

**DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING**

**AI23431 – WEB TECHNOLOGY AND MOBILE APPLICATION**

**(REGULATION 2023)**

### RAJALAKSHMI ENGINEERING COLLEGE

**Thandalam, Chennai-602015**

**Name: AFRAH M**

**Register No: 2116231501008**

**Year / Branch / Section: 2nd / AIML / FA Semester: IV**

**Academic Year: 2024 - 2025**

|  |  |
| --- | --- |
| **EXP:01** | **EMBEDDED MAP AND TYPES OF CSS** |

**AIM:**

HTML & CSS

1. Create a web page to embed a map along with hot spot, frames &amp; links.
2. Create a web page using an embedded, external and inline CSS file.

**CODE:**

#### a.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Interactive World Map</title>

<style>

body {

font-family: Arial, sans-serif; background-color: #f4f4f4; text-align: center;

margin: 0; padding: 20px;

}

.container {

background: white; padding: 20px; border-radius: 10px;

box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1); max-width: 800px;

margin: auto;

}

h2 {

color: #333;

}

.map-container { display: flex;

justify-content: center; align-items: center; margin: 20px 0;

}

img {

max-width: 100%; height: auto; border-radius: 10px;

}

ul {

list-style: none; padding: 0;

}

li a {

text-decoration: none; color: #007BFF;

font-weight: bold;

}

iframe {

border-radius: 10px; width: 100%; height: 400px; border: none;

}

</style>

</head>

<body>

<div class="container">

<h2>Explore the World Map</h2>

<p>Click on the highlighted areas to learn more.</p>

<div class="map-container">

<img src="img1.jpg" usemap="#worldmap" alt="World Map">

</div>

<map name="worldmap">

<area shape="rect" coords="120,180,180,240" href="https://en.wikipedia.org/wiki/United\_States" alt="USA">

<area shape="rect" coords="850,350,900,400" href="https://en.wikipedia.org/wiki/India" alt="India">

</map>

<h3>Quick Links:</h3>

<ul>

<li><a href=["https://www.google.com/maps"](http://www.google.com/maps) target="\_blank">Google Maps</a></li>

<li><a href=["https://www.bing.com/maps"](http://www.bing.com/maps) target="\_blank">Bing Maps</a></li>

</ul>

<h3>Embedded Google Maps</h3>

<iframe src="https://[www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3151.8354345093707!](http://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3151.8354345093707!) 2d144.95373631531963!3d-

37.81627937975171!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x6ad642af

0f11fd81%3A0xf5772c53a8ef1b87!2sMelbourne%20VIC%2C%20Australia!5e0!3m2!1sen!2s us!4v1605250581944!5m2!1sen!2sus"></iframe>

</div>

</body>

</html>

#### b.

try.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CSS Styling Example</title>

<!-- External CSS -->

<link rel="stylesheet" href="styles.css">

<!-- Embedded CSS -->

<style>

h1 {

color: blue;

text-align: center;

}

.embedded-style { font-size: 18px; color: green;

border: 2px solid black; padding: 10px;

}

</style>

</head>

<body>

<h1>Welcome to the Styled Web Page</h1>

<p style="color: red; font-size: 16px;">This is an inline CSS example.</p>

<p class="embedded-style">This paragraph is styled using embedded CSS.</p>

<p class="external-style">This paragraph is styled using external CSS.</p>

</body>

</html>

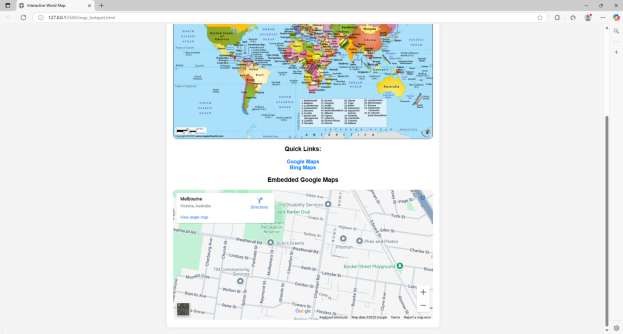
Styles.css

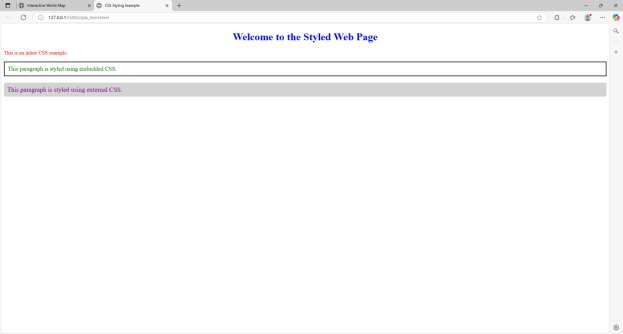
.external-style { font-size: 20px; color: purple;

background-color: lightgray; padding: 10px;

border-radius: 5px;

}

**OUTPUT:**

****

**Result**: Thus, a basic webpage with embedded maps, and a webpage illustrating different css was completed successfully.

|  |  |
| --- | --- |
| **EXP:02** | **REGISTRATION FORM VALIDATION** |

### AIM:

Write JavaScript to validate the following fields of the Registration page.

1. First Name (Name should contains alphabets and the length Should not be less than 6 characters).
2. Password (Password should not be less than 6 characters length).
3. E-mail id (should not contain any invalid and must follow the standard pattern [name@domain.com)](mailto:name@domain.com)
4. Mobile Number (Phone number should contain 10 digits only).
5. Last Name and Address (should not be Empty).

### CODE:

<!**DOCTYPE** html>

<**html** lang="en">

<**head**>

<**meta** charset="UTF-8">

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

<**title**>Registration Form</**title**>

<**style**>

body {

font-family: 'Poppins', sans-serif;

background: linear-gradient(to right, #00c6ff, #0072ff); text-align: center;

margin: 0; padding: 20px;

}

.container {

background: white; padding: 30px; border-radius: 15px;

box-shadow: 0px 4px 15px rgba(0, 0, 0, 0.2); max-width: 450px;

margin: auto;

animation: fadeIn 1s ease-in-out;

}

@keyframes fadeIn {

from { opacity: 0; transform: translateY(-20px); } to { opacity: 1; transform: translateY(0); }

}

h2 {

color: #333;

margin-bottom: 20px;

}

input {

width: 100%; padding: 12px; margin: 8px 0;

border: 1px solid #ccc;

border-radius: 8px;

font-size: 16px;

}

button {

width: 100%;

background-color: #007BFF; color: white;

padding: 12px; border: none; border-radius: 8px; font-size: 18px; cursor: pointer;

transition: background 0.3s;

}

button:hover {

background-color: #0056b3;

}

</**style**>

</**head**>

<**body**>

<**div** class="container">

<**h2**>Register Here</**h2**>

<**form** onsubmit="return validateForm()">

<**input** type="text" id="firstName" placeholder="First Name">

<**input** type="text" id="lastName" placeholder="Last Name">

<**input** type="password" id="password" placeholder="Password">

<**input** type="email" id="email" placeholder="Email">

<**input** type="text" id="mobile" placeholder="Mobile Number">

<**input** type="text" id="address" placeholder="Address">

<**button** type="submit">Register</**button**>

</**form**>

</**div**>

<**script**>

function validateForm() {

let firstName = document.getElementById("firstName").value; let lastName = document.getElementById("lastName").value; let password = document.getElementById("password").value; let email = document.getElementById("email").value;

let mobile = document.getElementById("mobile").value; let address = document.getElementById("address").value;

Z]{2,}$/;

let emailPattern = /^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA- let mobilePattern = /^[0-9]{10}$/;

if (firstName.length < 5 || !/^[a-zA-Z]+$/.test(firstName)) { alert("First Name must contain only alphabets and be at least

5 characters long.");

return false;

}

if (password.length < 6) {

alert("Password must be at least 6 characters long.");

return false;

