ENTERPRISE JAVA JOURNAL

TABLE OF CONTENTS

7.	a. Develop simple EJB application to demonstrate Servlet Hit count using Singleton Session Beans.	15/09/25	
	b. Develop simple visitor Statistics application using Message Driven Bean [Stateless Session Bean].		
	c. Develop simple Marks Entry Application to demonstrate accessing Database using EJB.		
8.	a. Develop a simple Inventory Application Using JPA.	24/09/25	
	b. Develop a Guestbook Application Using JPA.		
	c. Create simple JPA application to store and retrieve Book details.		
9.	a. Develop a JPA Application to demonstrate use of ORM associations.	06/10/25	
	b. Develop a Hibernate application to store Feedback of Website Visitor in MySQL Database.		
	c. Develop a Hibernate application to store and retrieve employee details in MySQL Database.		
10.	a. Develop an application to demonstrate Hibernate One-To -One Mapping Using Annotation.	13/10/25	
	b. Develop Hibernate application to enter and retrieve course details with ORM Mapping.		
	c. Develop a five page web application site using any two or three Java EE Technologies.		

PRACTICAL NO. 1

Implement the following Simple Servlet applications.

a. Create a simple calculator application using servlet.

index.jsp

```
<!DOCTYPE html>
<html>
<head>
   <title>Simple Calculator</title>
</head>
<body>
   <h2>Simple Calculator</h2>
   <form action="calculate" method="post">
      Number 1: <input type="text" name="num1" required><br><br>
      Number 2: <input type="text" name="num2" required><br><br>
      Operation:
      <select name="operation">
         <option value="add">Add (+)</option>
         <option value="subtract">Subtract (-)</option>
         <option value="multiply">Multiply (*)</option>
         <option value="divide">Divide (/)</option>
      </select><br><br>
      <input type="submit" value="Calculate">
   </form>

   <h3>
      <% if (request.getAttribute("result") != null) { %>
         Result: <%= request.getAttribute("result") %>
      <% } else if (request.getAttribute("error") != null) { %>
         <span style="color:red;"><%= request.getAttribute("error") %></span>
      <% } %>
```

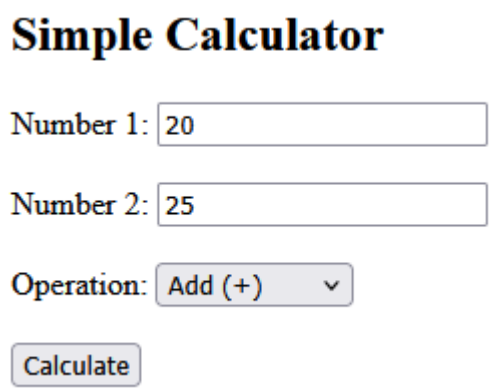
```
    </h3>
</body>
</html>
```

CalculatorServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class CalculatorServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        double num1 = Double.parseDouble(request.getParameter("num1"));
        double num2 = Double.parseDouble(request.getParameter("num2"));
        String operation = request.getParameter("operation");
        double result = 0;

        switch (operation) {
            case "add": result = num1 + num2; break;
            case "subtract": result = num1 - num2; break;
            case "multiply": result = num1 * num2; break;
            case "divide":
                if (num2 != 0) result = num1 / num2;
                else request.setAttribute("error", "Cannot divide by zero.");
                break;
        }

        request.setAttribute("result", result);
        RequestDispatcher dispatcher = request.getRequestDispatcher("index.jsp");
        dispatcher.forward(request, response);
```

```
    }
}
```

Output:



---

b. Create a servlet for a login page. If the username and password are correct then it says message "Hello" else a message "login failed"

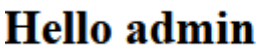
index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Login Page</title>
</head>
<body>
    <h2>Login Form</h2>
    <form action="LoginServlet" method="post">
        Username: <input type="text" name="username" /><br><br>
        Password: <input type="password" name="password" /><br><br>
        <input type="submit" value="Login" />
    </form>
</body>
</html>
```

LoginServlet.java

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

@WebServlet("/LoginServlet")  // This replaces web.xml configuration
public class LoginServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

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

        if ("admin".equals(username) && "admin123".equals(password)) {
            out.println("<h2>Hello " + username + "</h2>");
        } else {
            out.println("<h2>Login failed</h2>");
        }

        out.close();
    }
}
```

Output:



c. Create a registration servlet in Java using JDBC. Accept the details such as Username, Password, Email, and Country from the user using HTML Form and store the registration details in the database.

Prerequisites:

1. Download the mysql-connector-j-9.4.0.jar file and add the jar file to your project bu following these steps -> Go to your project -> right click and go to properties -> select the option libraries -> click Add jar file -> select the jar file from your local machine (Go to the path where you have kept the downloaded jar file in your system) and click ok.
2. Create a database and a table as follows:

```
CREATE DATABASE userdb;
USE userdb;
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(50) NOT NULL,
    email VARCHAR(100),
    country VARCHAR(50)
);
```

index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>User Registration</title>
</head>
<body>
    <h2>Register Here</h2>
```

```
    <form action="RegisterServlet" method="post">
        Username: <input type="text" name="username" required><br><br>
        Password: <input type="password" name="password" required><br><br>
        Email: <input type="email" name="email" required><br><br>
        Country: <input type="text" name="country" required><br><br>
        <input type="submit" value="Register">
    </form>
</body>
</html>
```

RegisterServlet.java

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

@WebServlet("/RegisterServlet")
public class RegisterServlet extends HttpServlet {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/userdb";
    private static final String DB_USER = "root"; // your DB username
    private static final String DB_PASS = "";     // your DB password

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

        // Get form values
        String username = request.getParameter("username");
        String password = request.getParameter("password");
        String email = request.getParameter("email");
        String country = request.getParameter("country");
```

```
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        try {
            // Load JDBC Driver
            Class.forName("com.mysql.cj.jdbc.Driver");

            // Connect to DB
            Connection conn = DriverManager.getConnection(
    "jdbc:mysql://localhost:3306/userdb", "root", "root");



            // Insert query
            String sql = "INSERT INTO users (username, password, email, country) VALUES (?,
?, ?, ?)";
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, username);
            stmt.setString(2, password);
            stmt.setString(3, email);
            stmt.setString(4, country);

            int rows = stmt.executeUpdate();

            if (rows > 0) {
                out.println("<h2>Registration successful!</h2>");
            } else {
                out.println("<h2>Registration failed.</h2>");
            }

            conn.close();
```

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

Output:

PRACTICAL NO. 2

Implement the following Servlet applications with Cookies and Sessions.

a. Using Request Dispatcher Interface create a Servlet which will validate the password entered by the user, if the user has entered "Servlet" as password, then he will be forwarded to Welcome Servlet else the user will stay on the index.html page and an error message will be displayed.

Note: Keep the **WelcomeServlet.java** in the default package and **ValidateServlet.java** in the **ValidateServlet_**package.

index.jsp

