equals("mypassword");

}

**}**

****

**Practical 3.4**

**Write a Servlet application to perform sign up on website.**

**Fields : username, password, Name ,Email and Mobile number**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

/\*\*

\* @author Yash

\*/

public class **Unit3Practical4Servlet** extends HttpServlet {

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

// Get the user's signup information from the request

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

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

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

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

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

// Check if the user already exists

if (userExists(username)) {

// If the user already exists, show an error message

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<html><head><title>Signup Error</title></head>");

out.println("<body><h2>Signup Error</h2>");

out.println("<p>The username you entered is already in use. Please choose another username.</p>");

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

} else {

// If the user does not already exist, add the user to the database

addUser(username, password, name, email, mobile);

// Set the authenticated user's name in the session

HttpSession session = request.getSession(true);

session.setAttribute("user", username);

// Redirect the user to a welcome page

response.sendRedirect("welcome.jsp");

}

}

private boolean userExists(String username) {

// This method should be implemented to check if the user already exists

// For example, you could check a database to see if the username is already in use

// For simplicity, we'll just hard-code a single user with a username

return username.equals("myuser");

}

private void addUser(String username, String password, String name, String email, String mobile) {

// This method should be implemented to add the user to the database

// For example, you could insert the user's information into a database table

// For simplicity, we'll just print the user's information to the console

System.out.println("New user added:");

System.out.println("Username: " + username);

System.out.println("Password: " + password);

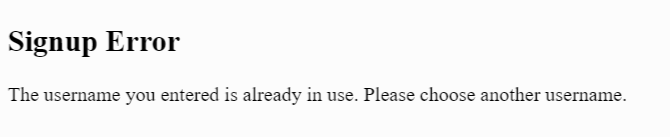
System.out.println("Name: " + name);

System.out.println("Email: " + email);

System.out.println("Mobile: " + mobile);

}

}

****

**Practical 3.5**

**Write a Servlet application to fetch Mark sheet of a given enrollment number.**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.sql.\*;

/\*\*

\* @author Yash

\*/

public class **Unit3Practical5Servlet** extends HttpServlet {

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

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String enrollmentNumber = request.getParameter(101);

try {

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

Connection con = DriverManager.getConnection(jdbc:mysql://localhost:3306/advancejavapractical","root","Yash@2233);

PreparedStatement ps = con.prepareStatement("select \* from marksheet where enrollment\_number=?");

ps.setString(1,enrollmentNumber);

ResultSet rs = ps.executeQuery();

if(rs.next()){

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

out.println("<h2>Mark Sheet of "+rs.getString("name")+"</h2>");

out.println("<table border='1'>");

out.println("<tr><td>Subject1 Marks:</td><td>"+rs.getInt("MATHS")+"</td></tr>");

out.println("<tr><td>Subject2 Marks:</td><td>"+rs.getInt("PHYSICS")+"</td></tr>");

out.println("<tr><td>Subject3 Marks:</td><td>"+rs.getInt("SCIENCE")+"</td></tr>");

out.println("<tr><td>Total Marks:</td><td>"+rs.getInt("total\_marks")+"</td></tr>");

out.println("</table>");

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

}else{

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

out.println("<h2>No Record Found for Enrollment Number "+enrollmentNumber+"</h2>");

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

}

} catch (Exception e) {

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

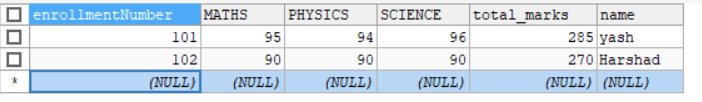
out.println("<h2>Error Occurred:"+e+"</h2>");

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

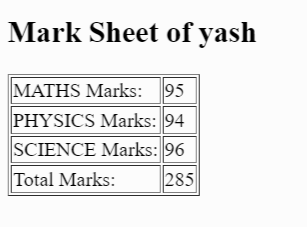
}

}

}



[Marksheet Table]



**Practical 3.6**

**To develop a web application using servlet event handling and filters.**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

