CSI THOOTHUKUDI-NAZARETH DIOCESE

DR.G.U.POPE COLLEGE OF ENGINEERING

POPE NAGAR SAWYERPURAM-628 251



Register No :

**Certified that this is the bonafide record of work done by**

**Selvan/Selvi ……………………………………………………………………… of ……….**

**Semester ……….. branch for the lab ……………………………………………………**

**During the year…………………**

**Staff In-charge H. O.D**

**Submitted for the university practical Examination held on …………..**

**Internal Examiner External Examiner**

INDEX

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.No | Date | Title | Page | Marks | Sign |
| 1 |  |  | 3 |  |  |
| 2 |  |  | 7 |  |  |
| 3 |  |  | 10 |  |  |
| 4 |  |  | 14 |  |  |
| 5 |  |  | 16 |  |  |
| 6 |  |  | 18 |  |  |
| 7 |  |  | 27 |  |  |

|  |  |
| --- | --- |
| **ExNo : 1** | **Design an HTML web page to mark the hotspots of TN in such a way that , by clicking the hotspots it generates info about the hotspots .** |
|  |

**Aim :** Design an HTML web page to mark the hotspots of TN in such a way that , by clicking the hotspots it generates info about the hotspots .

**Procedure:**

**Installation:** Install VSCode of its latest versions.

**Algorithm :**

1. Create an folder named **“WEB TECH”** in your desktop,then create an other folder inside it named “**IMAGE MAPPING**”.
2. Inside image mapping folder create an other folder named **‘Map’**.
3. Download an TN map from browser & rename it as ‘TN Map’ , then place the map inside the folder (‘Map’) you have created.
4. Open VScode ,**<FILE> <OPEN FOLDER>,** open your “IMAGE MAPPIMG” from desktop.
5. Place your cursor on folder ‘image mapping’ , then right click it , then select new file and name it as “INDEX.html” . Then pursue your HTML scripts.
6. To change the aspect coordinates of the hotspot ,open an web browser and browse <https://www.image-map.net/>
7. Place your downloaded map there and check the map coordinates then replace it in your ‘index.html’file.
8. Follow Step 5 to create individual hotspot files & map it with index.html.
9. In VSCode open extensions panel and browse “Live Server - Ritwick Dey” , install it.
10. Move your cursor to file ‘index.html’ select and right click it , you’ll get an option ‘Run using live server’ Click it . then the browser opens automatically.Surf through the map to see the results.

**Program :**

**File Name:** “index.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>TN Map with Four Political Districts</title>

</head>

<body>

<center>

<img src="img/(map name and format)" usemap="#tn" alt="TN Map">

</center>

<map name="TN">

<area shape="circle" coords="395,819,9" href="chennai.html" alt="Chennai">

<area shape="circle" coords="322,956,1" href="tuty.html" alt="tuty">

<area shape="circle" coords="290,900,1" href="coimbatore.html" alt="coimbatore">

<area shape="circle" coords="290,930,22" href="theni.html" alt="theni">

</map>

</body>

**File Name:** “chennai.html”

<!DOCTYPE html>

<html>

<head>

<title>Chennai</title>

</head>

<body bgcolor="white">

<font color="black" size="5">

<pre>

<center>

<b>Chennai</b> is the capital of <b>Tamil Nadu</b><br>

It is the <i><b>"Queen Of South India"</b></i><br>

Mother tongue is <i><b>"Tamil"</b></i><br>

It contains many tourist spots.

</center>

</pre>

</font>

</body>

</html>

**File Name:** “coimbatore.html”

<!DOCTYPE html>

<html>

<head>

<title>Coimbaotre</title>

</head>

<body bgcolor="white">

<font color="black" size="5">

<pre>

<center>

<b>Coimbatore</b> is the major city of <b>Tamil Nadu</b><br>

It is the <i><b>"MANCHESTER of south India"</b></i><br>

It is the industrial and educatonal hubs <i><b>"Nature's Gift"</b></i><br>

The city boasts a pleasant climate, beautiful landscapes, and is surrounded by lush greenery and hills.

It is also renowned for its educational institutions and healthcare facilities. Coimbatore is a gateway to several popular hill stations like Ooty and Coonoor.