```
<!DOCTYPE html>
<html>
<head>
   <title>Password Validation</title>
</head>
<body>
   <h2>Login Page</h2>
   <form action="ValidateServlet" method="post">
      Enter Password: <input type="password" name="password" />
      <input type="submit" value="Login" />
   </form>

   <p style="color:red;">
      ${errorMsg}
   </p>
</body>
</html>
```

WelcomeServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
```

```
public class WelcomeServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        out.println("<h2>Welcome! You have successfully logged in.</h2>");
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        doPost(request, response);
    }
}
```

ValidateServlet/ValidateServlet.java

```
package ValidateServlet;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class ValidateServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

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

        if ("Servlet".equals(password)) {
            // Forward to WelcomeServlet if password is correct
```

```
            RequestDispatcher rd = request.getRequestDispatcher("WelcomeServlet");
            rd.forward(request, response);
        } else {
            // Stay on index.html with error message
            PrintWriter out = response.getWriter();
            RequestDispatcher rd = request.getRequestDispatcher("index.jsp");
            rd.include(request, response);  // include index.html content

            out.println("<center><p style='color:red;'>Invalid Password! Try
Again.</p></center>");
        }
    }
}
```

Output:

Login Page

Enter Password: ••••••• Login

Enter the password as: Servlet

Welcome! You have successfully logged in.

b. Create a servlet that uses Cookies to store the number of times a user has visited servlet.

index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Visit Counter App</title>
</head>
<body>
    <h2>Welcome!</h2>
    <p><a href="VisitCounterServlet">Click here to check visit count</a></p>
```

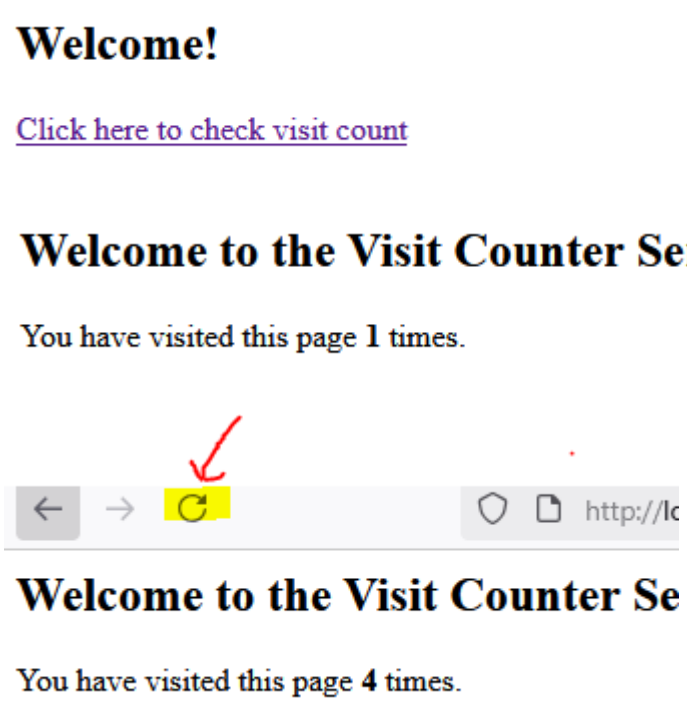
```
</body>
</html>
```

VisitCounterServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class VisitCounterServlet extends HttpServlet {
   protected void doGet(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {

      response.setContentType("text/html");
      PrintWriter out = response.getWriter();

      int visitCount = 0;

      // Get cookies from request
      Cookie[] cookies = request.getCookies();
      if (cookies != null) {
         for (Cookie c : cookies) {
            if (c.getName().equals("visitCount")) {
               // Retrieve the old count
               visitCount = Integer.parseInt(c.getValue());
            }
         }
      }

      // Increment the count
      visitCount++;
```

```
        // Store updated count back into a cookie
        Cookie visitCookie = new Cookie("visitCount", String.valueOf(visitCount));
        visitCookie.setMaxAge(60 * 60 * 24);
        response.addCookie(visitCookie);

        // Display result
        out.println("<html><body>");
        out.println("<h2>Welcome to the Visit Counter Servlet</h2>");
        out.println("<p>You have visited this page <b>" + visitCount + "</b> times.</p>");
        out.println("</body></html>");
    }
}
```

Output:



c. Create a servlet demonstrating the use of session creation and destruction. Also check whether the user has visited this page first time or has visited earlier also using sessions.

index.html

```
<!DOCTYPE html>
<html>
<head>
   <title>Session Demo</title>
</head>
<body>
   <h1>Session Tracking Example</h1>
   <p>Click below to test session creation and visit tracking.</p>

   <!-- Link to servlet -->
   <form action="SessionServlet" method="get">
      <input type="submit" value="Go to Session Servlet">
   </form>
</body>
</html>
```

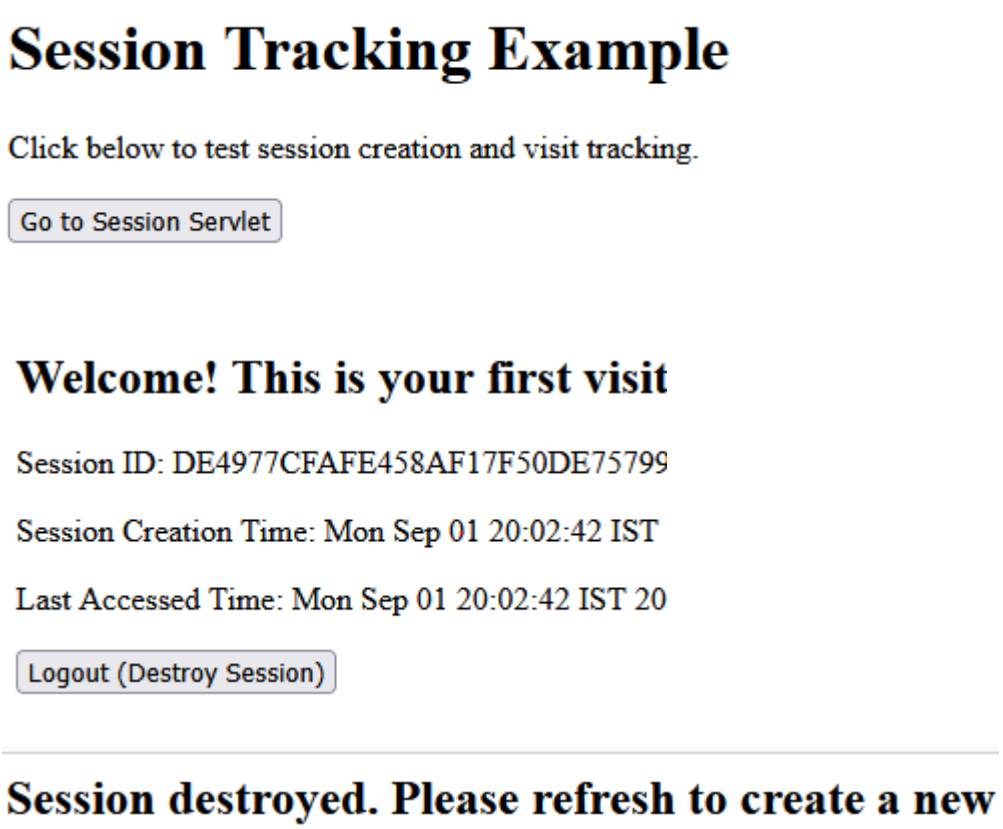
SessionServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class SessionServlet extends HttpServlet {