/\*\*

 \* @author Yash

 \*/

public class FlterServlet extends HttpServlet {

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

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        out.println("<br>Welcome to Servlet <br>");

    }

}

import java.io.\*;

import javax.servlet.\*;

/\*\*

 \* @author Admin

 \*/

public class Filter1 implements Filter {

    public void init(FilterConfig arg0) throws ServletException{}

    public void doFilter(ServletRequest request,ServletResponse response,FilterChain chain) throws IOException, ServletException

    {

        PrintWriter out = response.getWriter();

        out.print("Filter is invoked before");

        chain.doFilter(request, response);

        out.print("Filter is invoked after");

    }

    public void destroy(){}

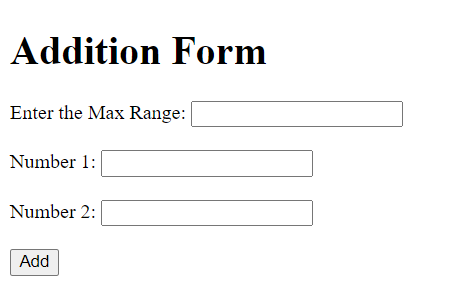
}

**Practical 3.7**

**Develop a web application that demonstrates the use of session level events.**

**Implement the servlet that adds two numbers. Use the two filters in a chain mapped to servlet such that first filter checks the number for valid format and second checks them for a range. If both conditions are fulfilled then only servlet perform the operation and displays the result, otherwise appropriate message from respective filter.**

|  |  |
| --- | --- |
|  | <!DOCTYPE html> |
|  |
|  |
|  |
|  |
|  | <html> |
|  | <head> |
|  | <title>Addition Form</title> |
|  | </head> |
|  | <body> |
|  | <h1>Addition Form</h1> |
|  | <form action="addition" method="post"> |
|  |  |
|  | <label for="range">Enter the Max Range:</label> |
|  | <input type="text" id="range" name="range" required><br><br> |
|  |  |
|  | <label for="num1">Number 1:</label> |
|  | <input type="text" id="num1" name="num1" required><br><br> |
|  |  |
|  | <label for="num2">Number 2:</label> |
|  | <input type="text" id="num2" name="num2" required><br><br> |
|  |  |
|  | <input type="submit" value="Add"> |
|  | </form> |
|  | </body> |
|  | </html> |
|  |  |



import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.\*;

@WebServlet("/addition")

public class AdditionServlet extends HttpServlet {

    protected void doPost(HttpServletRequest request, HttpServletResponse response)

            throws ServletException, IOException {

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

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

        int result = num1 + num2;

        response.getWriter().println("Result: " + result);

    }

}

import java.io.IOException;

import javax.servlet.\*;

public class NumberFormatFilter implements Filter {

    public void init(FilterConfig config) throws ServletException {

    }

    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)

            throws IOException, ServletException {

        try {

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

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

            // If the numbers are in a valid format, proceed to the next filter or servlet

            chain.doFilter(request, response);

        } catch (NumberFormatException e) {

            // Numbers are not in a valid format, send an error message

            response.getWriter().println("Invalid number format");

        }

    }

    public void destroy() {

    }

}

import java.io.IOException;

import javax.servlet.\*;

public class NumberRangeFilter implements Filter {

    public void init(FilterConfig config) throws ServletException {

    }

    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)

            throws IOException, ServletException {

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

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

        // Define your desired number range here

        int min = 0;

        int max = Integer.parseInt(request.getParameter("range"));

        if (num1 >= min && num1 <= max && num2 >= min && num2 <= max) {

            // Numbers are within the range, proceed to the servlet

            chain.doFilter(request, response);

        } else {

            // Numbers are not within the range, send an error message

            response.getWriter().println("Number out of range");

        }

    }

    public void destroy() {

    }

}

***Unit 4 : Java Server Page***

**Practical 4.1**

**To develop a web application using JSP and JDBC.**

**Write two HTML pages named as register.html and login.html.**

**Where, “register.html‟ has one UI-form, which collects user’s details likefull name, age, mobile number, email address,password and confirm password. This register.html page will submit data to servlet named as “RegisterServlet.java”. RegisterServlet will insert user‟s details to database and on successful insertion, it will redirect user tologin.html with appropriate message otherwise show appropriate error message.**