}

if (!emailPattern.test(email)) {

alert("Enter a valid email address in the format [name@domain.com.](mailto:name@domain.com)");

return false;

}

if (!mobilePattern.test(mobile)) {

alert("Mobile number must contain exactly 10 digits."); return false;

}

if (lastName.trim() === "" || address.trim() === "") { alert("Last Name and Address cannot be empty."); return false;

}

alert("Form submitted successfully!"); return true;

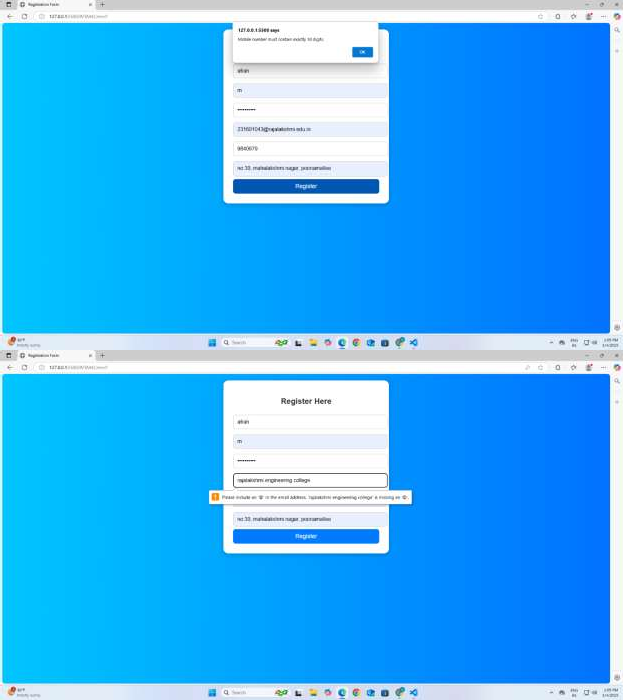
}

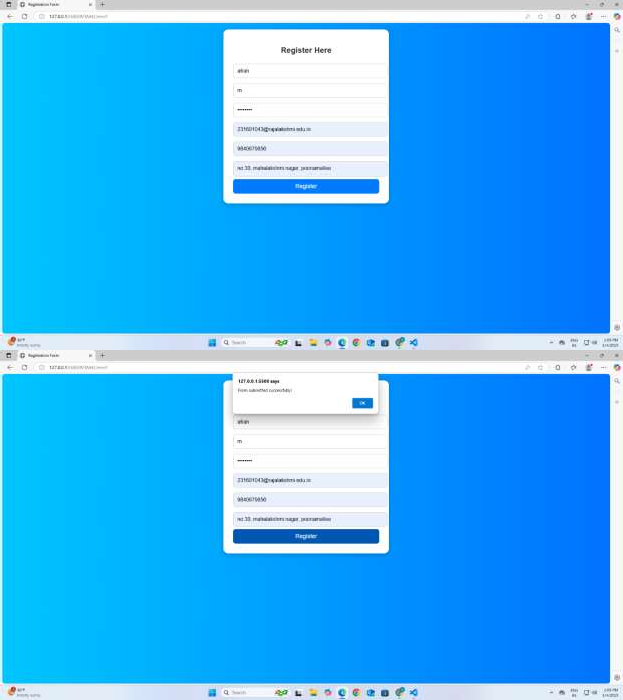
</**script**>

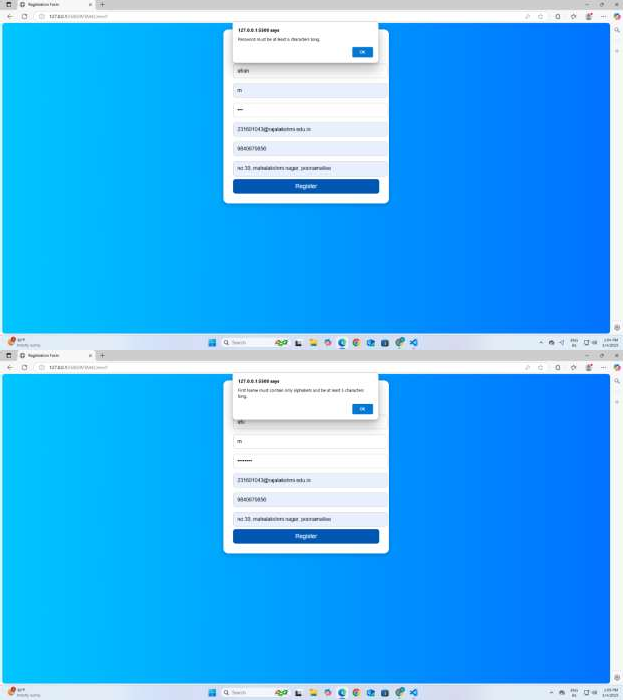
</**body**>

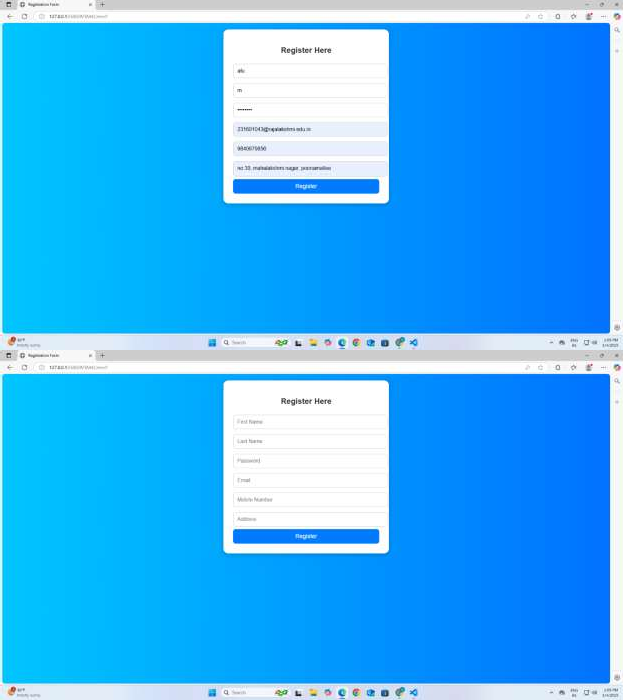
</**html**>

### OUTPUT:

****

****

****

****

**RESULT:** Thus, a registration form was executed successfully.

|  |  |
| --- | --- |
| **EXP:03** | **HELLO WORLD-SERVLET** |

##### AIM :

To create a **web page** that displays **"Hello, World!"** inside a **decorative box** centered on the screen using **HTML & CSS**.

##### ALGORITHM :

1. Start
2. Open NetBeans and create a new web project.
3. In the Web Pages folder, create or edit the index.html file.
4. Add HTML structure with a heading inside a <div> box.
5. Apply CSS styles to:
   1. Center the box on the page.
   2. Add a background gradient.
   3. Style the box with border, shadow, and animation.
6. Save the file.
7. Run the project in NetBeans.
8. Open a browser and visit
9. The page should display "Hello, World!" in a stylish box.
10. End

##### CODE:

<!DOCTYPE html>

<html>

<head>

<title>Hello World</title>

<meta charset="UTF-8">

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

<style>

/\* Full-page styling \*/ body

{

display: flex;

justify-content: center; align-items: center;

height: 100vh;

background: linear-gradient(135deg, #ff9a9e, #fad0c4); margin: 0; font-family: Arial, sans-serif;

}

/\* Stylish box \*/

.box {

background: white; padding: 30px 50px; border-radius: 15px;

box-shadow: 0px 10px 20px rgba(0, 0, 0, 0.2); text-align: center;

font-size: 28px; font-weight: bold; color: #333;

border: 5px solid #ff6b81; position:

relative;

animation: fadeIn 1.5s ease-in-out;

}

/\* Glowing Effect \*/

.box:hover {

box-shadow: 0px 0px 20px rgba(255, 107, 129, 0.8); transition: 0.3s;

}

/\* Animation \*/ @keyframes fadeIn {

from {

opacity: 0;

transform: translateY(-20px);

}

to {

opacity: 1;

transform: translateY(0);

}

}

</style>

</head>

<body>

<div class="box">✨ Hello, World! ✨</div>

</body>

</html>

### OUTPUT :

****

**RESULT :**

Thus, Servlet program to execute “Hello World” was completed successfully.

|  |  |
| --- | --- |
| **EXP:04** | **BASIC SERVLET FORM** |

###### AIM:

To create a web-based Java Servlet application that accepts a user's name and age through an HTML form and displays the submitted data back on the browser using Java Servlet technology.