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

      response.setContentType("text/html");
      PrintWriter out = response.getWriter();
```

```
        // Get or create session
        HttpSession session = request.getSession();

        // Check if it's a new session
        if (session.isNew()) {
            out.println("<h2>Welcome! This is your first visit.</h2>");
            session.setAttribute("visitCount", 1); // start counting visits
        } else {
            Integer count = (Integer) session.getAttribute("visitCount");
            if (count == null) count = 0;
            count++;
            session.setAttribute("visitCount", count);

            out.println("<h2>Welcome back! You have visited this page " + count + "
times.</h2>");
        }

        // Display session details
        out.println("<p>Session ID: " + session.getId() + "</p>");
        out.println("<p>Session Creation Time: " + new
java.util.Date(session.getCreationTime()) + "</p>");
        out.println("<p>Last Accessed Time: " + new
java.util.Date(session.getLastAccessedTime()) + "</p>");

        // Option to destroy the session
        out.println("<form action=" method='post'>");
        out.println("<input type='submit' value='Logout (Destroy Session)'>");
        out.println("</form>");
    }
```

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

        HttpSession session = request.getSession(false); // don't create new if not exists
        if (session != null) {
            session.invalidate(); // destroy session
        }

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<h2>Session destroyed. Please refresh to create a new session.</h2>");
    }
}
```

Output:



PRACTICAL NO. 3

Implement the Servlet IO and File applications.

a. Create a Servlet application to upload and download a file.

index.html

```
<!DOCTYPE html>
<html>
<head>
   <title>File Upload and Download</title>
</head>
<body>
   <h2>Upload File</h2>
   <form action="UploadServlet" method="post" enctype="multipart/form-data">
      <input type="file" name="file"><br><br>
      <input type="submit" value="Upload">
   </form>

   <h2>Download File</h2>
   <form action="DownloadServlet" method="get">
      <input type="text" name="filename" placeholder="Enter file name"><br><br>
      <input type="submit" value="Download">
   </form>
</body>
</html>
```

UploadServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.annotation.MultipartConfig;
import javax.servlet.http.*;

@MultipartConfig
```

```
public class UploadServlet extends HttpServlet {
    private static final String UPLOAD_DIR = "uploads";

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

        Part filePart = request.getPart("file"); // get file
        String fileName = filePart.getSubmittedFileName();

        // Save file to uploads folder inside webapp
        String uploadPath = getServletContext().getRealPath("") + File.separator +
UPLOAD_DIR;
        File uploadDir = new File(uploadPath);
        if (!uploadDir.exists()) uploadDir.mkdir();

        filePart.write(uploadPath + File.separator + fileName);

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<h3>File " + fileName + " uploaded successfully!</h3>");
        out.println("<a href='index.html'>Back</a>");
    }
}
```

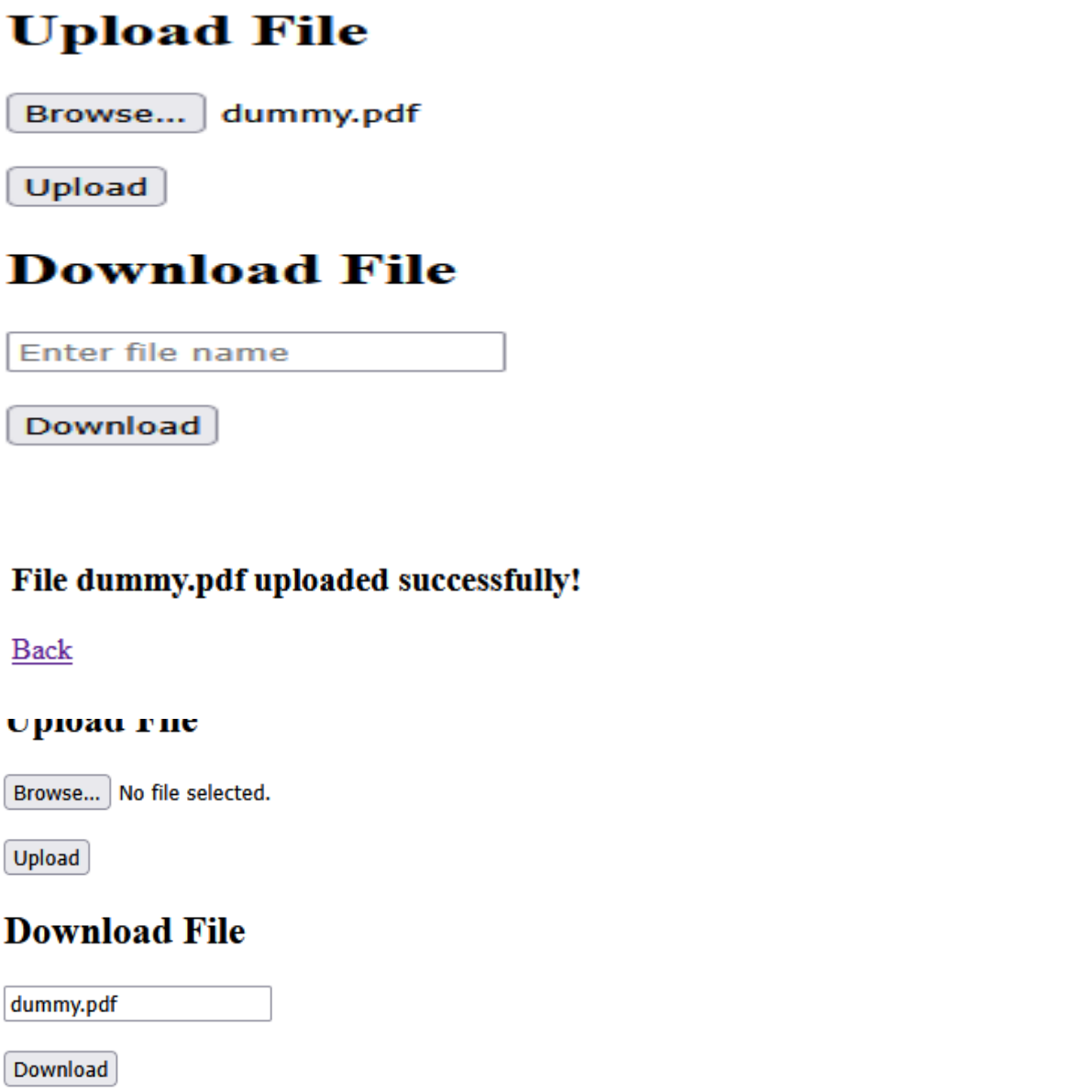
DownloadServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class DownloadServlet extends HttpServlet {
    private static final String UPLOAD_DIR = "uploads";
```