</center>

</pre>

</font>

</body>

</html>

**File Name:** “theni.html”

<!DOCTYPE html>

<html>

<head>

<title>THENI</title>

</head>

<body bgcolor="white">

<font color="black" size="5">

<pre>

<center>

<b>Theni</b> priceless beauty of <b>Tamil Nadu</b><br>

It is my personal to mention is as <i><b>"MY LOVE"</b></i><br>

a good explorable spot which is an <i><b>"Nature's Gift"</b></i><br>

It is surrounded by the Western Ghats and is famous for its scenic beauty, hill stations,

waterfalls, and agricultural produce, especially cardamom and grapes.

Theni is also a gateway to tourist spots like Meghamalai and the Periyar Wildlife Sanctuary.

</center>

</pre>

</font>

</body>

</html>

**File Name:** “tuty.html”

<!DOCTYPE html>

<html>

<head>

<title>Tuticorin</title>

</head>

<body bgcolor="white">

<font color="black" size="5">

<pre>

<center>

<b>Tuticorin</b> is the salt palace of <b>Tamil Nadu</b><br>

It is the <i><b>"Taste associator of India"</b></i><br>

Special sweet cornerization <i><b>"Macroon"</b></i><br>

Tuticorin always reffered as "TUTY" is an district of taste buds

with spice,sweet etc.

</center>

</pre>

</font>

</body>

</html>

**Output :**

****

**Result :** Thus designing an HTML web page to mark the hotspots of TN in such a way that , by clicking the hotspots it generates info about the hotspots has been done successfully

|  |  |
| --- | --- |
| **ExNo : 2** | **Design an HTML web page with our college info in External Style Sheet, Embedded Style Sheet, Inline style Sheet.** |
|  |

**Aim :** To Design an HTML web page with our college info in External Style Sheet, Embedded Style Sheet, Inline style Sheet.

**Algorithm :**

1. Open VSCode , open WEB TECH folder , move your on web tech folder select, right click it <New Folder>.
2. Name the new folder as “**CASCADING STYLE SHEETS**”.
3. Then open new files in CASCADING STYLE SHEETS and save the codings with their extensions.
4. Open live server to view the resultant web pages.

**Program:**

**File name :** “external.css”

/\* external.css \*/

body {

background-color: lightyellow;

}

h1 {

color: darkgreen;

text-align: center;

}

p {

font-size: 18px;

color: navy;

}

**File Name** : “external.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>External Styles Example</title>

<link rel="stylesheet" type="text/css" href="external.css">

</head>

<body>

<h1>Dr.G.U.Pope.College Of Engineering</h1>

<p>Dr. G.U. Pope College of Engineering stands as the fulfillment of the earnest aspirations of Sawyerpuram’s well-wishers,

who longed for the accessibility of technical education in their vicinity. Commencing its service to the Sawyerpuram

community in the academic year 2002-2003, the college operates under the auspices of the CSI Thoothukudi-Nazareth Diocese.

This institution was established by devoted and sincere missionaries committed to delivering quality and disciplined education to remote areas.

Nestled in a picturesque location amidst natural beauty, the college is situated on the Thoothukudi-Sawyerpuram ‘Theri Road, providing an ideal setting for academic pursuits</p>

</body>

</html>

**File Name** : “inline.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>Inline Styles Example</title>

</head>

<body style="background-color: lightgray;">

<h1 style="color: purple; text-align: center;">Dr.G.U.Pope.College Of Engineering</h1>

<p style="color: blue; font-weight: bold;">Dr. G.U. Pope College of Engineering stands as the fulfillment of the earnest aspirations of Sawyerpuram’s well-wishers,

who longed for the accessibility of technical education in their vicinity. Commencing its service to the Sawyerpuram

community in the academic year 2002-2003, the college operates under the auspices of the CSI Thoothukudi-Nazareth Diocese.

This institution was established by devoted and sincere missionaries committed to delivering quality and disciplined education to remote areas.

Nestled in a picturesque location amidst natural beauty, the college is situated on the Thoothukudi-Sawyerpuram ‘Theri Road, providing an ideal setting for academic pursuits</p>

</body>

</html>

**File Name** : “embedded.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>Embedded Styles Example</title>

<style>

body {

background-color: lightblue;

}

h1 {

color: darkred;

text-align: center;

}

p {

font-style: italic;

color: green;

}

</style>

</head>

<body>

<h1>Dr.G.U.Pope.College Of Engineering</h1>

<p>Dr. G.U. Pope College of Engineering stands as the fulfillment of the earnest aspirations of Sawyerpuram’s well-wishers,

who longed for the accessibility of technical education in their vicinity. Commencing its service to the Sawyerpuram

community in the academic year 2002-2003, the college operates under the auspices of the CSI Thoothukudi-Nazareth Diocese.