###### CODE:

<!DOCTYPE html>

<html>

<head>

<title>User Form</title>

<style>

body {

font-family: Arial, sans-serif; background-color: #f4f4f4; display: flex;

justify-content: center;

align-items: center; height: 100vh; margin: 0;

}

.container { background: #fff; padding: 20px; border-radius: 10px;

box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1); text- align: center;

width: 300px;

}

h2 {

color: #333;

}

input[type="text"], input[type="number"] { width: 90%; padding: 10px; margin:

10px 0; border: 1px solid #ccc; border-radius: 5px;

}

input[type="submit"] { background: #007BFF; color: white;

padding: 10px 15px; border: none; border-radius: 5px; cursor: pointer; font- size: 16px; width: 100%;

}

input[type="submit"]:hover { background: #0056b3;

}

</style>

</head>

<body>

<div class="container">

<h2>Enter Your Details</h2>

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

<input type="text" name="username" placeholder="Enter your name" required><br>

<input type="number" name="age" placeholder="Enter your age" required><br>

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

</form>

</div>

</body>

</html>

Java Servlet :

import java.io.IOException; import java.io.PrintWriter;

import javax.servlet.ServletException; import javax.servlet.annotation.WebServlet; import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest; import javax.servlet.http.HttpServletResponse;

@WebServlet("/Exp4") // Make sure your form action matches this URL public class Exp4 extends HttpServlet {

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

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

// Retrieving form data

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

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

out.println("<h2>Welcome, " + name + "!</h2>"); out.println("<p>Your age is: " + age + "</p>"); out.println("</body></html>");

}

}

Configure web.xml:

<web-app xmlns="<http://java.sun.com/xml/ns/javaee>" version="3.0">

<servlet>

<servlet-name>UserServlet</servlet-name>

<servlet-class>UserServlet</servlet-class>

</servlet>

<servlet-mapping>

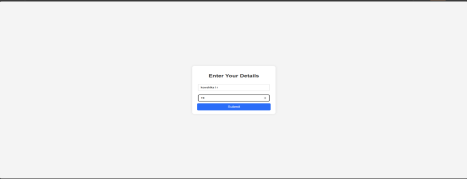
<servlet-name>UserServlet</servlet-name>

<url-pattern>/UserServlet</url-pattern>

</servlet-mapping>

</web-app>

### OUTPUT:

****



##### RESULT :

Thus the give program is executed successfully.

|  |  |
| --- | --- |
| **EXP:05** | **LOGIN FORM IN SERVLET(SESSION TRACKING)** |

AIM:

Write a Servlet to demonstrate session tracking using HttpSession. Implement a simple login system where the users session is tracked.

Code: log.html

<!-- login.html -->

<!**DOCTYPE** html>

<**html**>

<**head**><**title**>Login Page</**title**></**head**>

<**body**>

<**h2**>Login Form</**h2**>

<**form** action=*"login"* method=*"post"*>

Name: <**input** type=*"text"* name=*"username"*><**br**><**br**>

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

</**form**>

</**body**>

</**html**>

loginservlet.java

package com.user;

import jakarta.servlet.ServletException; import jakarta.servlet.annotation.WebServlet; import jakarta.servlet.http.HttpServlet; import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse; import jakarta.servlet.http.HttpSession;

import java.io.IOException;

/\*\*

\* Servlet implementation class LoginServlet

\*/

*@WebServlet*("/LoginServlet")

public class LoginServlet extends HttpServlet { private static final long ***serialVersionUID*** = 1L;

/\*\*

\* **@see** HttpServlet#HttpServlet()

\*/

public LoginServlet() { super();

// **TODO** Auto-generated constructor stub

}

/\*\*

\* **@see** HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// **TODO** Auto-generated method stub

//response.getWriter().append("Served at:

").append(request.getContextPath());

}

/\*\*

\* **@see** HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// **TODO** Auto-generated method stub

//doGet(request, response);

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

// Start session

HttpSession session = request.getSession(); session.setAttribute("name", name);

// Redirect to welcome page response.sendRedirect("welcome");

}

}

welcomeservlt.java package com.user;

import jakarta.servlet.ServletException; import jakarta.servlet.annotation.WebServlet; import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest; import jakarta.servlet.http.HttpServletResponse; import jakarta.servlet.http.HttpSession;

import java.io.IOException; import java.io.PrintWriter;

/\*\*

\* Servlet implementation class WelcomeServlet

\*/ @WebServlet("/welcome")

public class WelcomeServlet extends HttpServlet { private static final long serialVersionUID = 1L;

/\*\*

\* @see HttpServlet#HttpServlet()

\*/

public WelcomeServlet() { super();

// TODO Auto-generated constructor stub

}

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

//response.getWriter().append("Served at: ").append(request.getContextPath());

HttpSession session = request.getSession(false); // don’t create if it

doesn't exist

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

if (session != null && session.getAttribute("name") != null) { String name = (String) session.getAttribute("name"); out.println("<h1>Welcome, " + name + "!</h1>"); out.println("<a href='logout'>Logout</a>");

} else {

out.println("<h2>You are not logged in!</h2>"); out.println("<a href='log.html'>Login Again</a>");

}

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

//doGet(request, response);

}

}

logoutservlet.java package com.user;

import jakarta.servlet.ServletException; import jakarta.servlet.annotation.WebServlet; import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest; import jakarta.servlet.http.HttpServletResponse; import jakarta.servlet.http.HttpSession;

import java.io.IOException; import java.io.PrintWriter;

/\*\*

\* Servlet implementation class LogoutServlet

\*/ @WebServlet("/logout")

public class LogoutServlet extends HttpServlet { private static final long serialVersionUID = 1L;

/\*\*

\* @see HttpServlet#HttpServlet()

\*/

public LogoutServlet() { super();

// TODO Auto-generated constructor stub

}

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// TODO Auto-generated method stub

//response.getWriter().append("Served at: ").append(request.getContextPath());

HttpSession session = request.getSession(false); if (session != null) {

session.invalidate(); // Bye-bye session

}

response.setContentType("text/html"); PrintWriter out = response.getWriter(); out.println("<h2>You are logged out.</h2>"); out.println("<a href='log.html'>Login Again</a>");

}

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request,

HttpServletResponse response)

\*/

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

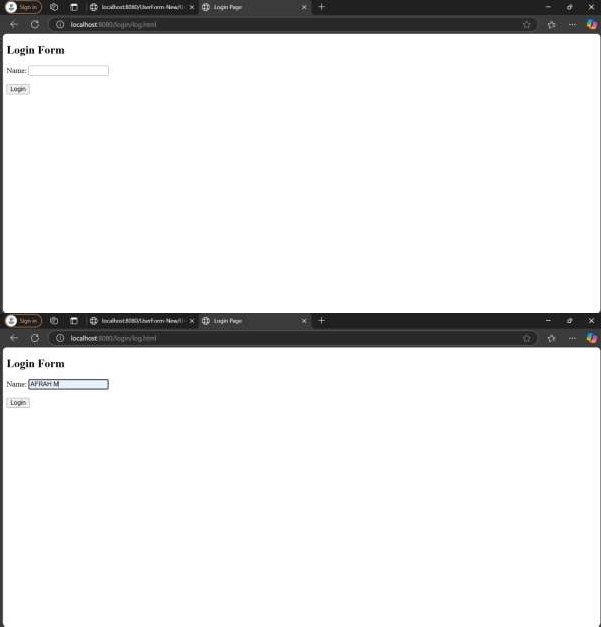
// TODO Auto-generated method stub

//doGet(request, response);

}

}

### Output:





Result: Login System working successfully.

|  |  |
| --- | --- |
| **EXP:06** | **GET METHOD AND POST METHOD** |

Aim:

To demonstrate the difference between the HTTP GET and POST methods using a Java Servlet, where a form will be created and requests will be handled accordingly by the GET and POST methods.