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

        String fileName = request.getParameter("filename");
        String filePath = getServletContext().getRealPath("") + File.separator + UPLOAD_DIR
+ File.separator + fileName;

        File downloadFile = new File(filePath);
        if (!downloadFile.exists()) {
            response.getWriter().println("<h3>File not found: " + fileName + "</h3>");
            return;
        }

        FileInputStream inStream = new FileInputStream(downloadFile);

        // Set MIME type
        String mimeType = getServletContext().getMimeType(filePath);
        if (mimeType == null) {
            mimeType = "application/octet-stream";
        }
        response.setContentType(mimeType);
        response.setContentLength((int) downloadFile.length());

        // Set response header
        response.setHeader("Content-Disposition", "attachment; filename=\"" +
downloadFile.getName() + "\"");

        // Write file to response
        OutputStream outStream = response.getOutputStream();
        byte[] buffer = new byte[4096];
        int bytesRead;
```

```
        while ((bytesRead = inStream.read(buffer)) != -1) {
            outStream.write(buffer, 0, bytesRead);
        }

        inStream.close();
        outStream.close();
    }
}
```

Output:



b. Develop Simple Servlet Question Answer Application using Database.

Prerequisites:

Create a database as follows:

```
CREATE DATABASE qadb;

USE qadb;

CREATE TABLE qa(id INT AUTO_INCREMENT PRIMARY KEY,question
VARCHAR(255),answer VARCHAR(255));
```

index.html

```
<!DOCTYPE html>
<html>
<head><title>Q&A App</title></head>
<body>
  <h2>Add Question & Answer</h2>
  <form action="QAServlet" method="post">
    Question: <input type="text" name="question"><br><br>
    Answer: <input type="text" name="answer"><br><br>
    <input type="submit" value="Save">
  </form>
  <br>
  <a href="QAServlet">View All</a>
</body>
</html>
```

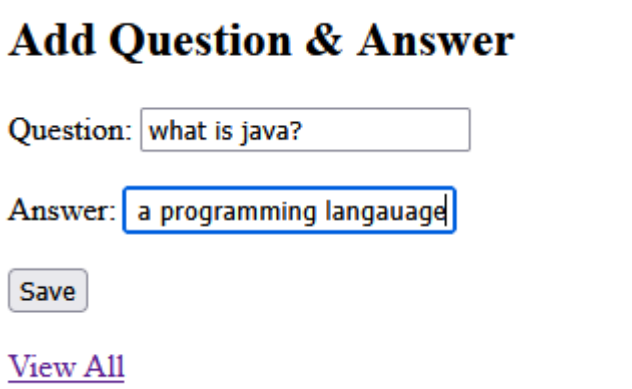
QAServlet.java

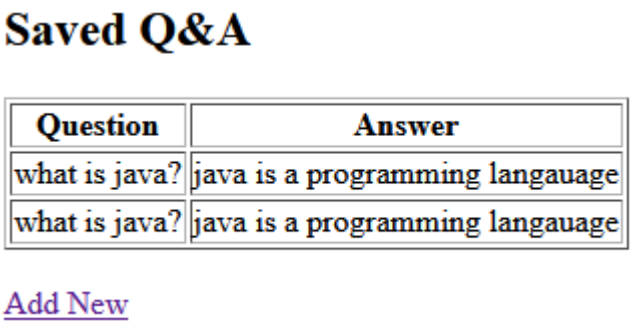
```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;
import java.util.*;

public class QAServlet extends HttpServlet {
```

```
    Connection con;

    public void init() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection(
                "jdbc:mysql://localhost:3306/qadb", "root", "root"); // change pass if needed
        } catch (Exception e) { e.printStackTrace(); }
    }

    protected void doPost(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException {
        String q = req.getParameter("question");
        String a = req.getParameter("answer");
        try {
            PreparedStatement ps = con.prepareStatement(
                "INSERT INTO qa(question, answer) VALUES(?,?)");
            ps.setString(1, q);
            ps.setString(2, a);
            ps.executeUpdate();
        } catch (Exception e) { e.printStackTrace(); }
        res.sendRedirect("QAServlet");
    }

    protected void doGet(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        out.println("<h2>Saved Q&A</h2><table
border=1><tr><th>Question</th><th>Answer</th></tr>");
        try {
```

```
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("SELECT question, answer FROM qa");
            while (rs.next()) {
                out.println("<tr><td>"+rs.getString(1)+"</td><td>"+rs.getString(2)+"</td></tr>");
            }
        } catch (Exception e) { e.printStackTrace(); }
        out.println("</table><br><a href='index.html'>Add New</a>");
    }
}
```

Output:





c. Create simple Servlet application to demonstrate Non-Blocking Read Operation.

index.html

```
<!DOCTYPE html>
<html>
<head>
```

```
    <title>Non-Blocking Read Demo</title>
</head>
<body>
    <h2>Non-Blocking Servlet Read Example</h2>
    <form action="NonBlockingServlet" method="post">
        <textarea name="data" rows="5" cols="40"></textarea><br><br>
        <input type="submit" value="Send Data">
    </form>
</body>
</html>
```

NonBlockingServlet.java

```
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;

@WebServlet(urlPatterns = "/nonBlockingServlet", asyncSupported = true)
public class NonBlockingServlet extends HttpServlet {

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

        // Start async mode
        AsyncContext asyncContext = request.startAsync();
        ServletInputStream input = request.getInputStream();

        input.setReadListener(new ReadListener() {
            private StringBuilder buffer = new StringBuilder();
```

```
@Override
public void onDataAvailable() throws IOException {
    byte[] b = new byte[1024];
    int len;
    while (input.isReady() && (len = input.read(b)) != -1) {
        buffer.append(new String(b, 0, len, StandardCharsets.UTF_8));
    }
}

@Override
public void onAllDataRead() throws IOException {
    HttpServletResponse resp = (HttpServletResponse) asyncContext.getResponse();
    resp.setContentType("text/plain");
    resp.getWriter().write("Data received (Non-Blocking):\n");
    resp.getWriter().write(buffer.toString());
    asyncContext.complete(); // must end async
}