**“login.html” asks either mobile number OR email address and password details for login.**

**For valid user, redirect him/her to welcome.jsp page and also start new session. In welcome.jsp,**

**1. Check whether user is eligible for Vote or not using session data and display appropriate message.**

**2. Display logout button, so that once user will click on logout, it will close the user's session and redirect to login.html.**

**For invalid user, display appropriate message like at login page**

**1. Invalid mobile Number/email address or password. OR**

**2. Mobile Number/email address doesn't exist.**

**Index.html**

<!DOCTYPE html>

<html>

<head>

<title>Unit4 Practical1</title>

<meta charset="UTF-8">

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

</head>

<body>

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

 </body>

</html>

**Login.html**

<html>

<head>

<title>Practical 4.1</title>

<meta charset="UTF-8">

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

</head>

<body>

<div>

<h1>LOGIN</h1>

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

<input type="email" name="email" id="email" placeholder="E-MAIL"><br>

<input type="password" name="pwd" id="pwd" placeholder="PASSWORD"><br><br>

<input type="submit" name="btn" id="btn" class="sbt" value="SUBMIT">

</form>

<p>Don't Have an Account ?<a href="register.html">Create an Account</a></p>

</div>

 </body>

</html>

**Register.html**

<html>

<head>

<title>Practical 4.1</title>

<meta charset="UTF-8">

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

</head>

<body>

<div >

<h1>REGISTER NOW</h1>

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

<input type="text" name="name" id="name" placeholder="NAME"><br>

<input type="email" name="email" id="email" placeholder="E-MAIL"><br>

<input type="tel" name="phone" id="phone" placeholder="PHONE"><br>

<input type="number" name="age" id="age" placeholder="AGE"><br>

<input type="text" name="pwd" id="pwd" placeholder="PASSWORD"><br>

<input type="text" name="cpwd" id="cpwd" placeholder="CONFIRM PASSWORD"><br><br>

<input type="submit" class="sbt" value="SUBMIT">

</form>

<p>Already Registered. <a href="login.html">Login</a></p>

</div>

 </body>

</html>

**Welcome.jsp**

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

<%@ page import = "java.io.\*,java.util.\*,java.sql.\*"%>

<%@ page import = "jakarta.servlet.http.\*,jakarta.servlet.\*" %>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Practical 4.1</title>

</head>

<body>

<%

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

int age = 0;

String name ="";

try {

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

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/ advancejavapractical ", "root", "Yash@2003");

PreparedStatement sa = con.prepareStatement("SELECT age FROM register WHERE email = ?;");

sa.setString(1, yash@mail.com);

ResultSet rsa = sa.executeQuery();

if (rsa.next()) {

age = rsa.getInt(1);

}

PreparedStatement sn = con.prepareStatement("SELECT name FROM register WHERE email = ?;");

sn.setString(1, yash@mail.com);

ResultSet rsn = sn.executeQuery();

if (rsn.next()) {

name = new String(rsn.getString(1));

}

if(age >= 18){

%>

<h1><%=name%> is eligible for Vote</h1>

<%

}

else{

%>

<h1><%=name%> is not eligible for Vote</h1>

<%

}

} catch (Exception e) {

out.println("sql exception occured" + e);

}

%>

<a href="login.html">Log Out</a>

 </body>

</html>

**LoginServlet.java**

import java.io.\*;

import jakarta.servlet.\*;

import jakarta.servlet.http.\*;

import java.sql.\*;

public class loginServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

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

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

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

try {

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

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/ advancejavapractical ", "root", "Yash@2003");

PreparedStatement ps = con.prepareStatement("SELECT \* FROM register where email=? and pwd=?");

ps.setString(1, email);

ps.setString(2, pwd);

ResultSet rs = ps.executeQuery();

if (rs.next()) {

response.sendRedirect("welcome.jsp?email="+email);