###### CODE:

**HTML :**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>GET vs POST Example</title>

<style>

body {

font-family: Arial, sans-serif; background-color: #f4f4f9; color: #333;

margin: 0;

padding: 0;

}

h2 {

color: #4CAF50; text- align: center; margin- top: 50px;

}

.container { width: 50%;

margin: 0 auto; background-color: #fff; padding: 30px;

border-radius: 8px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

}

.form-group {

margin-bottom: 20px;

}

label {

font-size: 16px; color: #555;

}

input[type="text"] { width: 100%;

padding: 10px; margin-top: 5px; border-radius: 4px;

border: 1px solid #ccc; box- sizing: border-box;

}

button {

background-color: #4CAF50; color: white;

padding: 10px 20px; border: none; border-radius: 4px; cursor: pointer; font- size: 16px;

}

button:hover {

background-color: #45a049;

}

.form-container { margin- bottom: 30px;

}

hr {

border: 1px solid #ddd;

}

.result {

background-color: #f9f9f9; border- left: 4px solid #4CAF50; padding: 20px;

margin-top: 20px; font-size: 16px; border-radius: 4px;

}

</style>

</head>

<body>

<div class="container">

<h2>GET vs POST Request Demonstration</h2>

<!-- Form to demonstrate GET method -->

<div class="form-container">

<h3>GET Method</h3>

<form action="DemoServlet" method="GET">

<div class="form-group">

<label for="data">Enter Data (GET):</label>

<input type="text" name="data" required>

</div>

<button type="submit">Submit (GET)</button>

</form>

</div>

<hr>

<!-- Form to demonstrate POST method -->

<div class="form-container">

<h3>POST Method</h3>

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

<div class="form-group">

<label for="data">Enter Data (POST):</label>

<input type="text" name="data" required>

</div>

<button type="submit">Submit (POST)</button>

</form>

</div>

<!-- Result section will be dynamically updated -->

<div class="result" id="result">

<!-- Display GET or POST request result here -->

</div>

</div>

</body>

</html>

Servlet.java:

import java.io.\*; import javax.servlet.\*;

import javax.servlet.http.\*;

public class DemoServlet extends HttpServlet {

// Handle GET request @Override

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

// Set response content type response.setContentType("text/html"); PrintWriter out = response.getWriter(); String data = request.getParameter("data");

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

out.println("<h2>GET Request Received</h2>");

if (data != null && !data.isEmpty()) {

out.println("<p>Data received via GET method: " + data + "</p>");

} else {

out.println("<p>No data received in GET request.</p>");

}

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

}

// Handle POST request @Override

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

// Set response content type response.setContentType("text/html"); PrintWriter out = response.getWriter(); String data = request.getParameter("data"); out.println("<html><body>"); out.println("<h2>POST Request Received</h2>");

if (data != null && !data.isEmpty()) {

out.println("<p>Data received via POST method: " + data + "</p>");

} else {

out.println("<p>No data received in POST request.</p>");

}

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

}

}

web.xml :

<web-app xmlns="<http://java.sun.com/xml/ns/javaee>" xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>" xsi:schemaLocation="<http://java.sun.com/xml/ns/javaee>

<http://java.sun.com/xml/ns/javaee/web-app_3_1.xsd>" version="3.1">

<servlet>

<servlet-name>DemoServlet</servlet-name>

<servlet-class>DemoServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>DemoServlet</servlet-name>

<url-pattern>/DemoServlet</url-pattern>

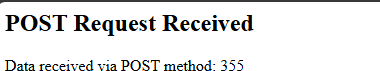
</servlet-mapping>

</web-app>

**OUTPUT :**

****



****

### RESULT :

Thus the give program is executed successfully.

|  |  |
| --- | --- |
| **EXP:07** | **USER PREFERENCE COOKIES** |

AIM:

Write a Servlet program to store a users preferences (like theme or language) using cookies. Retrieve and display these preferences on subsequent visits.

CODE:

package com.example;

import jakarta.servlet.ServletException; import jakarta.servlet.annotation.WebServlet; import jakarta.servlet.http.Cookie;

import jakarta.servlet.http.HttpServlet; import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse; import java.io.IOException;

import java.io.PrintWriter;

/\*\*

\* Servlet implementation class PreferenceServlet

\*/

*@WebServlet*("/preferences")

public class PreferenceServlet extends HttpServlet { private static final long ***serialVersionUID*** = 1L;

/\*\*

\* **@see** HttpServlet#HttpServlet()

\*/

public PreferenceServlet() { super();

// **TODO** Auto-generated constructor stub

}

/\*\*

\* **@see** HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

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

// **TODO** Auto-generated method stub

//response.getWriter().append("Served at: ").append(request.getContextPath());

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

// Retrieve existing cookies String theme = "Default"; String language = "English";

Cookie[] cookies = request.getCookies(); if (cookies != null) {

for (Cookie cookie : cookies) {

if (cookie.getName().equals("theme")) { theme = cookie.getValue();

}

if (cookie.getName().equals("language")) { language = cookie.getValue();

}

}

}

// Display preferences out.println("<html><body>"); out.println("<h2>Your Preferences</h2>"); out.println("<p>Theme: " + theme + "</p>"); out.println("<p>Language: " + language + "</p>");

// Form to set preferences out.println("<form method='post'>");

out.println("Select Theme: <select name='theme'>"

+ "<option>Light</option>"

+ "<option>Dark</option>"

+ "<option>Colorful</option>"

+ "</select><br><br>");

out.println("Select Language: <select name='language'>"

+ "<option>English</option>"

+ "<option>Hindi</option>"

+ "<option>Spanish</option>"

+ "</select><br><br>");

out.println("<input type='submit' value='Save Preferences'>"); out.println("</form>");

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

}

/\*\*

\* **@see** HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

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

// **TODO** Auto-generated method stub

//doGet(request, response);

// Read user inputs

String theme = request.getParameter("theme"); String language = request.getParameter("language");

// Create cookies

Cookie

themeCookie = new Cookie("theme", theme);

Cookie languageCookie = new Cookie("language", language);

// Set cookie expiry (e.g., 7 days) themeCookie.setMaxAge(7 \* 24 \* 60 \* 60);

languageCookie.setMaxAge(7 \* 24 \* 60 \* 60);

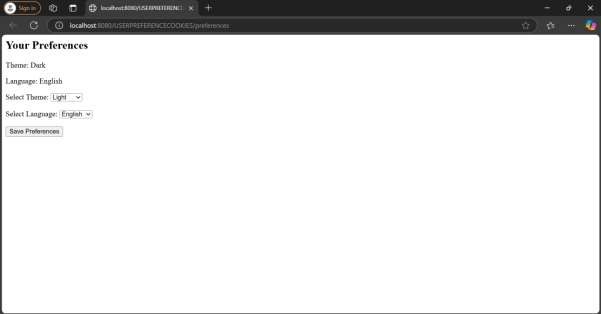
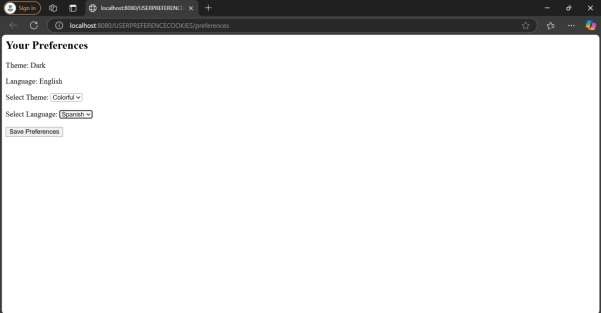
// Add cookies to response response.addCookie(themeCookie); response.addCookie(languageCookie);

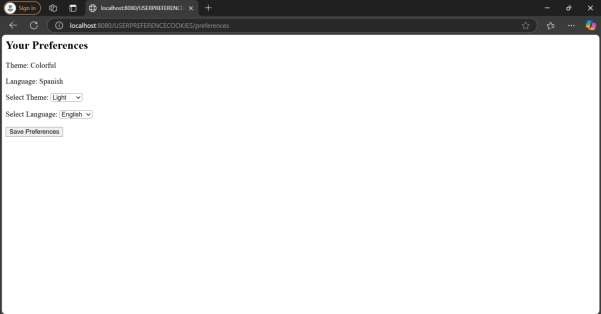
// Redirect to GET method to display updated preferences response.sendRedirect("preferences");

}

}

OUTPUT:





RESULT: Thus user preference cookies was implemented successfully.

|  |  |
| --- | --- |
| **EXP NO:8** | **LIBRARY MANAGEMENT SYSTEM** |

AIM:

Consider a Library Management System. Develop a JavaScript program that will validate the controls in

the forms you have created for the application. State the assumptions you make (business logic you are

taking into consideration). Note: Your application must access a database using Servlet/JSP.

CODE:

<!DOCTYPE html>

<html>

<head>

<title>Library Form</title>

<script>

function validateForm() {

let bookId = document.getElementById("bookId").value;

let title = document.getElementById("title").value;

if (bookId === "" || isNaN(bookId)) {

alert("Book ID must be a number.");

return false;

}

if (title.length < 3) {

alert("Book title must be at least 3 characters.");

return false;

}

return true;

}

</script>

</head>

<body>

<h2>Add Book</h2>

<form action="LibraryServlet" method="post" onsubmit="return validateForm()">

Book ID: <input type="text" name="bookId" id="bookId"><br><br>

Book Title: <input type="text" name="title" id="title"><br><br>

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

</form>

</body>

</html>  
  
  
package com.example;

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class LibraryServlet extends HttpServlet {

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

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

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

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

out.println("<h2>Book Submitted Successfully!</h2>");

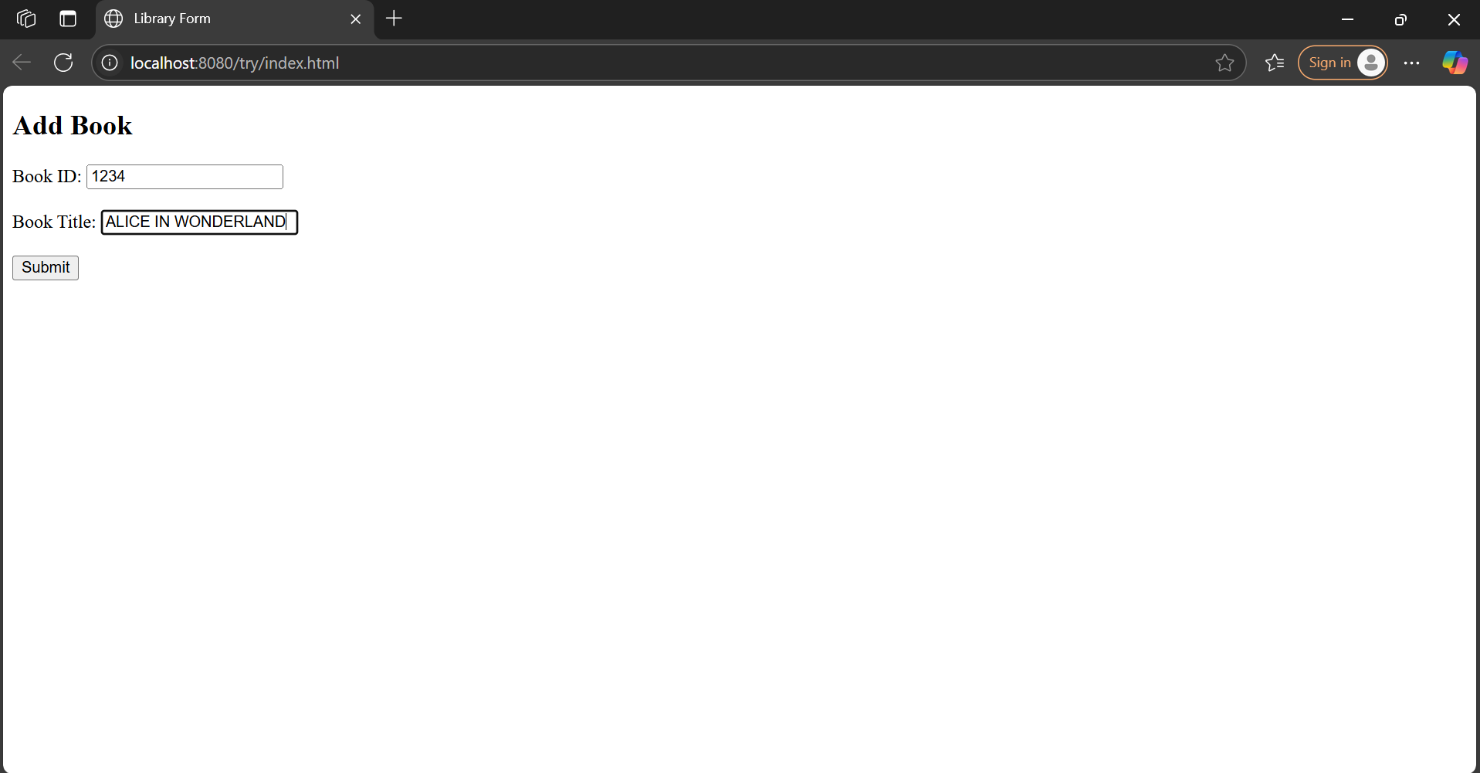
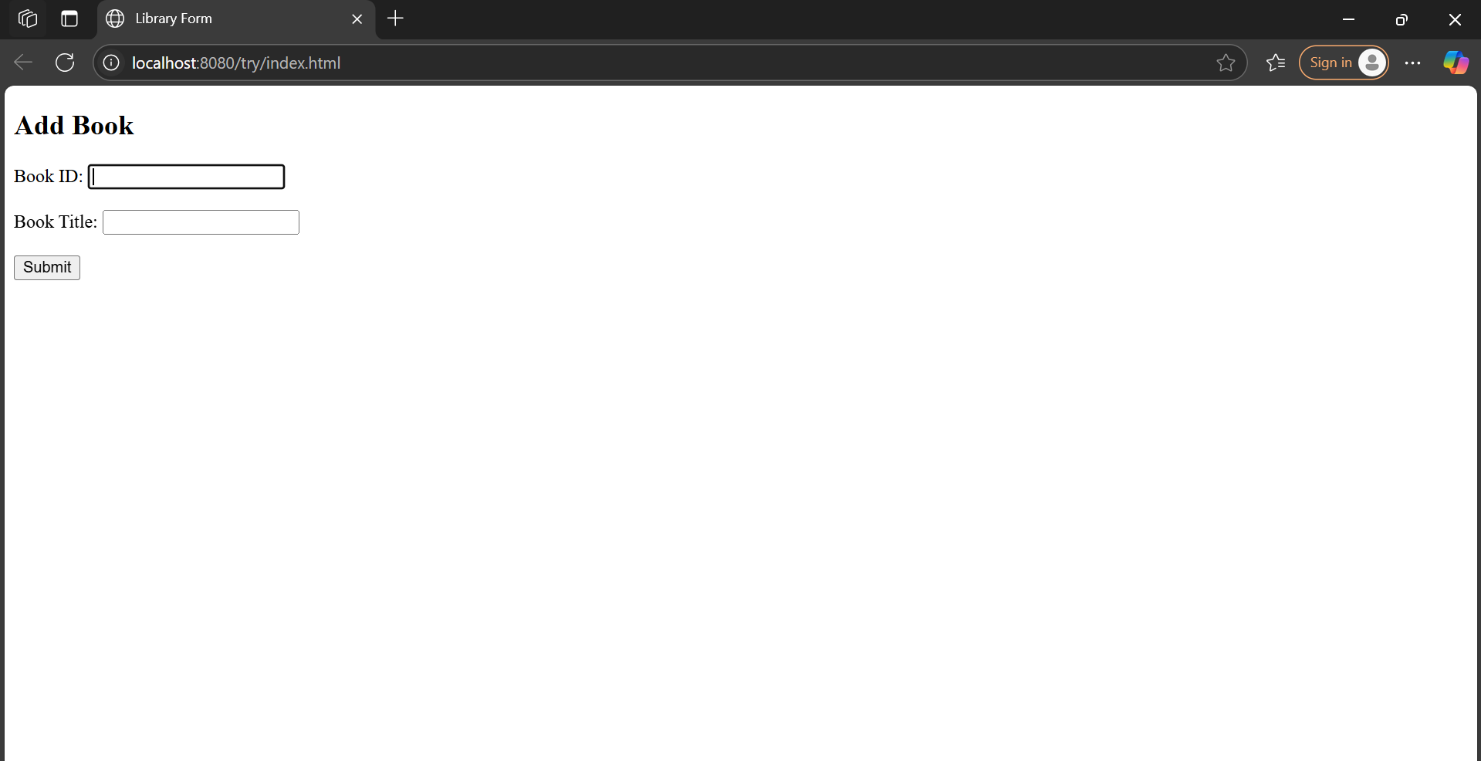
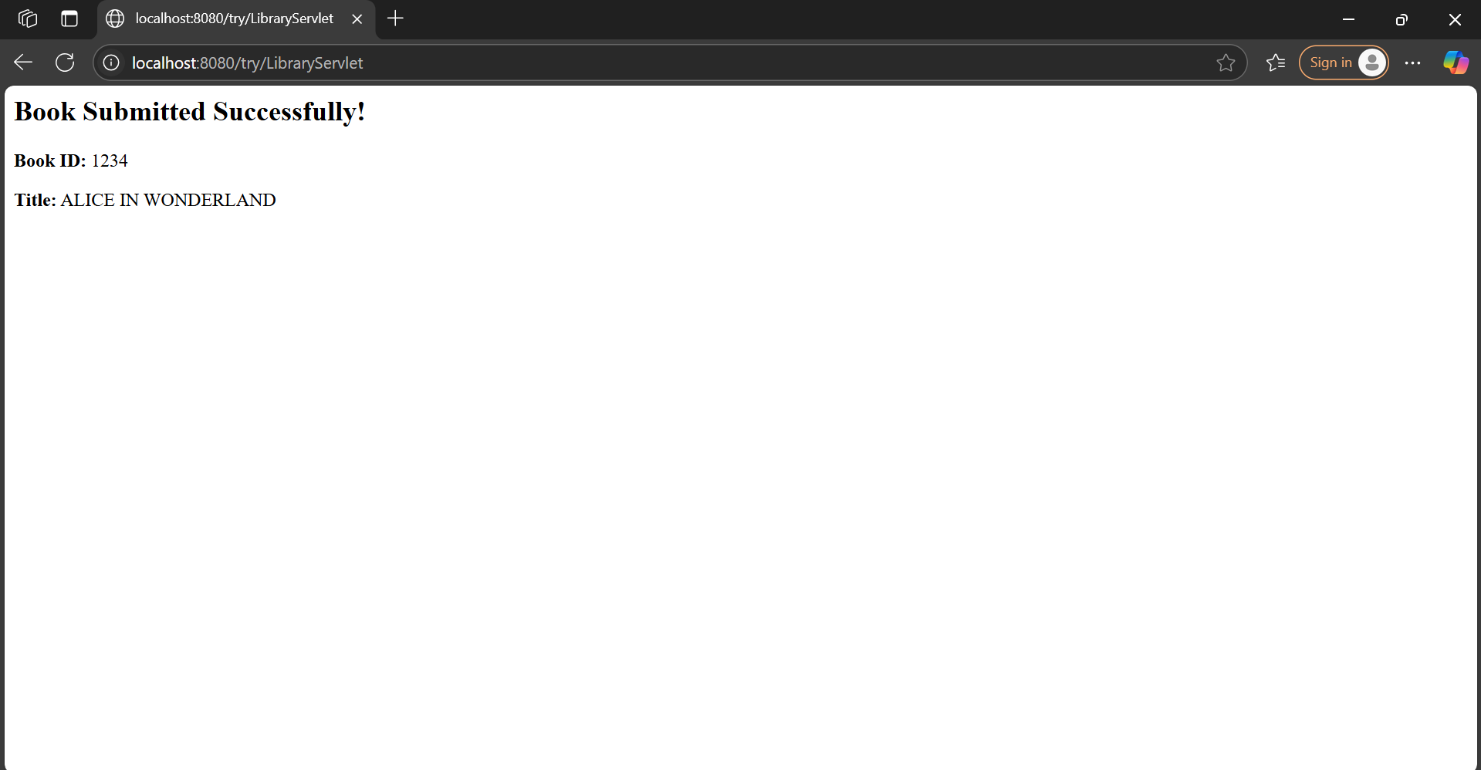
out.println("<p><strong>Book ID:</strong> " + bookId + "</p>");

out.println("<p><strong>Title:</strong> " + title + "</p>");

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

}

}  
  
  
  
  
OUTPUT:



RESULT: THUS, CODE IMPLEMENTED SUCCESSFULLY.

|  |  |
| --- | --- |
| **EXP:09** | **CALCULATOR APPLICATION** |

AIM: To create an application using Kotlin to perform basic calculator operations like addition, subtraction, multiplication and deletion.

CODE:

MainActivity.kt

package com.example.simplecalculator

import android.os.Bundle import android.widget.Toast

import androidx.activity.enableEdgeToEdge import androidx.appcompat.app.AppCompatActivity import androidx.core.view.ViewCompat

import androidx.core.view.WindowInsetsCompat

import com.example.simplecalculator.databinding.ActivityMainBinding

class MainActivity : AppCompatActivity() {

private lateinit var binding : ActivityMainBinding override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

binding=ActivityMainBinding.inflate(*layoutInflater*) setContentView(binding.*root*)

binding.addbutton.setOnClickListener **{** calculatelogic(1) **}** binding.subtractbutton.setOnClickListener **{** calculatelogic(2) **}** binding.multiplybutton.setOnClickListener **{** calculatelogic(3) **}** binding.divisionbutton.setOnClickListener **{** calculatelogic(4) **}**

}

fun calculatelogic(operation : Int){

var firstNumberText=binding.firstnumber.*text*.toString() var secondNumberText=binding.secondnumber.*text*.toString()

if(firstNumberText.*isEmpty*()||secondNumberText.*isEmpty*())

{

Toast.makeText(this,"Fill first and second number",Toast.*LENGTH\_SHORT*).show()

return

}

var firstNumber=firstNumberText.*toIntOrNull*() var secondNumber=secondNumberText.*toIntOrNull*() if(firstNumber != null && secondNumber!=null){

var result=when(operation){

1. -> firstNumber+secondNumber
2. -> firstNumber-secondNumber
3. -> firstNumber\*secondNumber
4. -> if(secondNumber!=0)firstNumber/secondNumber else{ Toast.makeText(this,"You cannot divide a number by

0",Toast.*LENGTH\_SHORT*).show()

return

}

else -> 0

}

binding.resultText.*text*=result.toString()

}

else{

Toast.makeText(this, "Please add only numbers",Toast.*LENGTH\_SHORT*).show()

}

}

}

activity\_main.xml

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

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android[="http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android)

xmlns:app[="http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) xmlns:tools[="http://schemas.android.com/tools"](http://schemas.android.com/tools) android:id="@+id/main" android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:context=".MainActivity">

<EditText

android:id="@+id/firstnumber" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:hint="First Number" android:layout\_marginTop="100dp" android:layout\_marginStart="50dp" android:layout\_marginEnd="50dp" app:layout\_constraintStart\_toStartOf="parent" app:layout\_constraintEnd\_toEndOf="parent" app:layout\_constraintTop\_toTopOf="parent"

/>

<EditText

android:id="@+id/secondnumber" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:hint="Second Number" android:layout\_marginTop="20dp" android:layout\_marginStart="50dp" android:layout\_marginEnd="50dp" app:layout\_constraintStart\_toStartOf="parent" app:layout\_constraintEnd\_toEndOf="parent" app:layout\_constraintTop\_toBottomOf="@id/firstnumber"

/>

<GridLayout

android:id="@+id/operationLayout" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_marginTop="64dp" app:layout\_constraintEnd\_toEndOf="parent" app:layout\_constraintHorizontal\_bias="0.353" app:layout\_constraintStart\_toStartOf="parent" app:layout\_constraintTop\_toBottomOf="@id/secondnumber">

<Button

android:id="@+id/addbutton" android:layout\_width="100dp" android:layout\_height="wrap\_content" android:layout\_row="0" android:layout\_column="0" android:layout\_marginStart="60dp" android:text="+" android:textSize="30sp" />

<Button

android:id="@+id/subtractbutton" android:layout\_width="100dp" android:layout\_height="wrap\_content" android:layout\_row="0" android:layout\_column="1" android:layout\_marginStart="20dp" android:text="-" android:textSize="30sp" />

<Button

android:id="@+id/multiplybutton" android:layout\_width="100dp" android:layout\_height="wrap\_content" android:layout\_row="1" android:layout\_column="0" android:layout\_marginStart="60dp" android:layout\_marginTop="30dp" android:text="x" android:textSize="30sp" />

<Button

android:id="@+id/divisionbutton" android:layout\_width="100dp" android:layout\_height="wrap\_content" android:layout\_row="1" android:layout\_column="1" android:layout\_marginStart="20dp" android:layout\_marginTop="30dp" android:text="/" android:textSize="30sp"/>

</GridLayout>

<TextView

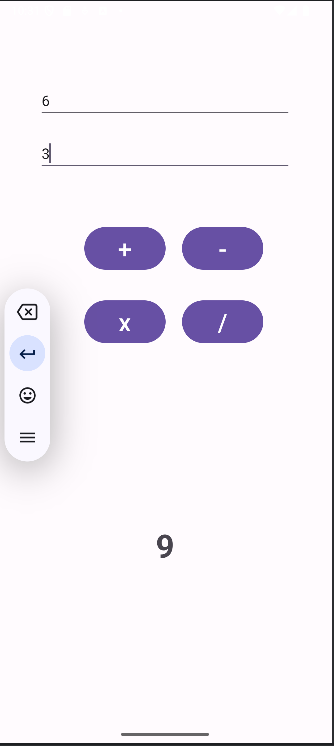
android:id="@+id/resultText" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="Result: " android:textSize="40sp" android:textStyle="bold" app:layout\_constraintStart\_toStartOf="parent" app:layout\_constraintEnd\_toEndOf="parent"

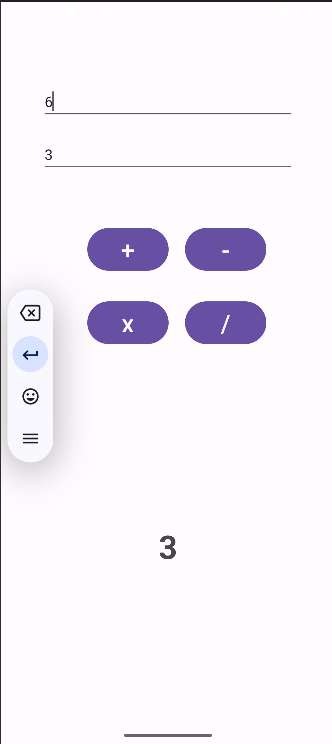
app:layout\_constraintBottom\_toBottomOf="parent" app:layout\_constraintTop\_toBottomOf="@id/operationLayout"

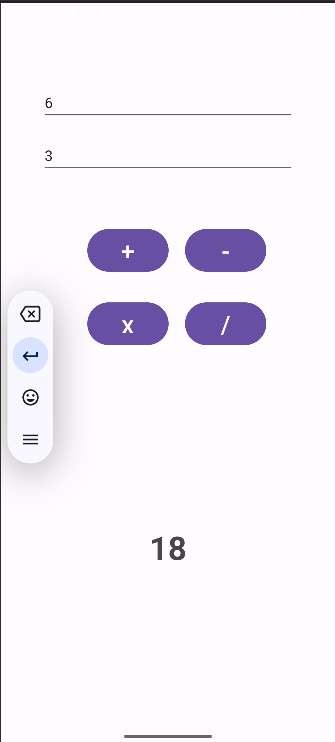
/>

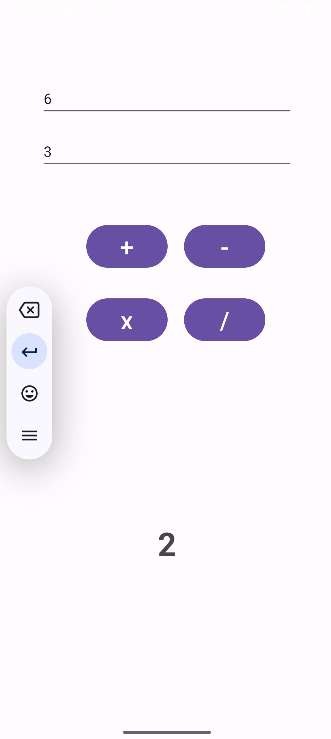
</androidx.constraintlayout.widget.ConstraintLayout>

OUTPUT:









Result: APP SUCCESSFULLY DONE.

|  |  |
| --- | --- |
| **EXP:10** | **CHANGE BACKGROUND COLOUR AND FONT STYLE** |

AIM:

Develop an application to change the font and colour of the text and display toast message when the user presses the button.

CODE:

activity\_main.xml

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

<RelativeLayout xmlns:android[="http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) android:id="@+id/mainLayout"

android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:padding="16dp" android:background="#FFFFFF">

<TextView

android:id="@+id/myText" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="Hello Kotlin!" android:textSize="24sp" android:layout\_centerInParent="true" android:textStyle="normal" />

<Button

android:id="@+id/changeButton" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:layout\_below="@id/myText" android:layout\_centerHorizontal="true" android:layout\_marginTop="20dp" android:text="Change Style" />

</RelativeLayout>

MainActivity.kt

package com.example.myapplication

import android.graphics.Color import android.graphics.Typeface import android.os.Bundle

import androidx.appcompat.app.AppCompatActivity import android.widget.Button

import android.widget.RelativeLayout import android.widget.TextView

class MainActivity : AppCompatActivity() { private var toggled = false

override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.*activity\_main*)

val layout = findViewById<RelativeLayout>(R.id.*mainLayout*) val text = findViewById<TextView>(R.id.*myText*)

val button = findViewById<Button>(R.id.*changeButton*)

button.setOnClickListener **{**

if (!toggled) { layout.setBackgroundColor(Color.parseColor("#FFEB3B")) //

Yellow

text.setTypeface(null, Typeface.*BOLD\_ITALIC*)

} else {

layout.setBackgroundColor(Color.*WHITE*) text.setTypeface(null, Typeface.*NORMAL*)

}

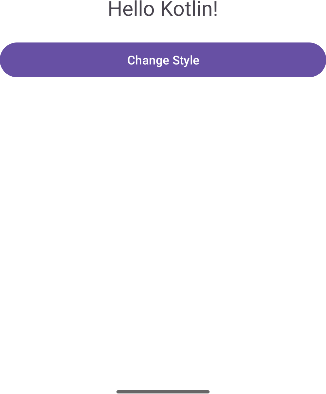
toggled = !toggled

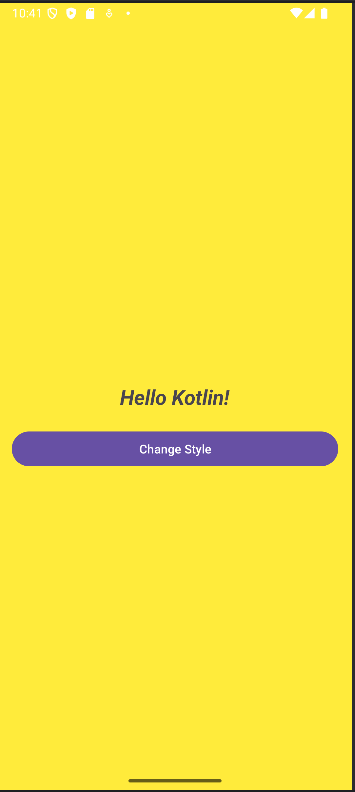
**}**

}

}

Output:





Result: Thus, background colour and font style was changed on click, successfully

|  |  |
| --- | --- |
| **EXP:11** | **SD CARD** |

AIM: TO BUILD AN APPLICATION USING KOTLIN TO WRITE INTO SD. CODE:

activity\_main.xml

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

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android[="http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android)

xmlns:app[="http://schemas.android.com/apk/res-auto"](http://schemas.android.com/apk/res-auto) xmlns:tools[="http://schemas.android.com/tools"](http://schemas.android.com/tools) android:id="@+id/main" android:layout\_width="match\_parent" android:layout\_height="match\_parent" tools:context=".MainActivity">

<EditText

android:id="@+id/editText" android:layout\_width="0dp" android:layout\_height="wrap\_content" android:layout\_margin="16dp" android:layout\_marginTop="200dp" android:hint="Type something to save" app:layout\_constraintEnd\_toEndOf="parent" app:layout\_constraintStart\_toStartOf="parent" app:layout\_constraintTop\_toTopOf="parent" />

<Button

android:id="@+id/saveButton" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:layout\_marginTop="168dp" android:text="Save to File" app:layout\_constraintEnd\_toEndOf="parent" app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@id/editText" />

<TextView

android:id="@+id/statusTextView" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:text="" app:layout\_constraintTop\_toBottomOf="@id/saveButton" android:textColor="@android:color/black" android:textSize="16sp" tools:layout\_editor\_absoluteX="37dp" tools:layout\_editor\_absoluteY="340dp" />

</androidx.constraintlayout.widget.ConstraintLayout>

AndroidManifest.xml

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

<manifest xmlns:android[="http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) package="com.example.sd">

<application

android:allowBackup="true" android:icon="@mipmap/ic\_launcher" android:label="@string/app\_name" android:theme="@style/Theme.AppCompat.DayNight.NoActionBar">

<activity android:name=".MainActivity" android:exported="true">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

</application>

</manifest>

MainActivity.kt

package com.example.sd

import android.Manifest

import android.content.ContentValues import android.content.pm.PackageManager import android.os.Build

import android.os.Bundle import android.os.Environment

import android.provider.MediaStore import android.widget.Button import android.widget.EditText import android.widget.TextView import android.widget.Toast

import androidx.appcompat.app.AppCompatActivity import androidx.core.app.ActivityCompat

import androidx.core.content.ContextCompat import java.io.OutputStream

class MainActivity : AppCompatActivity() { private val REQUEST\_CODE = 101

private lateinit var statusTextView: TextView // Declare a reference to the TextView

override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.*activity\_main*)

val editText = findViewById<EditText>(R.id.*editText*) val saveButton = findViewById<Button>(R.id.*saveButton*)

statusTextView = findViewById(R.id.*statusTextView*) // Initialize the TextView

saveButton.setOnClickListener **{**

val text = editText.*text*.toString()

if (checkPermission()) {

saveToFile(text) // Save to shared storage using MediaStore

} else {

requestPermission()

}

**}**

}

private fun saveToFile(data: String) {

// Prepare the content values for the file to be created val contentValues = ContentValues().*apply* **{**

put(MediaStore.MediaColumns.*DISPLAY\_NAME*, "myfile.txt") // File

name

put(MediaStore.MediaColumns.*MIME\_TYPE*, "text/plain") put(MediaStore.MediaColumns.*RELATIVE\_PATH*,

Environment.*DIRECTORY\_DOCUMENTS*) // or Environment.DIRECTORY\_DOWNLOADS for Downloads folder

**}**

// Get content resolver and insert into MediaStore val resolver = *contentResolver*

val uri = resolver.insert(MediaStore.Files.getContentUri("external"), contentValues)

uri?.*let* **{**

try {

// Open an output stream to write the data val outputStream: OutputStream? =

resolver.openOutputStream(**it**)

outputStream?.write(data.*toByteArray*()) outputStream?.close()

TextView

// Show success in the TextView

statusTextView.*text* = "File saved successfully!" // Update

Toast.makeText(this, "File saved successfully!",

Toast.*LENGTH\_LONG*).show() // Optional Toast

} catch (e: Exception) {

// Handle any errors

statusTextView.*text* = "Error: ${e.message}" // Update TextView with error message

Toast.makeText(this, "Error: ${e.message}", Toast.*LENGTH\_LONG*).show()

}

**}** ?: *run* **{**

// Handle failure if URI is null

statusTextView.*text* = "Failed to create file" // Update TextView with failure message

Toast.makeText(this, "Failed to create file", Toast.*LENGTH\_LONG*).show()

**}**

}

private fun checkPermission(): Boolean {

return if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*Q*) {

// No need to request storage permission on Android 10+ for app- specific storage

true

} else {

// For Android versions below API 29 (Android 9 and below), check WRITE\_EXTERNAL\_STORAGE permission

val result = ContextCompat.checkSelfPermission( this, Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*

)

result == PackageManager.*PERMISSION\_GRANTED*

}

}

private fun requestPermission() {

// Request storage permission if needed (for Android versions below

10)

ActivityCompat.requestPermissions( this,

*arrayOf*(Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*), REQUEST\_CODE

)

}

override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<String>, grantResults: IntArray) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults)

if (requestCode == REQUEST\_CODE) {

if (grantResults.*isNotEmpty*() && grantResults[0] == PackageManager.*PERMISSION\_GRANTED*) {

Toast.makeText(this, "Permission Granted", Toast.*LENGTH\_SHORT*).show()

} else {

Toast.makeText(this, "Permission Denied", Toast.*LENGTH\_SHORT*).show()

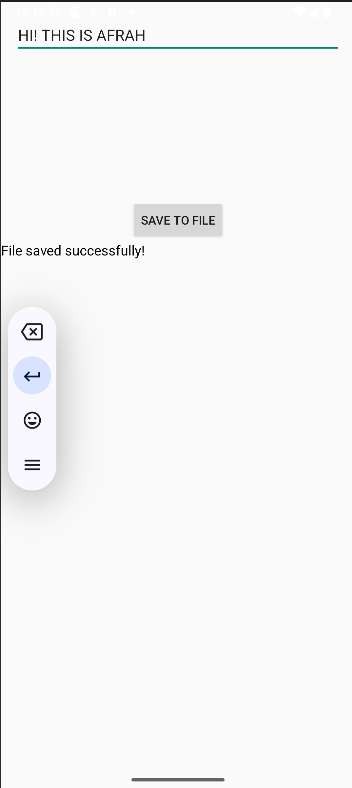
}

}

}

}

OUTPUT:



RESULT: APP BUILD SUCCESSFULLY.