@Override
public void onError(Throwable t) {
    try {
        HttpServletResponse resp = (HttpServletResponse) asyncContext.getResponse();
        resp.getWriter().write("Error: " + t.getMessage());
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        asyncContext.complete();
    }
}
```

```
        });
    }
}
```

Output:


Non-Blocking Servlet Read Example
Hello this is janhavi
Send Data
Data received (Non-Blocking):
data=Hello+this+is+janhavi

PRACTICAL NO. 4

Implement the following JSP applications.

a. Develop a simple JSP application to display values obtained from the use of intrinsic objects of various types.

index.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
  <title>Intrinsic Objects Example</title>
</head>
<body>
  <h2>Enter Your Name</h2>
  <form action="info.jsp" method="post">
    Name: <input type="text" name="uname" />
    <input type="submit" value="Submit" />
  </form>
</body>
</html>
```

info.jsp

```
<%@ page import="java.util.Date" %>
<!DOCTYPE html>
<html>
<head>
  <title>Intrinsic Objects Example</title>
</head>
<body>
  <h2>Values from Intrinsic Objects</h2>
```

```
    <%
        // Request object
        String name = request.getParameter("uname");

        // Session object
        session.setAttribute("username", name);

        // Application object
        application.setAttribute("appMsg", "This message is stored in application scope");

        // Response object - set content type
        response.setContentType("text/html");

        // Out object - printing message
        out.println("<p><b>Welcome, " + name + "</b></p>");
    %>

    <p><b>Request Object:</b> Your name = <%= request.getParameter("uname") %></p>
    <p><b>Session Object:</b> Username stored in session = <%=
session.getAttribute("username") %></p>
    <p><b>Application Object:</b> <%= application.getAttribute("appMsg") %></p>
    <p><b>Response Object:</b> Content type set = <%= response.getContentType() %></p>
    <p><b>Config Object:</b> JSP Page name = <%= config.getServletName() %></p>
    <p><b>Page Object:</b> JSP class = <%= page.getClass().getName() %></p>
    <p><b>PageContext Object:</b> Server info = <%=
pageContext.getServletContext().getServerInfo() %></p>
    <p><b>Out Object:</b> Current time = <%= new Date() %></p>

</body>
</html>
```

Output:

b. Develop a simple JSP application to pass values from one page to another with validations. (Name-txt, age-txt, hobbies-checkbox, email-txt, gender-radio button).

index.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<!DOCTYPE html>
<html>
<head>
    <title>User Form</title>
</head>
<body>
    <h2>Enter Your Details</h2>
    <form action="display.jsp" method="post">
        Name: <input type="text" name="name"><br><br>
```

```
        Age: <input type="text" name="age"><br><br>
        Email: <input type="text" name="email"><br><br>

        Gender:
        <input type="radio" name="gender" value="Male"> Male
        <input type="radio" name="gender" value="Female"> Female <br><br>

        Hobbies:<br>
        <input type="checkbox" name="hobbies" value="Reading"> Reading
        <input type="checkbox" name="hobbies" value="Sports"> Sports
        <input type="checkbox" name="hobbies" value="Music"> Music
        <input type="checkbox" name="hobbies" value="Travel"> Travel
        <br><br>

        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

display.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>
<%@ page import="java.util.*" %>
<!DOCTYPE html>
<html>
<head>
    <title>Display Details</title>
</head>
<body>
<%
    String name = request.getParameter("name");
    String ageStr = request.getParameter("age");
```

```
String email = request.getParameter("email");
String gender = request.getParameter("gender");
String[] hobbies = request.getParameterValues("hobbies");

boolean valid = true;
String errorMsg = "";

// Validation
if(name == null || name.trim().equals("")) {
    valid = false;
    errorMsg += "Name is required.<br>";
}
int age = 0;
try {
    age = Integer.parseInt(ageStr);
    if(age <= 0) {
        valid = false;
        errorMsg += "Age must be greater than 0.<br>";
    }
} catch(Exception e) {
    valid = false;
    errorMsg += "Age must be a number.<br>";
}

if(email == null || !email.matches("^[A-Za-z0-9+_.-]+@(.+)$")) {
    valid = false;
    errorMsg += "Invalid email format.<br>";
}

if(gender == null) {
```

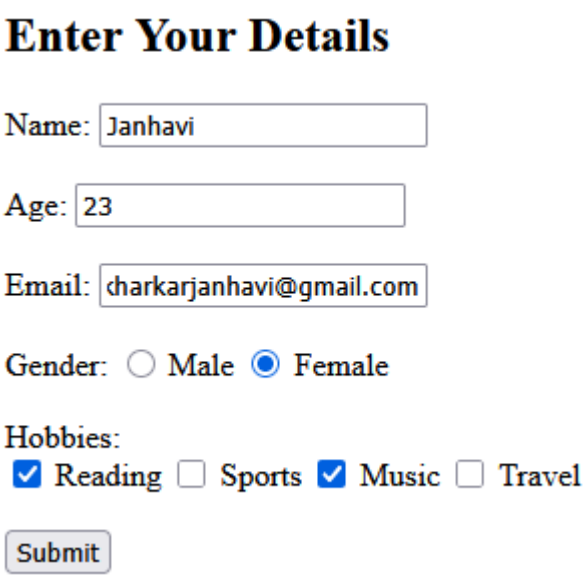
```
        valid = false;
        errorMsg += "Please select gender.<br>";
    }

    if(hobbies == null || hobbies.length == 0) {
        valid = false;
        errorMsg += "Please select at least one hobby.<br>";
    }

    if(valid) {
%>
        <h2>Details Entered</h2>
        <b>Name:</b> <%= name %><br>
        <b>Age:</b> <%= age %><br>
        <b>Email:</b> <%= email %><br>
        <b>Gender:</b> <%= gender %><br>
        <b>Hobbies:</b>
        <ul>
            <% for(String h : hobbies) { %>
                <li><%= h %></li>
            <% } %>
        </ul>
<%
    } else {
%>
        <h2 style="color:red;">Validation Errors</h2>
        <%= errorMsg %>
        <br><a href="index.jsp">Go Back</a>
<%
    }
```

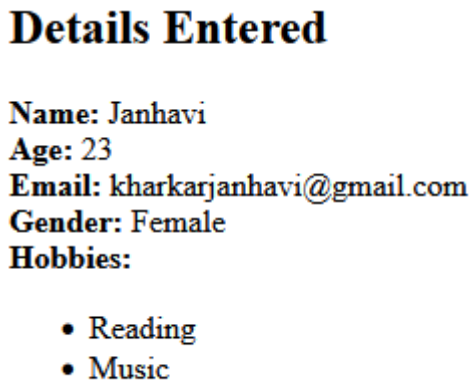
%>

</body>

</html>

Output:





c. Create a registration and login JSP application to register and authenticate the user based on username and password using JDBC.

Prerequisites:

Create a database as follows:

CREATE DATABASE userdb;

USE userdb;

CREATE TABLE users (

id INT AUTO_INCREMENT PRIMARY KEY,

username VARCHAR(50) UNIQUE NOT NULL,