This institution was established by devoted and sincere missionaries committed to delivering quality and disciplined education to remote areas.

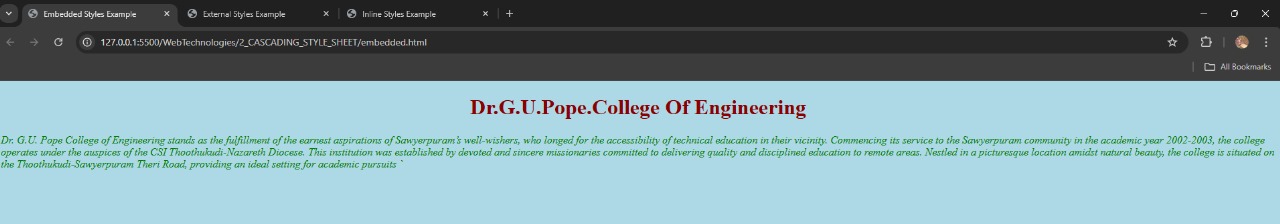
Nestled in a picturesque location amidst natural beauty, the college is situated on the Thoothukudi-Sawyerpuram ‘Theri Road, providing an ideal setting for academic pursuits `<style>` tag in the head.</p>

</body>

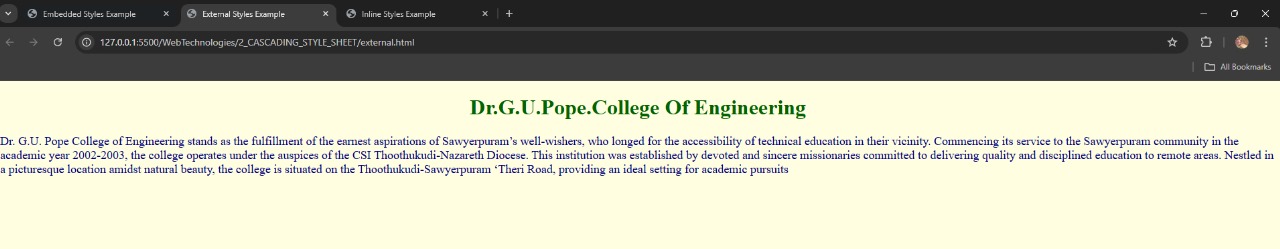
</html>

**Output :**

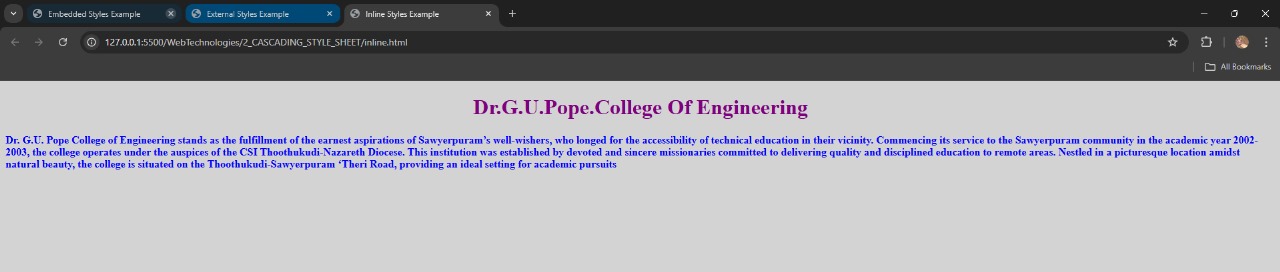
**Embedded Style**

****

**External Style**

****

**Inline Style**

****

**RESULT :** Thus to Design an HTML web page with our college info in External Style Sheet, Embedded Style Sheet, Inline style Sheet has been executed successfully.

|  |  |
| --- | --- |
| **ExNo : 3** | **Design an HTML web page to client side scripts for Validating web form controls** |
|  |

**AIM :** To Design an HTML web page to client side scripts for Validating web form controls .

**Algorithm :**

1. Create your new folder for “WEB FORM VALIDATION” & new file.
2. Feed your codings in it .  
   Run with live server.
3. Each web page contains an activity box with some restrictions and an submit button in it.
4. Each entry will be checked for its validity.
5. Run and observe the web forms.