} else {

out.print("Invalid email address or password.!");

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

rd.include(request, response);

}

rs.close();

ps.close();

con.close();

} catch (ClassNotFoundException e) {

out.println(e);

} catch (SQLException e) {

out.println("sql exception occured");

}

}

}

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

@Override

public String getServletInfo() {

return "Short description";

}

}

**RegisterServlet.java**

import java.io.\*;

import jakarta.servlet.\*;

import jakarta.servlet.http.\*;

import java.sql.\*;

public class registerServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

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

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

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

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

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

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

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

if (pwd.equals(cpwd) && name != "" && email != "" && phone != "" && age != "") {

try {

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

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/ advancejavapractical ", "root", "Yash@2003");

out.println("hello");

String q = "INSERT INTO register VALUES (?,?,?,?,?)";

PreparedStatement stmt = con.prepareStatement(q);

stmt.setString(1, name);

stmt.setString(2, email);

stmt.setLong(3, Long.parseLong(phone));

stmt.setInt(4, Integer.parseInt(age));

stmt.setString(5, pwd);

stmt.executeUpdate();

con.close();

response.sendRedirect("login.html");

} catch (Exception e) {

out.println("sql exception occured" + e);

}

} else {

out.print("Enter Valid Information!");

RequestDispatcher rd = request.getRequestDispatcher("register.html");

rd.include(request, response);

}

}

}

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

@Override

public String getServletInfo() {

return "Short description";

}

}

**Practical 4.2**

**Write a JSP page to demonstrate various tag of SQL tag library.**

<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>

<a href="newjsp.jsp">JSP</a>

 </body>

</html>

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

<%@ page import = "java.io.\*,java.util.\*,java.sql.\*"%>

<%@ page import = "jakarta.servlet.http.\*,jakarta.servlet.\*" %>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix = "c"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix = "sql"%>

<sql:setDataSource var="db" driver="com.mysql.jdbc.Driver"

url="jdbc:mysql://localhost:3306/ advancejavapractical "

user="root" password="Yash@2003"/>

<sql:query dataSource="${db}" var="rs">

SELECT \* from register;

</sql:query>

<sql:transaction dataSource="${db}">

<sql:update var="insert">

INSERT INTO register VALUES ("abc","abc@gmail.com",123,10,"Abc@123");

</sql:update>

<sql:update var="insert">

INSERT INTO register VALUES ("xyz","xyz@gmail.com",789,21,"Xyz@123");

</sql:update>

</sql:transaction>

<c:set var="name" value="xyz"/>

<sql:update dataSource="${db}" var="update">

DELETE FROM register WHERE name = ?

<sql:param value="${name}" />

</sql:update>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>JSP Page</title>

</head>

<body>

<table border="2" width="auto">

<tr>

<th>Name</th>

<th>Email</th>

<th>Phone</th>

<th>Age</th>

<th>Password</th>

</tr>

<c:forEach var="table" items="${rs.rows}">

<tr>

<td><c:out value="${table.name}"/></td>

<td><c:out value="${table.email}"/></td>

<td><c:out value="${table.phone}"/></td>

<td><c:out value="${table.age}"/></td>

<td><c:out value="${table.pwd}"/></td>

</tr>

</c:forEach>

</table>

 </body>

</html>

***Unit 5 : Java Server Faces***

**Practical 5.1**

**Write a program to demonstrate the use of JSF Convertor Tag and Validation Tag.**

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://java.sun.com/jsf/html"

xmlns:f="http://java.sun.com/jsf/core">

<h:head>

<title>Age Validation</title>

</h:head>

<h:body>

<h:form>

<h:outputLabel for="ageInput" value="Enter your age:"/>

<h:inputText id="ageInput" value="#{ageBean.age}" required="true">

<f:convertNumber/>

</h:inputText>

<h:message for="ageInput" style="color:red"/>

<br/>

<h:commandButton value="Submit" action="#{ageBean.submit}"/>

</h:form>

</h:body>

</html>

import javax.faces.bean.ManagedBean;

import javax.faces.bean.RequestScoped;