```
    password VARCHAR(100) NOT NULL
);
```

dbconnect.jsp

```
<%@ page import="java.sql.*" %>
<%
    String url = "jdbc:mysql://localhost:3306/userdb";
    String user = "root";     // change as per your MySQL
    String pass = "root";     // change as per your MySQL

    Connection conn = null;
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        conn = DriverManager.getConnection(url, user, pass);
    } catch(Exception e) {
        out.println("Database connection error: " + e);
    }
%>
```

index.jsp

```
<%
    // If user already logged in, send to welcome page
    if (session.getAttribute("username") != null) {
        response.sendRedirect("welcome.jsp");
    } else {
        // Otherwise, send to login page
        response.sendRedirect("register.jsp");
    }
%>
```

login.jsp

```
<%@ include file="dbconnect.jsp" %>
<html>
```

```
<head><title>Login</title></head>
<body>
   <h2>User Login</h2>
   <form action="loginProcess.jsp" method="post">
      Username: <input type="text" name="username" required><br><br>
      Password: <input type="password" name="password" required><br><br>
      <input type="submit" value="Login">
   </form>
</body>
</html>
```

loginProcess.jsp

```
<%@ include file="dbconnect.jsp" %>
<%
   String uname = request.getParameter("username");
   String pwd = request.getParameter("password");

   try {
      PreparedStatement ps = conn.prepareStatement("SELECT * FROM users WHERE
username=? AND password=?");
      ps.setString(1, uname);
      ps.setString(2, pwd);
      ResultSet rs = ps.executeQuery();
      if(rs.next()){
         session.setAttribute("username", uname);
         response.sendRedirect("welcome.jsp");
      } else {
         out.println("<h3>Invalid username or password</h3>");
         out.println("<a href='login.jsp'>Try Again</a>");
      }
   } catch(Exception e) {
      out.println("Error: " + e.getMessage());
```

```
    }
%>
```

logout.jsp

```
<%
    session.invalidate();
    response.sendRedirect("login.jsp");
%>
```

register.jsp

```
<%@ include file="dbconnect.jsp" %>
<html>
<head><title>Register</title></head>
<body>
    <h2>User Registration</h2>
    <form action="registerProcess.jsp" method="post">
        Username: <input type="text" name="username" required><br><br>
        Password: <input type="password" name="password" required><br><br>
        <input type="submit" value="Register">
    </form>
    <p>Already registered? <a href="login.jsp">Login here</a></p>
</body>
</html>
```

registerProcess.jsp

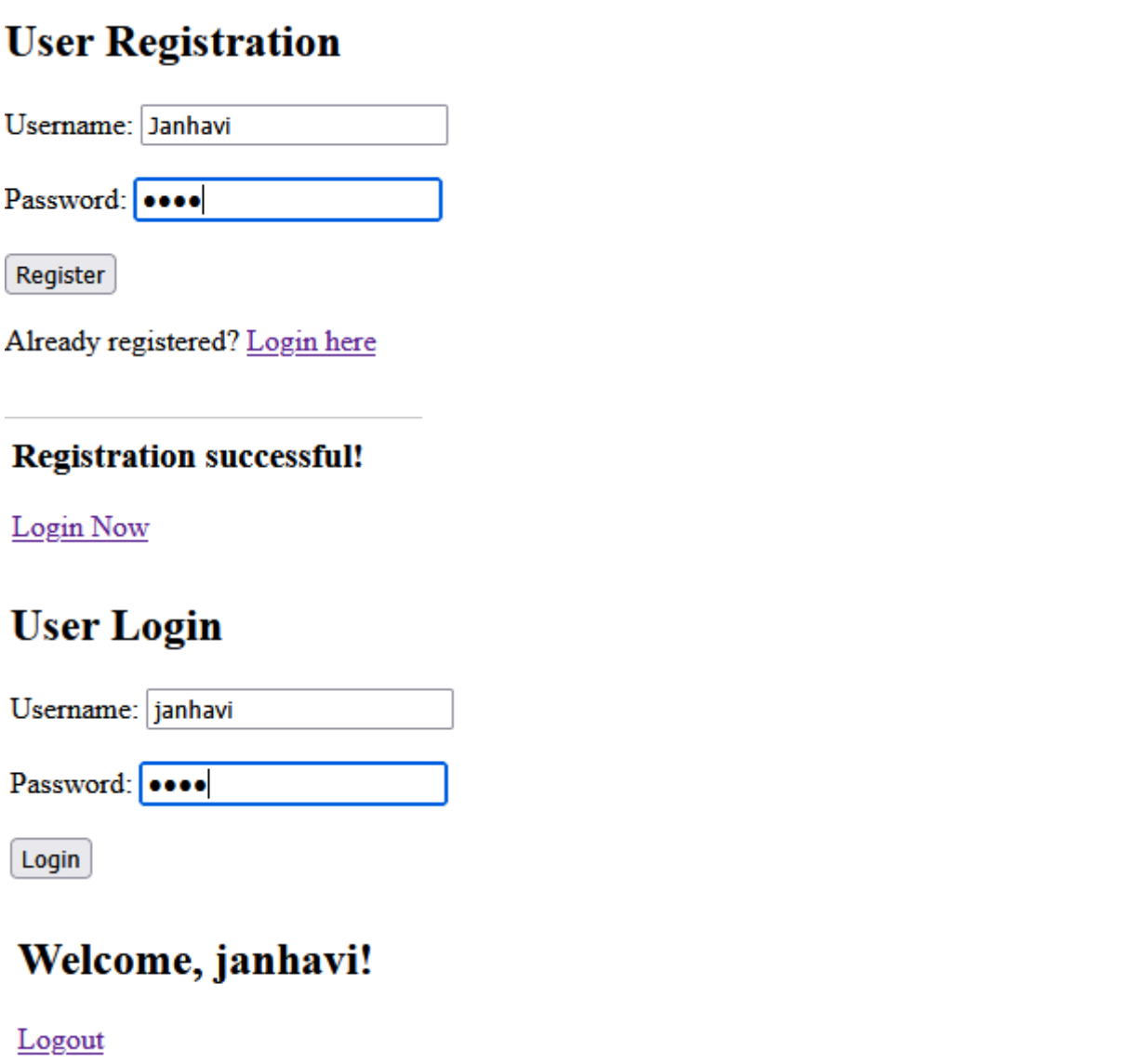
```
<%@ include file="dbconnect.jsp" %>
<%
    String uname = request.getParameter("username");
    String pwd = request.getParameter("password");

    try {
        PreparedStatement ps = conn.prepareStatement("INSERT INTO users(username,
password) VALUES (?, ?)");
        ps.setString(1, uname);
        ps.setString(2, pwd); // In real apps, hash the password
```