**Program :**

**File Name** : “email.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>Time Validation</title>

<script>

function validateTime() {

var time = document.getElementById("time").value;

var regex = /^[0-9]{2}:[0-9]{2}$/;

if (regex.test(time)) {

alert("Time is valid.");

} else {

alert("Please enter a time in HH:MM format.");

}

return false; // Prevent form submission for demonstration

}

</script>

</head>

<body>

<form onsubmit="return validateTime()">

<label for="time">Time (HH:MM):</label>

<input type="text" id="time" name="time" required>

<input type="submit" value="Submit">

</form>

</body>

</html>

**File Name** : “number.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>Number Validation</title>

<script>

function validateNumber() {

var number = document.getElementById("number").value;

if (!isNaN(number) && number !== "") {

alert("Number is valid.");

} else {

alert("Please enter a valid number.");

}

return false; // Prevent form submission for demonstration

}

</script>

</head>

<body>

<form onsubmit="return validateNumber()">

<label for="number">Number:</label>

<input type="text" id="number" name="number" required>

<input type="submit" value="Submit">

</form>

</body>

</html>

**File Name** : “password.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>Password Validation</title>

<script>

function validatePassword() {

var password = document.getElementById("password").value;

if (password.length >= 8) {

alert("Password is valid.");

} else {

alert("Password must be at least 8 characters long.");

}

return false; // Prevent form submission for demonstration

}

</script>

</head>

<body>

<form onsubmit="return validatePassword()">

<label for="password">Password:</label>

<input type="password" id="password" name="password" required>

<input type="submit" value="Submit">

</form>

</body>

</html>

**File Name** : “time.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>Time Validation</title>

<script>

function validateTime() {

var time = document.getElementById("time").value;

var regex = /^[0-9]{2}:[0-9]{2}$/;

if (regex.test(time)) {

alert("Time is valid.");

} else {

alert("Please enter a time in HH:MM format.");

}

return false; // Prevent form submission for demonstration

}

</script>

</head>

<body>

<form onsubmit="return validateTime()">

<label for="time">Time (HH:MM):</label>

<input type="text" id="time" name="time" required>

<input type="submit" value="Submit">

</form>

</body>

</html>

**Output :**

**RESULT :** Thus to Design an HTML web page to client side scripts for Validating web form controls has been executed successfully.

|  |  |
| --- | --- |
| **ExNo : 4** | **Design an HTML web page & server program to client side scripts for invoking servelets.** |
|  |

**AIM :**  Design an HTML web page & server program to client side scripts for invoking servelets.

**Algorithm :**

1. Create your new folder for “INVOKING SERVELET FROM APPLET”.
2. Create an other folder inside Invoking servelet…. as “public” & create a new file,save it with html extension.
3. Open a terminal in VSCode & **cd** to current working folder , then install “**npm init -y**”, “**npm install express**”.
4. Create an js file in in your invoking serve… folder to add server codings in it.
5. Save all files , then navigate the path in terminal ,then type “**node server.js**” to run the file.
6. That generates an http localhost, C&P it in a browser.
7. Observe the result from the webpage .

**Program :**

**File Name** : “index.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>Node.js Server Example</title>

<script>

function fetchServerData() {

var xhr = new XMLHttpRequest();

xhr.open("GET", "/data", true);

xhr.onreadystatechange = function() {

if (xhr.readyState === 4 && xhr.status === 200) {

document.getElementById("response").innerHTML = xhr.responseText;

}

};

xhr.send();

}

</script>

</head>

<body>

<h1>Node.js Server Example</h1>

<button onclick="fetchServerData()">Fetch Data from Server</button>

<div id="response"></div>

</body>

</html>

**File Name** : “server.js”

const express = require('express');

const app = express();

const port = 3000;

app.use(express.static('public'));

app.get('/data', (req, res) => {

res.send('<html><body><h1>Hi,This is Mr.David in front of you !!!!....</h1></body></html>');

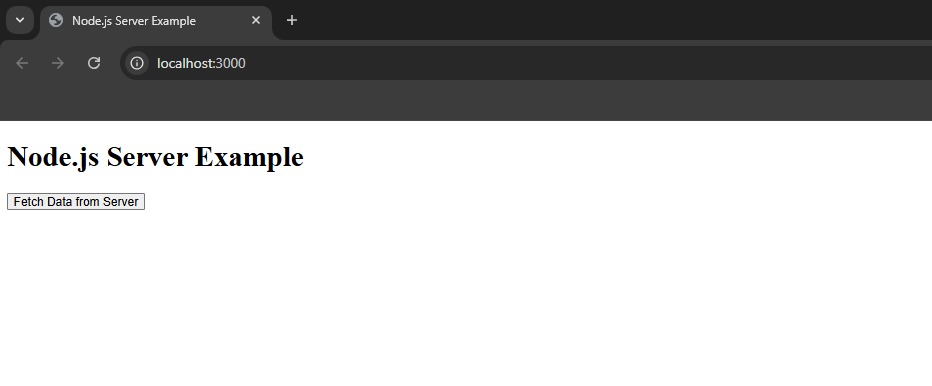
});

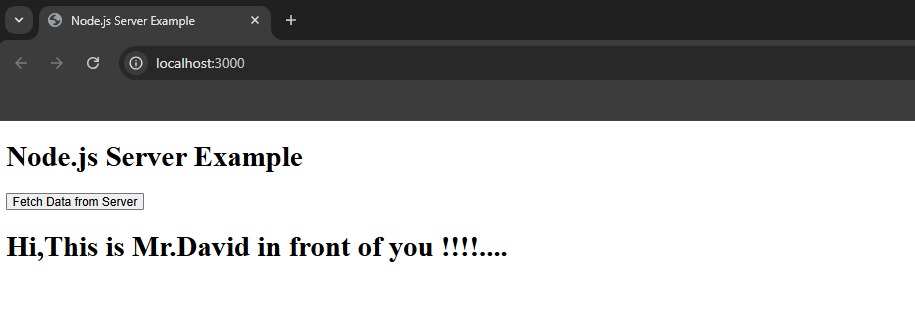
app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

});

**Output :**





**RESULT :**  Thus to Design an HTML web page & server program to client side scripts for invoking servelets has been executed successfully.

|  |  |
| --- | --- |
| **ExNo : 5** | **Design an HTML web page & server program to client side scripts for invoking servelets under HTML forms and Session tracking**. |
|  |

**AIM :**  To Design an HTML web page & server program to client side scripts for invoking servelets under HTML forms.

**Algorithm :**

1. Create a “new folder” named “**INVOKING SERVELETS HTMLFORMS**”.
2. Create a new file,save it with html extension.
3. Create an js file in in your invoking serve… folder to add server codings in it.
4. Save all files , then navigate the path in terminal ,then type “**node server.js**” to run the file.
5. That generates an http localhost, C&P it in a browser.
6. Observe the result from the webpage .

**Program :**

**File Name** : “form.html”

<!DOCTYPE html>

<html lang="en">

<head>

<title>Submit Form to Node.js</title>

</head>

<body>

<h1>Enter Your Details</h1>

<form action="http://localhost:3000/submit" method="POST">

<label for="username">Username:</label><br>

<input type="text" id="username" name="username"><br><br>

<label for="email">Email:</label><br>

<input type="email" id="email" name="email"><br><br>

<input type="submit" value="Submit">

</form>

</body>

</html>

**Filename :** server.js

const express = require('express');

const bodyParser = require('body-parser');

const mongoose = require('mongoose');

const app = express();

const port = 3000;

// MongoDB connection

mongoose.connect('mongodb://localhost:27017/formdata')

.then(() => {

console.log('Connected to MongoDB');

})

.catch(err => {

console.error('MongoDB connection error:', err);

});

// Define a schema and model for storing form data

const formDataSchema = new mongoose.Schema({

username: String,

email: String

});

// Specify the collection name explicitly

const FormData = mongoose.model('FormData', formDataSchema, 'recieved\_info');

// Middleware to parse the body of POST requests

app.use(bodyParser.urlencoded({ extended: true }));

// Serve the HTML file

app.get('/', (req, res) => {

res.sendFile(\_\_dirname + '/form.html');

});

// Handle form submission

app.post('/submit', async (req, res) => {

const username = req.body.username;

const email = req.body.email;

console.log('Received data:', { username, email }); // Log received data

// Create a new document

const formData = new FormData({

username: username,

email: email

});

try {

// Save the document to MongoDB

await formData.save();

res.send(`

<html>

<body>

<h2>Form Data Received and Stored:</h2>

<p>Username: ${username}</p>

<p>Email: ${email}</p>

</body>

</html>

`);

console.error('Error saving data to MongoDB:', err); // Improved logging

res.status(500).send('Error saving data to MongoDB');

}

});

app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

});

**Output :**

**RESULT :**  Thus to Design an HTML web page & server program to client side scripts for invoking servelets html forms and session tracking has been executed successfully.

|  |  |
| --- | --- |
| **ExNo : 6** | **Design an HTML web page & server program to create three-tier application for conducting online exams, displaying student mark list.** |
|  |

**AIM :**  To Design an HTML web page & server program to create three-tier application for conducting online exams, displaying student mark-list.

**Algorithm :**

1. Create your new folder for “3TIER APPLICATION”.
2. Create a folder in it “lib” and include “mysql-connector-j-9.1.0.jar” file to establish java connection to mysql in it.
3. Create an other folder “src” in it and add all your java files in it. Then create a folder”webcontent” in it and add your xml , jsp , sql docs in it.
4. That generates an http localhost, C&P it in a browser.
5. Observe the result from the webpage .

**Program :**

**Folder Name** : “src”

**File name:**  “DatabaseConnection.java”

import java.sql.\*;

public class DatabaseConnection {

public static Connection initializeDatabase() throws SQLException, ClassNotFoundException {

String url = "jdbc:mysql://localhost:3306/ExamSystem";

String user = "root";

String password = "password";

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

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

}

}

**File Name** : “ExamServelet.java”

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.IOException;

import java.util.ArrayList;

import java.util.List;

public class ExamServlet extends HttpServlet {

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

List<Question> questions = QuestionDAO.getQuestions();

request.setAttribute("questions", questions);

request.getRequestDispatcher("exam.jsp").forward(request, response);

}

}

**File Name :** “LoginServelet.java”

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.IOException;

import java.sql.\*;

public class LoginServlet extends HttpServlet {

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

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

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

try (Connection con = DatabaseConnection.initializeDatabase();

PreparedStatement ps = con.prepareStatement("SELECT \* FROM students WHERE username=? AND password=?")) {

ps.setString(1, username);

ps.setString(2, password);

ResultSet rs = ps.executeQuery();

if (rs.next()) {

request.getSession().setAttribute("studentId", rs.getInt("id"));

response.sendRedirect("ExamServlet");

} else {

response.getWriter().println("Invalid credentials");

}

} catch (Exception e) {

e.printStackTrace();

}

}

}

**File Name :** “QuestionDAO.java”

// QuestionDAO.java

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class QuestionDAO {

public static List<Question> getQuestions() {

List<Question> questions = new ArrayList<>();

try (Connection con = DatabaseConnection.initializeDatabase();

Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM exams")) {

while (rs.next()) {

questions.add(newQuestion(rs.getInt("id"),rs.getString("question"),rs.getString("option1"), rs.getString("option2"), rs.getString("option3"), rs.getString("option4"), rs.getInt("answer")));

}

} catch (Exception e) {

e.printStackTrace();

}

return questions;

}

}

**File Name :** “StudentDAO.java”

import java.sql.\*;

public class StudentDAO {

public static void updateScore(int studentId, int score) {

try (Connection con = DatabaseConnection.initializeDatabase();

PreparedStatement ps = con.prepareStatement("UPDATE students SET score=? WHERE id=?")) {

ps.setInt(1, score);

ps.setInt(2, studentId);

ps.executeUpdate();

} catch (Exception e) {

e.printStackTrace();

}

}

}

**File Name :** “SubmitServelet”

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.IOException;

import java.util.List;

public class SubmitServlet extends HttpServlet {

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

int score = 0;

List<Question> questions = QuestionDAO.getQuestions();

for (Question q : questions) {

String selectedOption = request.getParameter("q" + q.getId());

if (selectedOption != null && Integer.parseInt(selectedOption) == q.getAnswer()) {

score++;

}

}

int studentId = (int) request.getSession().getAttribute("studentId");

StudentDAO.updateScore(studentId, score);

request.setAttribute("score", score);

request.setAttribute("totalQuestions", questions.size());

request.getRequestDispatcher("result.jsp").forward(request, response);

}

}

**Folder Name :** “WebContent”

**File Name :** “web.xml”

<!-- web.xml -->

<web-app>

<servlet>

<servlet-name>LoginServlet</servlet-name>

<servlet-class>LoginServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>LoginServlet</servlet-name>

<url-pattern>/LoginServlet</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>ExamServlet</servlet-name>

<servlet-class>ExamServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>ExamServlet</servlet-name>

<url-pattern>/ExamServlet</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>SubmitServlet</servlet-name>

<servlet-class>SubmitServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>SubmitServlet</servlet-name>

<url-pattern>/SubmitServlet</url-pattern>

</servlet-mapping>

</web-app>

**File Name :** “exam.jsp”

<!-- exam.jsp -->

<%@ page import="java.util.List" %>

<%@ page import="models.Question" %>

<%

List<Question> questions = (List<Question>) request.getAttribute("questions");

%>

<!DOCTYPE html>

<html>

<head>

<title>Online Exam</title>

</head>

<body>

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

<%

for (Question q : questions) {

%>

<p><%= q.getQuestion() %></p>

<input type="radio" name="q<%= q.getId() %>" value="1"> <%= q.getOption1() %><br>

<input type="radio" name="q<%= q.getId() %>" value="2"> <%= q.getOption2() %><br>

<input type="radio" name="q<%= q.getId() %>" value="3"> <%= q.getOption3() %><br>

<input type="radio" name="q<%= q.getId() %>" value="4"> <%= q.getOption4() %><br>

<%

}

%>

<input type="submit" value="Submit">

</form>

</body>

</html>

**File Name :** “login.jsp”

<!-- login.jsp -->

<!DOCTYPE html>

<html>

<head>

<title>Login</title>

</head>

<body>

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

Username: <input type="text" name="username"><br>

Password: <input type="password" name="password"><br>

<input type="submit" value="Login">

</form>

</body>

</html>

**File Name :** “results.jsp”

<!-- result.jsp -->

<!DOCTYPE html>

<html>

<head>

<title>Exam Result</title>

</head>

<body>

<h2>Your Score: <%= request.getAttribute("score") %> / <%= request.getAttribute("totalQuestions") %></h2>

</body>

</html>

**File Name :** “DB.sql”

CREATE DATABASE ExamSystem;

USE ExamSystem;

-- Table for storing student details

CREATE TABLE students (

id INT PRIMARY KEY AUTO\_INCREMENT,

username VARCHAR(50) NOT NULL,

password VARCHAR(50) NOT NULL,

score INT DEFAULT 0

);

-- Table for storing exam questions

CREATE TABLE exams (

id INT PRIMARY KEY AUTO\_INCREMENT,

question TEXT NOT NULL,

option1 VARCHAR(100),

option2 VARCHAR(100),

option3 VARCHAR(100),

option4 VARCHAR(100),

answer INT

);

**Output :**

**RESULT :**  Thus to Design an HTML web page & server program to create three-tier application for conducting online exams, displaying student mark-list has been done successfully.

|  |  |
| --- | --- |
| **ExNo : 7** | **PROGRAMMING USING XML SCHEMA- XSLT/XSL** |
|  |

**AIM :**  To program using xml schema- xslt/xsl.

**Algorithm :**

1. Create your new folder for “XML SCHEMA”.
2. Create coding files,save it with xml,xsd,xslt extensions.
3. Save all files , then navigate the path in terminal ,then use command “**xsltproc transform.xslt data.xml > output.html**”to run the file.
4. That generates an html file as an output file , run the file to view the output.
5. Observe the result from the webpage .

**Program :**

**File Name** : “data.xml”

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

<library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="library.xsd">

<book>

<title>1984</title>

<author>George Orwell</author>

<year>1949</year>

</book>

<book>

<title>Brave New World</title>

<author>Aldous Huxley</author>

<year>1932</year>

</book>

</library>

**File Name** : “library.xsd”

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="library">

<xs:complexType>

<xs:sequence>

<xs:element name="book" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="title" type="xs:string"/>

<xs:element name="author" type="xs:string"/>

<xs:element name="year" type="xs:int"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>

**File Name** : “transform.xslt”

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

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="html" encoding="UTF-8" indent="yes"/>

<xsl:template match="/library">

<html>

<head>

<title>Book Library</title>

</head>

<body>

<h1>My Book Collection</h1>

<table border="1">

<tr>

<th>Title</th>

<th>Author</th>

<th>Year</th>

</tr>

<xsl:for-each select="book">

<tr>

<td><xsl:value-of select="title"/></td>

<td><xsl:value-of select="author"/></td>

<td><xsl:value-of select="year"/></td>

</tr>

</xsl:for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

**Output :**

**RESULT :**  Thus to program using xml schema- xslt/xsl has been done successfully.