@ManagedBean

@RequestScoped

public class AgeBean {

private Integer age;

public Integer getAge() {

return age;

}

public void setAge(Integer age) {

this.age = age;

}

public String submit() {

if (age >= 18) {

return "adult";

} else {

return "minor";

}

}

}

**Practical 5.2**

**Write a program that demonstrates the use of JSF Event Handling and Database Access.**

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://java.sun.com/jsf/html"

xmlns:f="http://java.sun.com/jsf/core">

<h:head>

<title>Customer Management</title>

</h:head>

<h:body>

<h:form>

<h:outputLabel for="nameInput" value="Name:"/>

<h:inputText id="nameInput" value="#{customerBean.name}" required="true"/>

<br/>

<h:outputLabel for="emailInput" value="Email:"/>

<h:inputText id="emailInput" value="#{customerBean.email}" required="true"/>

<br/>

<h:commandButton value="Add Customer" action="#{customerBean.addCustomer}">

<f:ajax execute="@form" render="customerTable"/>

</h:commandButton>

</h:form>

<h:dataTable id="customerTable" value="#{customerBean.customers}" var="customer">

<h:column>

<f:facet name="header">Name</f:facet>

#{customer.name}

</h:column>

<h:column>

<f:facet name="header">Email</f:facet>

#{customer.email}

</h:column>

</h:dataTable>

</h:body>

</html>

import javax.faces.bean.ManagedBean;

import javax.faces.bean.ViewScoped;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.List;

@ManagedBean

@ViewScoped

public class CustomerBean implements Serializable {

private String name;

private String email;

private List<Customer> customers;

public CustomerBean() {

customers = new ArrayList<>();

}

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 List<Customer> getCustomers() {

return customers;

}

public void addCustomer() {

Customer customer = new Customer(name, email);

customers.add(customer);

name = null;

email = null;

}

}

***Unit 6 : Hibernate***

**Practical 6.1**

**Write a hibernate application to save object of Employee class into database.**

**Employee class contains following properties.**

**• EID**

**• ENAME**

**• ESalary**

**Write SQL query for table as well.**

public class Person

{

int id;

String name;

public Person()

{

}

public Person(int id, String name) {

super();

this.id = id;

this.name = name;

}

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;

}

}

import org.hibernate.SessionFactory;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

import org.hibernate.cfg.Configuration;

import org.hibernate.boot.registry.StandardServiceRegistry;

public class HibernateUtil {

    private static final SessionFactory sessionFactory;

    static {

            try {

                Configuration cfg = new Configuration().configure("hibernate.cfg.xml");

                StandardServiceRegistryBuilder sb = new StandardServiceRegistryBuilder();

                sb.applySettings(cfg.getProperties());

                StandardServiceRegistry standardServiceRegistry = sb.build();

                sessionFactory = cfg.buildSessionFactory(standardServiceRegistry);

            } catch (Throwable th) {

                    System.err.println("Enitial SessionFactory creation failed" + th);

                    throw new ExceptionInInitializerError(th);

            }

    }

    public static SessionFactory getSessionFactory() {

            return sessionFactory;

    }

}

CREATE TABLE employees (

eid BIGINT PRIMARY KEY AUTO\_INCREMENT,

ename VARCHAR(255) NOT NULL,

eSalary DOUBLE

);

**Practical 6.2**

**Write a menu based program to perform following operations using HQL (consider above class and table).**

**a. Insert new record**

**b. Update Existing Record**

**c. Delete Record**

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.query.Query;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

Session session = HibernateUtil.getSessionFactory().openSession();

boolean exit = false;

while (!exit) {

System.out.println("Please select an option:");

System.out.println("1. Insert new record");

System.out.println("2. Update existing record");

System.out.println("3. Delete record");

System.out.println("4. Exit");

int option = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

switch (option) {

case 1:

insertRecord(session, scanner);

break;

case 2:

updateRecord(session, scanner);

break;

case 3:

deleteRecord(session, scanner);

break;

case 4:

exit = true;

break;

default:

System.out.println("Invalid option. Please try again.");

}

}

session.close();

HibernateUtil.getSessionFactory().close();

}

private static void insertRecord(Session session, Scanner scanner) {

System.out.println("Enter the name of the employee:");

String ename = scanner.nextLine();

System.out.println("Enter the salary of the employee:");

double eSalary = scanner.nextDouble();

scanner.nextLine(); // Consume the newline character

Employee employee = new Employee();

employee.setEname(ename);

employee.setESalary(eSalary);

Transaction transaction = null;

try {

transaction = session.beginTransaction();

session.save(employee);

transaction.commit();

System.out.println("Employee record inserted successfully.");

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

}

private static void updateRecord(Session session, Scanner scanner) {

System.out.println("Enter the ID of the employee to update:");

long eid = scanner.nextLong();

scanner.nextLine(); // Consume the newline character

Employee employee = session.get(Employee.class, eid);

if (employee != null) {

System.out.println("Enter the new name of the employee:");

String ename = scanner.nextLine();

System.out.println("Enter the new salary of the employee:");

double eSalary = scanner.nextDouble();

scanner.nextLine(); // Consume the newline character

employee.setEname(ename);

employee.setESalary(eSalary);

Transaction transaction = null;

try {

transaction = session.beginTransaction();

session.update(employee);

transaction.commit();

System.out.println("Employee record updated successfully.");

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

} else {

System.out.println("Employee with ID " + eid + " does not exist.");

}

}

private static void deleteRecord(Session session, Scanner scanner) {

System.out.println("Enter the ID of the employee to delete:");

long eid = scanner.nextLong();

scanner.nextLine(); // Consume the newline character

Employee employee = session.get(Employee.class, eid);

if (employee != null) {

Transaction transaction = null;

try {

transaction = session.beginTransaction();

session.delete(employee);

transaction.commit();

System.out.println("Employee record deleted successfully.");

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

} else {

System.out.println("Employee with ID " + eid + " does not exist.");

}

}

}

***Unit 7 : Spring MVC – Java Web Framework***

**Practical 7.1**

**Create a MVC Spring Application which demonstrates Dependency Injection using constructor Consider an Employee (E-id, E\_name) class. Write all necessary classes/files.**

public class Employee {

private int id;

private String name;

public Employee(int id, String name) {

this.id = id;

this.name = name;

}

// Getters and setters

}

public interface EmployeeService {

void addEmployee(Employee employee);

List<Employee> getAllEmployees();

}

import java.util.ArrayList;

import java.util.List;

public class EmployeeServiceImpl implements EmployeeService {

private List<Employee> employees;

public EmployeeServiceImpl() {

employees = new ArrayList<>();

}

@Override

public void addEmployee(Employee employee) {

employees.add(employee);

}

@Override

public List<Employee> getAllEmployees() {

return employees;

}

}

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

@Controller

public class EmployeeController {

private final EmployeeService employeeService;

public EmployeeController(EmployeeService employeeService) {

this.employeeService = employeeService;

}

@GetMapping("/employees")

public String getAllEmployees(Model model) {

model.addAttribute("employees", employeeService.getAllEmployees());

return "employee-list";

}

}

<!DOCTYPE html>

<html>

<head>

<title>Employee List</title>

</head>

<body>

<h1>Employee List</h1>

<table>

<tr>

<th>ID</th>

<th>Name</th>

</tr>

<c:forEach items="${employees}" var="employee">

<tr>

<td>${employee.id}</td <td>${employee.name}</td>

</tr>

</c:forEach>

</table>

</body>

</html>

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MainApp {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("spring-config.xml");

EmployeeService employeeService = context.getBean("employeeService", EmployeeService.class);

employeeService.addEmployee(new Employee(1, "John Doe"));

employeeService.addEmployee(new Employee(2, "Jane Smith"));

EmployeeController employeeController = new EmployeeController(employeeService);

employeeController.getAllEmployees();

((ClassPathXmlApplicationContext) context).close();

}

}