```
        int i = ps.executeUpdate();
        if(i > 0){
            out.println("<h3>Registration successful!</h3>");
            out.println("<a href='login.jsp'>Login Now</a>");
        } else {
            out.println("<h3>Registration failed. Try again.</h3>");
        }
    } catch(Exception e) {
        out.println("Error: " + e.getMessage());
    }
%>
```

welcome.jsp

```
<%
    String user = (String)session.getAttribute("username");
    if(user == null){
        response.sendRedirect("login.jsp");
    }
%>
<html>
<head><title>Welcome</title></head>
<body>
    <h2>Welcome, <%= user %>!</h2>
    <a href="logout.jsp">Logout</a>
</body>
</html>
```

Output:


User Registration
Username: Janhavi
Password: ••••
Register
Already registered? Login here
Registration successful!
Login Now
User Login
Username: janhavi
Password: ••••
Login
Welcome, janhavi!
Logout

PRACTICAL NO. 5

Implement the following JSP JSTL and EL Applications.

a. Create an html page with fields, eno, name, age, desg, salary. Now on submit this data to a JSP page which will update the employee table of database with matching eno.

Prerequisites:

1.Add mysql connector jar file in project properties->libraries->Add jar file

2.Create a database as follows:

CREATE DATABASE yourDB;

USE yourDB;

CREATE TABLE employee (eno INT PRIMARY KEY,name VARCHAR(50),age INT,desg VARCHAR(50),salary DOUBLE);

UpdateEmployee.jsp

```
<%@ page import="java.sql.*" %>

<%

    // Fetch form data

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

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

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

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

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


    Connection con = null;

    PreparedStatement ps = null;


    try {

        // Load driver (example for MySQL)

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

        con = DriverManager.getConnection(

            "jdbc:mysql://localhost:3306/yourDB", "root", "root");
```

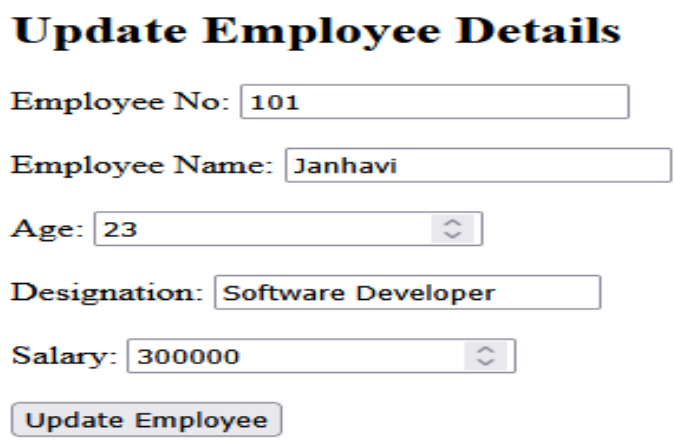
```
        // Update query
        String query = "UPDATE employee SET name=?, age=?, desg=?, salary=? WHERE
eno=?";
        ps = con.prepareStatement(query);
        ps.setString(1, name);
        ps.setInt(2, Integer.parseInt(age));
        ps.setString(3, desg);
        ps.setDouble(4, Double.parseDouble(salary));
        ps.setInt(5, Integer.parseInt(eno));

        int i = ps.executeUpdate();

        if (i > 0) {
            out.println("<h3>Employee record updated successfully!</h3>");
        } else {
            out.println("<h3>No record found with Employee No: " + eno + "</h3>");
        }

    } catch (Exception e) {
        out.println("<h3>Error: " + e.getMessage() + "</h3>");
    } finally {
        try { if (ps != null) ps.close(); } catch(Exception ex){}
        try { if (con != null) con.close(); } catch(Exception ex){}
    }
%>
```

update.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Update Employee</title>
</head>
```

```
<body>
   <h2>Update Employee Details</h2>
   <form action="UpdateEmployee.jsp" method="post">
      <label for="eno">Employee No:</label>
      <input type="text" id="eno" name="eno" required><br><br>

      <label for="name">Employee Name:</label>
      <input type="text" id="name" name="name" required><br><br>

      <label for="age">Age:</label>
      <input type="number" id="age" name="age" required><br><br>

      <label for="desg">Designation:</label>
      <input type="text" id="desg" name="desg" required><br><br>

      <label for="salary">Salary:</label>
      <input type="number" id="salary" name="salary" required><br><br>

      <input type="submit" value="Update Employee">
   </form>
</body>
</html>
```

Output:



Employee record updated successfully!

```
mysql> -- Insert some test data
mysql> INSERT INTO employee VALUES (101, 'Amit', 30, 'Developer', 45000);
Query OK, 1 row affected (0.04 sec)

mysql> INSERT INTO employee VALUES (102, 'Sneha', 28, 'Tester', 30000);
Query OK, 1 row affected (0.01 sec)

mysql> select* from employee;
+-----+---------+------+--------------------+--------+
| eno | name    | age  | desg               | salary |
+-----+---------+------+--------------------+--------+
| 101 | Janhavi |   23 | Software Developer | 300000 |
| 102 | Sneha   |   28 | Tester             |  30000 |
+-----+---------+------+--------------------+--------+
2 rows in set (0.22 sec)
```

Record before updation

Updated record

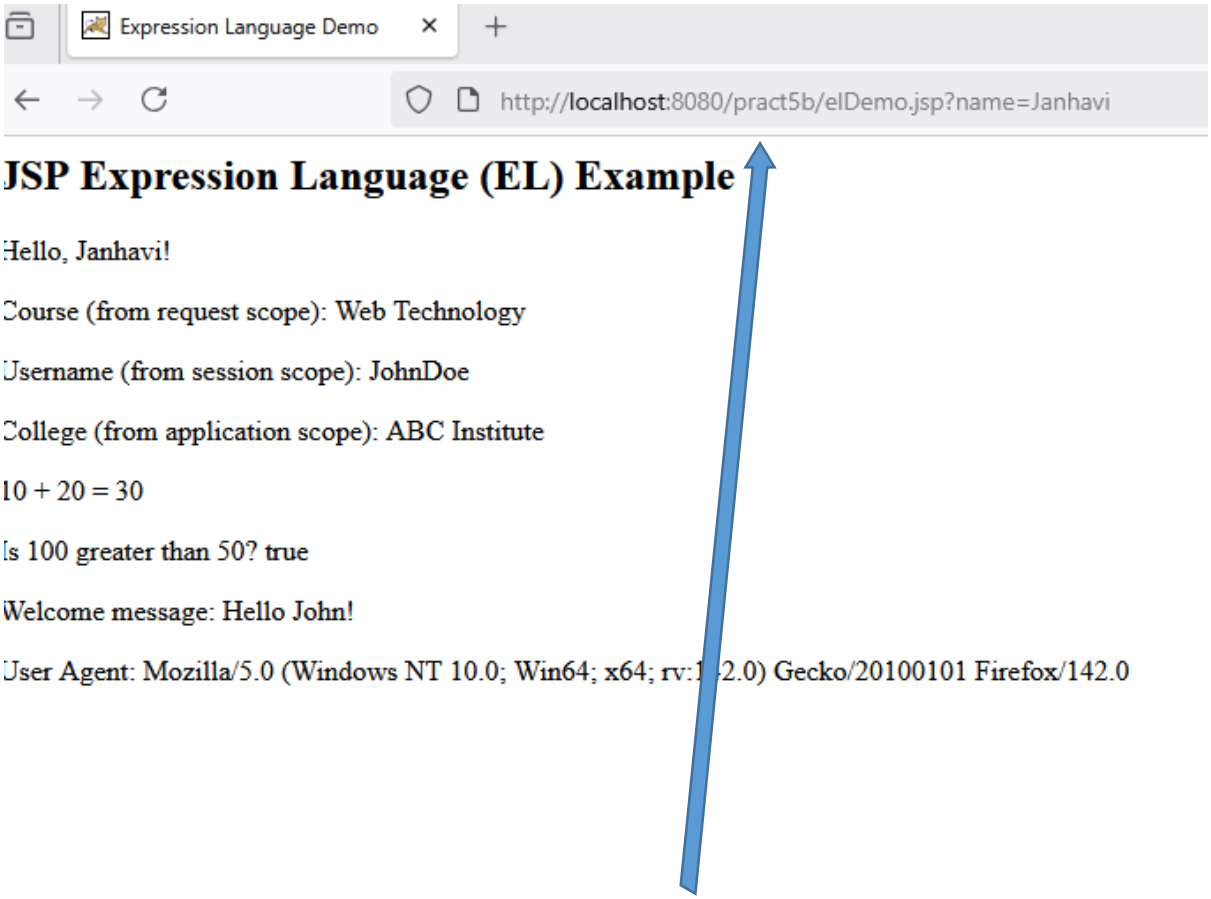
b. Create a JSP page to demonstrate the use of Expression language.

elDemo.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>Expression Language Demo</title>
</head>
<body>
    <h2>JSP Expression Language (EL) Example</h2>

    <!-- Reading request parameter directly -->
    <p>Hello, ${param.name}!</p>

    <!-- Accessing request, session, and application attributes -->
    <%
        request.setAttribute("course", "Web Technology");
        session.setAttribute("username", "JohnDoe");
```

```
        application.setAttribute("college", "ABC Institute");
    %>
    <p>Course (from request scope): ${course}</p>
    <p>Username (from session scope): ${username}</p>
    <p>College (from application scope): ${college}</p>

    <!-- Arithmetic operation using EL -->
    <p>10 + 20 = ${10 + 20}</p>

    <!-- Relational operator -->
    <p>Is 100 greater than 50? ${100 gt 50}</p>

    <!-- Conditional operator -->
    <p>Welcome message: ${username == 'JohnDoe' ? 'Hello John!' : 'Hello Guest!'}</p>

    <!-- Accessing header -->
    <p>User Agent: ${header["user-agent"]}</p>
</body>
</html>
```

Output:



Type this URL as is(You can change the name to your name as well)

c. Create a JSP application to demonstrate the use of JSTL.

Prerequisites:

Download and add the jar file to your project - jstl-1.2

index.jsp

```
<%@ page contentType="text/html; charset=UTF-8" %>
<html>
<head>
    <title>JSTL Demo</title>
</head>
<body>
    <h2>JSTL Demonstration</h2>

    <form action="result.jsp" method="post">
        Enter your name: <input type="text" name="username"/><br><br>
```

```
        Enter a number: <input type="text" name="num"/><br><br>
        <input type="submit" value="Submit"/>
    </form>
</body>
</html>
```

result.jsp

```
<%@ page contentType="text/html; charset=UTF-8" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
<head>
    <title>JSTL Result</title>
</head>
<body>
    <h2>JSTL Example Output</h2>

    <!-- Catch request parameters -->
    <c:set var="user" value="${param.username}" />
    <c:set var="number" value="${param.num}" />

    <!-- If/Choose condition -->
    <c:if test="${not empty user}">
        <p>Hello, <b>${user}</b>!</p>
    </c:if>

    <c:choose>
        <c:when test="${number % 2 == 0}">
            <p>The number <b>${number}</b> is Even.</p>
        </c:when>
        <c:otherwise>
            <p>The number <b>${number}</b> is Odd.</p>
        </c:otherwise>
```

```
    </c:choose>

    <!-- Loop example -->
    <h3>Numbers from 1 to ${number}:</h3>
    <ul>
        <c:forEach var="i" begin="1" end="${number}">
            <li>${i}</li>
        </c:forEach>
    </ul>

</body>
</html>
```

Output:

JSTL Example Output

Hello, **Janhavi**!

The number **20** is Even.

Numbers from 1 to 20:

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20

PRACTICAL NO. 6

Implement the following EJB Applications.

a. Create a Currency Converter application using EJB.

Output:

b. Develop a Simple Room Reservation System Application Using EJB.

Output:

c. Develop simple shopping cart application using EJB [Stateful Session Bean].

Output:

PRACTICAL NO. 7

Implement the following EJB applications with different types of Beans.

a. Develop simple EJB application to demonstrate Servlet Hit count using Singleton Session Beans.

Output:

b. Develop simple visitor Statistics application using Message Driven Bean [Stateless Session Bean].

Output:

c. Develop simple Marks Entry Application to demonstrate accessing Database using EJB.

Output:

PRACTICAL NO. 8

Implement the following JPA applications.

a. Develop a simple Inventory Application Using JPA.

Output:

b. Develop a Guestbook Application Using JPA.

Output:

c. Create simple JPA application to store and retrieve Book details.

Output:

PRACTICAL NO. 9

Implement the following JPA applications with ORM and Hibernate.

a. Develop a JPA Application to demonstrate use of ORM associations.

Output:

b. Develop a Hibernate application to store Feedback of Website Visitor in MySQL Database.

Output:

c. Develop a Hibernate application to store and retrieve employee details in MySQL Database.

Output:

PRACTICAL NO. 10

Implement the following Hibernate applications.

a. Develop an application to demonstrate Hibernate One- To -One Mapping Using Annotation.

Output:

b. Develop Hibernate application to enter and retrieve course details with ORM Mapping.

Output:

c. Develop a five page web application site using any two or three Java EE Technologies.

Output: