**RAJALAKSHMI ENGINEERING COLLEGE**

###### An Autonomous Institution, Affiliated to Anna University Rajalakshmi Nagar, Thandalam – 602 105

****

**LAB MANUAL**

**AI23431 Web Technology and Mobile Application**

**Name**

: . . . . . . . . . . . . SHARAN.M. . . . . . . . . . . . . . . . . . . . . . . . . .

**Register No**.

: . . . . . .. . . . . . . . . .231501151 . . . . . . . . . . . . . . . . . . . . . . . .

**Year / Branch / Section** : . . . . . . . . . . . . . . . 2/B.Tech AIML/AC. . . . . . . . . . . . . . . . . . . .

**Semester**

: . . . . . . . . . . . . . . . . . . . . 4-nd. . . . . . . . . . . . . . . . . . . . . . . . .

**Academic Year**

: . . . . . . . . . . . . . . . .2024-2025........... . . . . . . . . . . . .

**RAJALAKSHMI ENGINEERING COLLEGE**

###### An Autonomous Institution, Affiliated to Anna University Rajalakshmi Nagar, Thandalam – 602 105

**BONAFIDE CERTIFICATE**

Name ……………SHARAN.M…………………………….... Academic Year….2024-2025..Semester….IV..Branch..AIML..

UNIVERSITY REGISTER No.

2116231501151

Certified that this is the Bonafide record of work done by above student in the **AI23431-Web Technology and Mobile Application** Laboratory during the year 2024-2025

Signature of Faculty -in-Charge

Submitted for the Practical Examination held on ………………………..

Internal Examiner External Examiner

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Title** | **Page No.** |
| 1 | Create a Web Page to Embed a Map with Hotspots, Frames & Links | 4 |
| 2 | JavaScript Form Validation (Registration Page) | 9 |
| 3 | Servlet Program to Display "Hello World" | 14 |
| 4 | Web Form using Servlet to Accept Name & Age | 18 |
| 5 | HTTP GET vs POST using Java Servlet | 22 |
| 6 | Session Tracking using HttpSession in Servlet | 26 |
| 7 | Servlet Project: Store User Preferences using Cookies | 30 |
| 8 | Android App – Library Management System | 34 |
| 9 | Android App – Calculator App with Basic Operations | 38 |
| 10 | Android App – Font & Color Changer | 42 |

**EXP: 1 HTML & CSS**

## Create a Web Page to Embed a Map with Hotspots, Frames & Links

##### AIM:

Create a Web Page to Embed a Map with Hotspots, Frames & Links.

##### ALGORITHM:

**Step 1: Create an HTML file (index.html).**

* + Define the document structure using <html>, <head>, and <body>.
  + Set the page title and include internal CSS for basic styling.

### Step 2: Embed an Image Map.

* + Use the <img> tag to insert an image (map).
  + Define a <map> element with a name attribute.
  + Add <area> elements inside the <map> with different **shapes** (rectangle, circle, polygon).
  + Assign **href** attributes to the <area> elements to make them clickable.

### Step 3: Create Hyperlinks.

* + Add <a> tags that allow navigation to different pages.
  + Use the target attribute to open the linked pages in a frame.

### Step 4: Add an Inline Frame (iframe).

* + Use the <iframe> tag to display linked pages within the same webpage.
  + Set the name attribute for the iframe to target it from links.

**Step 5: Create Additional Pages (page1.html, page2.html).**

* + Define a simple HTML structure.
  + Apply **CSS styles** for a visually appealing design.
  + Test the project to ensure the map hotspots and frames work correctly.

##### PROGRAM:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Subject Entry Form</title>

<style>

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

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

}

body {

background-color: #f4f4f4; display: flex;

justify-content: center; align-items: center; height: 100vh;

}

.container { width: 400px;

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

box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.2);

}

h2 {

text-align: center; color: #333;

margin-bottom: 20px;

}

label {

font-weight: bold; color: #555;

}

input, select { width: 100%; padding: 10px; margin-top: 5px;

margin-bottom: 15px; border: 1px solid #ccc; border-radius: 5px; font-size: 16px;

}

button {

width: 100%; padding: 12px; background: #007bff; color: white;

border: none;

border-radius: 5px; font-size: 18px; cursor: pointer; transition: 0.3s;

}

button:hover { background: #0056b3;

}

.message {

margin-top: 15px; text-align: center; color: green;

font-weight: bold;

}

</style>

</head>

<body>

<div class="container">

<h2>📘 Subject Entry Form</h2>

<form id="subjectForm">

<label for="subjectCode">Subject Code:</label>

<input type="text" id="subjectCode" placeholder="Enter Subject Code" required>

<label for="subjectName">Subject Name:</label>

<input type="text" id="subjectName" value="WEB TECHNOLOGY AND MOBILE APPLICATION" readonly>

<label for="category">Category:</label>

<select id="category">

<option value="AI-DS & AI-ML">Artificial Intelligence & Data Science / Artificial Intelligence & Machine Learning</option>

</select>

<button type="submit">Submit</button>

</form>

<p class="message" id="message"></p>

</div>

<script>

document.getElementById("subjectForm").addEventListener("submit", function(event)

{

event.preventDefault(); // Prevent form from reloading document.getElementById("message").textContent = "✅ Subject details submitted

successfully!";

});

</script>

</body>

</html>

##### OUTPUT:

****

**RESULT:** The image map with hotspots, frames, and link**s** is successfully created**.**

## Create a Web Page Using Embedded, External & Inline CSS

##### AIM:

Create a Web Page Using Embedded, External & Inline CSS

**ALGORITHM:**

### Step 1: Create an External CSS File (style.css).

* + Define **body, headings, and paragraph styles** in style.css.
  + Save the CSS file in the same directory as index.html.

**Step 2: Create an HTML File (index.html).**

* + Define the structure using <html>, <head>, and <body>.
  + Add a <title> tag for the page title.

### Step 3: Link External CSS.

* + Use <link rel="stylesheet" href="style.css"> inside the <head> tag.

### Step 4: Apply Embedded CSS.

* + Add a <style> section inside <head>.
  + Define styles for a **div box** with width, height, color, and border-radius.

### Step 5: Apply Inline CSS.

* + Use the style attribute in an HTML element (<p>) to apply color and font weight directly.

### Step 6: Display Content.

* + Use headings (<h2>) and paragraphs (<p>) to demonstrate different CSS types.
  + Add a styled **div box** using embedded CSS.

### Step 7: Test the Page.

* + Open the HTML file in a browser to check if all styles (inline, embedded, external) are applied correctly.

##### PROGRAM: INDEX.HTML:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Elegant CSS Styling Example</title>

<link rel="stylesheet" href="styles.css"> <!-- External CSS -->

<style>

/\* Embedded CSS \*/

.embedded-style { color: #007bff; font-size: 20px; font-weight: bold; text-align: center; padding: 10px;

border: 2px solid #007bff; border-radius: 5px; width: 50%;

margin: 20px auto;

box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.1); background: #f0f8ff;

}

</style>

</head>

<body style="background-color: #f8f9fa; font-family: 'Poppins', sans-serif; text-align: center;">

<h1 style="color: #ff5733; font-size: 28px;">🌟 Styled with Inline CSS 🌟</h1>

<p class="embedded-style">✨ This text is styled using Embedded CSS ✨</p>

<p class="external-style">🎨 This text is styled using External CSS 🎨</p>

</body>

</html>

##### STYLE.CSS:

.external-style { color: #28a745; font-size: 22px; font-weight: bold; text-align: center; padding: 10px;

border: 2px solid #28a745; border-radius: 5px;

width: 60%; margin: 20px auto;

box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.1); background: #e9ffe9;

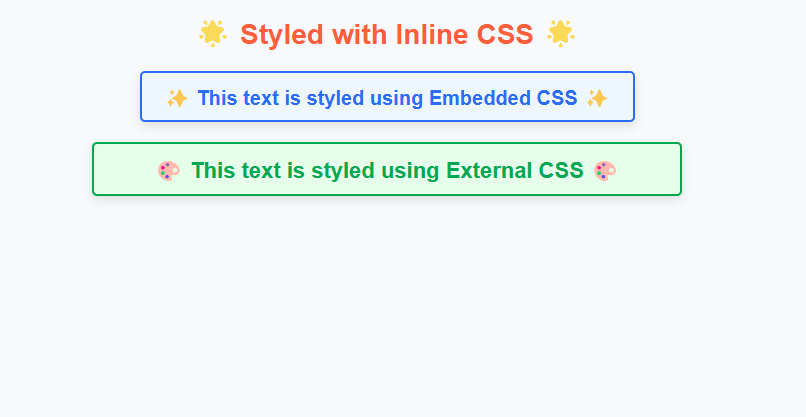
transition: 0.3s;

}

external-style:hover { background: #d4f8d4; transform: scale(1.05);

}

##### OUTPUT:

****

**RESULT:** The webpage demonstrates external, embedded, and inline CSS styles effectively**.**

**EXP: 2 JAVASCRIPT**

**2) 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)

1. **Mobile Number (Phone number should contain 10 digits only).**
2. **Last Name and Address (should not be Empty). AIM:**

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

**ALGORITHM:**

### Step 1: Create the HTML Form

* + Add input fields for First Name, Last Name, Email, Password, Mobile Number, and Address.
  + Add a Submit Button to trigger validation.

### Step 2: Apply Elegant CSS Styling

* + Add responsive styling with hover effects.
  + Improve form layout, input fields, and button styles.

### Step 3: Add JavaScript for Validation

* + Validate each field and show error messages if invalid.
  + Display "Registration Successful!" if all inputs are valid.

### Step 4: Full Working Code in One HTML File

* + Copy and paste the code below into an .html file and open it in a browser.

##### PROGRAM:

<!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: Arial, sans-serif;

background: linear-gradient(135deg, #ff9a9e, #fad0c4); display: flex;

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

}

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

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2); width: 350px;

text-align: center;

}

h2 {

color: #333;

}

input {

width: 100%; padding: 10px; margin: 8px 0;

border: 1px solid #ccc;

border-radius: 5px; font-size: 16px;

}

input:focus {

border-color: #ff6f61; outline: none;

box-shadow: 0 0 5px rgba(255, 111, 97, 0.5);

}

.error {

color: red;

font-size: 14px; text-align: left;

margin-bottom: 10px;

}

button {

width: 100%; padding: 10px; background: #ff6f61; color: white;

border: none; border-radius: 5px; font-size: 16px; cursor: pointer;

}

button:hover { background: #e55b50;

}

</style>

</head>

<body>

<div class="container">

<h2>Register Here</h2>

<form id="registrationForm">

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

<div class="error" id="firstNameError"></div>

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

<div class="error" id="lastNameError"></div>

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

<div class="error" id="emailError"></div>

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

<div class="error" id="passwordError"></div>

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

<div class="error" id="mobileError"></div>

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

<div class="error" id="addressError"></div>

<button type="button" onclick="validateForm()">Register</button>

</form>

</div>

<script>

function validateForm() { let isValid = true;

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

let password = document.getElementById("password").value.trim(); let mobile = document.getElementById("mobile").value.trim();

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

// Clear previous error messages document.getElementById("firstNameError").innerText = ""; document.getElementById("lastNameError").innerText = ""; document.getElementById("emailError").innerText = ""; document.getElementById("passwordError").innerText = ""; document.getElementById("mobileError").innerText = ""; document.getElementById("addressError").innerText = "";

// First Name Validation

if (!/^[A-Za-z]{6,}$/.test(firstName)) { document.getElementById("firstNameError").innerText = "First name must be at

least 6 letters.";

isValid = false;

}

// Last Name Validation if (lastName === "") {

document.getElementById("lastNameError").innerText = "Last name cannot be empty.";

isValid = false;

}

// Email Validation

if (!/^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/.test(email)) {

document.getElementById("emailError").innerText = "Enter a valid email (e.g., [name@domain.com).](mailto:name@domain.com)";

isValid = false;

}

// Password Validation

if (password.length < 6) {

document.getElementById("passwordError").innerText = "Password must be at least 6 characters.";

isValid = false;

}

// Mobile Number Validation if (!/^\d{10}$/.test(mobile)) {

document.getElementById("mobileError").innerText = "Enter a valid 10-digit mobile number.";

isValid = false;

}

// Address Validation if (address === "") {

document.getElementById("addressError").innerText = "Address cannot be empty."; isValid = false;

}

if (isValid) {

alert("Registration Successful!");

}

}

</script>

</body>

</html>

##### OUTPUT:

**RESULT:** The Registration Form was successfully created using HTML, CSS, and JavaScript.

**EXP 3 SERVLET PROGRAM**

## AIM:

To create an HTML webpage that displays "Hello World" in the center of the page inside a styled box using CSS for alignment and styling.

## ALGORITHM:

1. Start
2. Create an HTML document with a <head> and <body> section.
3. Inside the <head> section:
   * Set the title of the page.
   * Add meta tags for character set and viewport settings.
   * Define CSS styles to center the text and style the box.
4. In the <body> section:
   * Use a <div> element with a class box to wrap the text.
   * Inside the <div>, add an <h1> tag with the text "Hello World" in bold.
5. Use **CSS** Flexbox to center the box both vertically and horizontally.
6. Apply **s**tyles to the box, including:
   * Padding
   * Border
   * Border-radius
   * Box-shadow
   * Background color
7. Save the file and open it in a web browser.
8. End.

##### PROGRAM:

<!DOCTYPE html>

<html>

<head>

<title>hello world</title>

<meta charset="UTF-8">

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

<style>

body {

display: flex;

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

height: 100vh; margin: 0;

background-color: #f0f0f0;

}

.box {

padding: 20px; background-color: white; border: 2px solid black; border-radius: 10px;

box-shadow: 5px 5px 15px rgba(0, 0, 0, 0.2); text-align: center;

}

</style>

</head>

<body>

<div class="box">

<h1><b>HELLO WORLD!!!</b></h1>

</div>

</body>

</html>

###### Output:

****

**RESULT:** Thus, the servlet program was successfully executed.

**EXP 4: WEB FORM DEVELOPED USING SERVLET PROGRAM**

## AIM:

To create a web form that accepts a user's name and age, processes the data using JavaScript, and displays the submitted details with a stylish UI and animations.

##### ALGORITHM:

###### Step 1: Start

**Step 2: Design the Web Form**

* Create an HTML form with two input fields:
  + Name (Text Input)
  + Age (Number Input)
* Add a submit button.

###### Step 3: Apply CSS Styles

* Use **gradient background, smooth input effects, button hover effects, and animations**.
* Style the form, inputs, button, and result display area.

###### Step 4: Write JavaScript for Form Handling

* Attach an **event listener** to the form submit event.
* Get user input values (name and age).
* Validate input fields (ensure **name is not empty** and **age is valid**).
* If validation fails, **show an alert**.
* If valid, display the submitted details dynamically.

###### Step 5: Show Output Dynamically

* Use JavaScript to update the UI and **display the submitted data in a formatted box**.
* Use **smooth animations** for better user experience.

**Step 6: End**

##### PROGRAM:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>User Form</title>

<style>

/\* Import Google Font \*/ @import

url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap');

/\* Page Styling \*/ body {

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

background: linear-gradient(135deg, #ff9a9e, #fad0c4); display: flex;

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

animation: fadeIn 1s ease-in-out;

}

/\* Form Container \*/

.container { background: white; padding: 25px; border-radius: 12px;

box-shadow: 0px 5px 15px rgba(0, 0, 0, 0.2); width: 350px;

text-align: center;

transition: transform 0.3s ease-in-out;

}

.container:hover { transform: scale(1.05);

}

h2 {

color: #ff4d6d; font-weight: 600;

margin-bottom: 15px;

text-shadow: 1px 1px 2px rgba(0, 0, 0, 0.2);

}

/\* Input Fields \*/ label {

font-weight: 600;

color: #444; display: block; margin: 10px 0 5px;

}

input {

width: 100%; padding: 10px;

border: 2px solid #ff4d6d; border-radius: 8px;

font-size: 16px; outline: none;

transition: all 0.3s ease-in-out;

}

input:focus {

border-color: #ff165d;

box-shadow: 0px 0px 10px rgba(255, 22, 93, 0.4);

}

/\* Submit Button \*/ button {

background: #ff4d6d; color: white;

border: none; padding: 12px; cursor: pointer; width: 100%; font-size: 18px; font-weight: 600;

border-radius: 8px; margin-top: 15px;

transition: background 0.3s ease-in-out, transform 0.2s;

}

button:hover { background: #ff165d; transform: scale(1.05);

}

button:active { transform: scale(0.95);

}

/\* Output Box \*/

.output { display: none;

margin-top: 20px; padding: 15px; border-radius: 8px; background: #e0f7fa; color: #00796b;

font-size: 16px; text-align: left;

box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.1); animation: slideIn 0.5s ease-in-out;

}

/\* Animations \*/ @keyframes fadeIn {

from { opacity: 0; transform: translateY(-10px); } to { opacity: 1; transform: translateY(0); }

}

@keyframes slideIn {

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

}

</style>

</head>

<body>

<div class="container">

<h2>Enter Your Details</h2>

<form id="userForm">

<label for="name">Name:</label>

<input type="text" id="name" name="name" required placeholder="Enter your name...">

<label for="age">Age:</label>

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

<button type="submit">Submit</button>

</form>

<div class="output" id="output"></div>

</div>

<script>

document.getElementById("userForm").addEventListener("submit", function(event) { event.preventDefault(); // Prevent form submission

let name = document.getElementById("name").value.trim(); let age = document.getElementById("age").value.trim();

if (name === "" || age === "" || age <= 0) { alert("⚠◻ Please enter a valid name and age."); return;

}

// Display the submitted data with an animation

let outputBox = document.getElementById("output"); outputBox.innerHTML =

`<h3>✅ Submitted Details</h3>

<p><b>Name:</b> ${name}</p>

<p><b>Age:</b> ${age}</p>`; outputBox.style.display = "block";

});

</script>

</body>

</html>

##### OUTPUT:

**RESULT:** The creation of web form using servlet program was executed successfully.

**EXP 5 HTTP GET and POST methods**

## AIM:

To demonstrate the difference between HTTP GET and POST methods using a Java Servlet by creating a form that sends data using both methods.

## ALGORITHM:

Step 1: Create an HTML form with GET and POST methods.

Step 2: Develop a Servlet (GetPostServlet.java) to handle requests. Step 3: Implement doGet() to handle GET requests.

Step 4: Implement doPost() to handle POST requests.

Step 5: Configure web.xml to map the Servlet to /GetPostServlet. Step 6: Run the server and observe how GET and POST work differently. **PROGRAM:**

###### index.html (Form Page)

html CopyEdit

<!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 Demo</title>

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

</head>

<body>

<div class="container">

<h2>Choose GET or POST Method 🚀</h2>

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

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

<button type="submit">Submit with GET</button>

</form>

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

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

<button type="submit">Submit with POST</button>

</form>

</div>

</body>

</html>

###### styles.css

css CopyEdit body {

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

background: linear-gradient(to right, #36D1DC, #5B86E5); text-align: center;

display: flex;

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

}

.container { background: #fff; padding: 20px; border-radius: 15px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2); width: 300px;

}

h2 {

color: #5B86E5;

}

input {

width: 90%; padding: 10px; margin: 10px 0;

border: 1px solid #5B86E5; border-radius: 8px;

text-align: center;

}

button {

background: #36D1DC; color: white;

border: none; padding: 10px 15px; border-radius: 8px; cursor: pointer; transition: 0.3s;

}

button:hover {

background: #1e6e8c;

}

###### GetPostServlet.java (Servlet to Handle GET & POST)

java CopyEdit

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("/GetPostServlet")

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

// Handling GET Request

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

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

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

out.println("<html><body>"); out.println("<h2>GET Method Received</h2>");

out.println("<p>Hello, <b>" + name + "</b>! Data sent using GET.</p>"); out.println("</body></html>");

}

// Handling POST Request

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

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

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

out.println("<html><body>"); out.println("<h2>POST Method Received</h2>");

out.println("<p>Hello, <b>" + name + "</b>! Data sent using POST.</p>"); out.println("</body></html>");

}

}

###### web.xml (Servlet Configuration)

xml

CopyEdit

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

<servlet>

<servlet-name>GetPostServlet</servlet-name>

<servlet-class>GetPostServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>GetPostServlet</servlet-name>

<url-pattern>/GetPostServlet</url-pattern>

</servlet-mapping>

</web-app>

##### OUTPUT:

**RESULT:** The Get and Post methods were executed successfully.

**EXP 6 Demonstrate Session Tracking using HttpSession**

## AIM:

To implement a simple login system using **HttpSession** to track user sessions in a Java Servlet.

## ALGORITHM:

**Step 1:** Create an HTML **Login Form** to take **username & password** as input. **Step 2:** Develop a **Servlet (**LoginServlet.java**)** to handle login requests. **Step 3:** Validate the user credentials and start an **HttpSession**.

**Step 4:** Redirect users to the **DashboardServlet**, where session details are displayed.

**Step 5:** Provide a **LogoutServlet** to end the session.

**Step 6:** Run the server and test session tracking.

## PROGRAM:

#### index.html (Login Page)

html CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Login Page</title>

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

</head>

<body>

<div class="container">

<h2>Login System with Session Tracking</h2>

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

<input type="text" name="username" placeholder="Enter Username"

required>

<input type="password" name="password" placeholder="Enter

Password" required>

<button type="submit">Login</button>

</form>

</div>

</body>

</html>

#### styles.css

css CopyEdit body {

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

background: linear-gradient(to right, #fbc2eb, #a6c1ee); text-align: center;

display: flex;

justify-content: center; align-items: center; height: 100vh;

margin: 0;

}

.container {

background: #fff; padding: 20px; border-radius: 15px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2); width: 300px;

}

h2 {

}

color: #5B86E5;

input {

width: 90%; padding: 10px; margin: 10px 0;

border: 1px solid #5B86E5; border-radius: 8px;

text-align: center;

}

button {

background: #36D1DC; color: white; border: none; padding: 10px 15px; border-radius: 8px; cursor: pointer; transition: 0.3s;

}

button:hover { background: #1e6e8c;

}

#### LoginServlet.java (Login & Session Creation)

java CopyEdit

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; import javax.servlet.http.HttpSession;

@WebServlet("/LoginServlet")

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");

// Simple validation (In real apps, use a database) if(username.equals("admin") && password.equals("password")) {

HttpSession session = request.getSession(); session.setAttribute("user", username); response.sendRedirect("DashboardServlet"); // Redirect to

Dashboard

} else {

out.println("<h3>Invalid Credentials! Try Again.</h3>"); request.getRequestDispatcher("index.html").include(request,

response);

}

}

}

#### DashboardServlet.java (Session Tracking & Dashboard)

java CopyEdit

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; import javax.servlet.http.HttpSession;

@WebServlet("/DashboardServlet")

public class DashboardServlet extends HttpServlet {

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

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

HttpSession session = request.getSession(false); // Get session without creating a new one

if (session != null && session.getAttribute("user") != null) { String username = (String) session.getAttribute("user"); out.println("<h2>Welcome, " + username + "!</h2>"); out.println("<a href='LogoutServlet'>Logout</a>");

} else {

out.println("<h3>Session expired! Please login again.</h3>"); request.getRequestDispatcher("index.html").include(request,

response);

}

}

}

#### LogoutServlet.java (Session Invalidation)

java CopyEdit

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; import javax.servlet.http.HttpSession;

@WebServlet("/LogoutServlet")

public class LogoutServlet extends HttpServlet {

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

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

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

session.invalidate(); // Destroy session

}

out.println("<h3>You have been logged out successfully.</h3>"); request.getRequestDispatcher("index.html").include(request,

response);

}

}

#### web.xml (Servlet Configuration)

xml CopyEdit

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

<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>DashboardServlet</servlet-name>

<servlet-class>DashboardServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>DashboardServlet</servlet-name>

<url-pattern>/DashboardServlet</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>LogoutServlet</servlet-name>

<servlet-class>LogoutServlet</servlet-class>

</servlet>

<servlet-mapping>

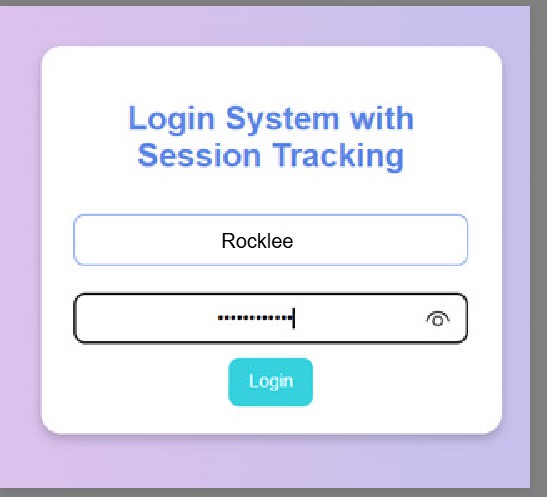
<servlet-name>LogoutServlet</servlet-name>

<url-pattern>/LogoutServlet</url-pattern>

</servlet-mapping>

</web-app>

##### OUTPUT:

****

**RESULT:** The Session Tracking using HttpSession were executed successfully.

**EXP 7 Servlet-based project is to store user preferences**

## AIM:

The goal of this Servlet-based project is to store user preferences (such as theme and language) using cookies and retrieve and display them on subsequent visits.

## ALGORITHM:

Step 1: Load Preferences Page **→ Browser checks for existing cookies.**

Step 2 Retrieve Preferences (GET Request) **→ Servlet reads cookies and applies saved settings.**

Step 3: User Selects Preferences **→ User picks theme and language.**

Step 4: Save Preferences (POST Request) **→ JavaScript sends data to Servlet.**

Step 5: Servlet Stores Preferences **→ Cookies are updated and sent back to the browser.** Step 6: Preferences Applied on Next Visit **→ Browser loads saved settings automatically. PROGRAM:**

### index.html (Frontend - Combined HTML, CSS & JavaScript)

html CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Preferences using Cookies</title>

<style>

body {

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

background: linear-gradient(to right, #ffafbd, #ffc3a0); text-align: center;

display: flex;

justify-content: center; align-items: center; height: 100vh;

margin: 0;

transition: background 0.5s ease-in-out;

}

.container {

background: white; padding: 20px; border-radius: 15px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

}

h2 {

}

width: 320px;

color: #ff6f61;

select, button { width: 90%; padding: 10px; margin: 10px 0;

border: 1px solid #ff6f61; border-radius: 8px;

text-align: center;

}

button {

background: #ff6f61; color: white; border: none; cursor: pointer; transition: 0.3s;

}

button:hover { background: #e6514a;

}

.pref-box {

margin-top: 20px; padding: 10px;

background: rgba(255, 255, 255, 0.8); color: black;

border-radius: 10px; font-weight: bold;

}

.hidden {

display: none;

}

.dark-mode {

background: linear-gradient(to right, #333333, #1e1e1e); color: white;

}

.dark-mode .container {

background: rgba(255, 255, 255, 0.1); backdrop-filter: blur(10px);

}

.dark-mode .pref-box {

background: rgba(255, 255, 255, 0.2); color: white;

}

</style>

</head>

<body>

<div class="container">

<h2>Set Your Preferences</h2>

<div id="form-section">

<label for="theme">Choose Theme:</label>

<select id="theme">

<option value="light">Light</option>

<option value="dark">Dark</option>

</select>

<label for="language">Choose Language:</label>

<select id="language">

<option value="English">English</option>

<option value="French">French</option>

</select>

<button onclick="savePreferences()">Save Preferences</button>

</div>

<div id="view-section" class="hidden">

<h2>Your Preferences</h2>

<div class="pref-box">

<p>🌟 Theme: <span id="saved-theme">-</span></p>

<p>🌍 Language: <span id="saved-language">-</span></p>

</div>

<button onclick="resetPreferences()">Change Preferences</button>

</div>

</div>

<script>

function savePreferences() {

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

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

fetch("PreferenceServlet", { method: "POST",

headers: { "Content-Type": "application/x-www-form-

urlencoded" },

body: `theme=${theme}&language=${language}`

}).then(() => updateView());

}

function updateView() { fetch("PreferenceServlet")

.then(response => response.json())

.then(data => {

document.getElementById("saved-theme").innerText =

data.theme; data.language;

=== "dark");

document.getElementById("saved-language").innerText = document.body.classList.toggle("dark-mode", data.theme

document.getElementById("form-

section").classList.add("hidden");

document.getElementById("view- section").classList.remove("hidden");

});

}

function resetPreferences() {

fetch("PreferenceServlet", { method: "DELETE" }).then(() => { document.getElementById("form-

section").classList.remove("hidden");

document.getElementById("view- section").classList.add("hidden");

});

}

window.onload = updateView;

</script>

</body>

</html>

**PreferenceServlet.java (Backend - Java Servlet)**

java CopyEdit

import java.io.IOException;

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

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest; import javax.servlet.http.HttpServletResponse; import org.json.JSONObject;

@WebServlet("/PreferenceServlet")

public class PreferenceServlet extends HttpServlet {

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

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

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

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

themeCookie.setMaxAge(60 \* 60 \* 24 \* 7); // 1 Week

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

response.addCookie(themeCookie); response.addCookie(languageCookie);

}

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

String theme = "light"; String language = "English";

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

for (Cookie cookie : cookies) {

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

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

}

}

JSONObject json = new JSONObject(); json.put("theme", theme); json.put("language", language);

response.setContentType("application/json"); response.getWriter().write(json.toString());

}

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

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

themeCookie.setMaxAge(0); languageCookie.setMaxAge(0);

response.addCookie(themeCookie); response.addCookie(languageCookie);

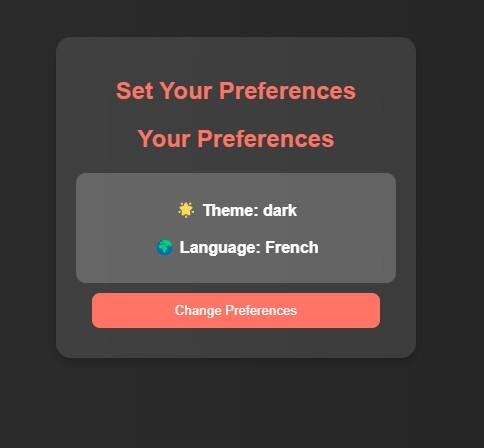
}

}

##### OUTPUT:

****



****

**RESULT:** The Servlet-based project to store user preferences using cookies were executed successfully.

**EXP 8 LIBRARY MANAGEMENT SYSTEM**

🎯 **AIM**

To develop a **Library Management System** using **Android Studio** that allows users to enter book details, validates the inputs on the frontend, and displays a cute success message upon submission. The app uses a beautiful UI and ensures input correctness without connecting to a backend.

# ALGORITHM

###### Start the Application.

* 1. Display the **book entry form** with fields:

Book Name, Author, ISBN, and Category.

* 1. Wait for the user to **enter the details**.

###### On Submit Button Click:

* + - Check if **all fields** are filled.
    - If any field is empty, show a **toast message**: “Please fill all fields”.
    - If all fields are valid:
      * Display a big **“Book Added Successfully”** message.
  1. End.

# ☎ CODE STRUCTURE

✅ MainActivity.kt

kotlin CopyEdit

package com.example.calci

import android.os.Bundle import android.widget.\*

import androidx.appcompat.app.AppCompatActivity class MainActivity : AppCompatActivity() {

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

val bookName = findViewById<EditText>(R.id.bookName) val author = findViewById<EditText>(R.id.author) val isbn = findViewById<EditText>(R.id.isbn)

val category = findViewById<EditText>(R.id.category)

val submitBtn = findViewById<Button>(R.id.submitBtn)

val message = findViewById<TextView>(R.id.successMessage)

submitBtn.setOnClickListener {

val bName = bookName.text.toString().trim() val auth = author.text.toString().trim() val isbnCode = isbn.text.toString().trim() val cat = category.text.toString().trim()

if (bName.isEmpty() || auth.isEmpty() || isbnCode.isEmpty() || cat.isEmpty()) {

Toast.makeText(this, "Please fill all fields", Toast.LENGTH\_SHORT).show()

} else {

message.text = "\uD83D\uDCDA Book Added Successfully!" message.visibility = TextView.VISIBLE

}

}

}

}

* activity\_main.xml

xml CopyEdit

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

<LinearLayout

xmlns:android=["http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) android:orientation="vertical"

android:padding="24dp" android:gravity="center" android:background="@color/pink\_bg" android:layout\_width="match\_parent" android:layout\_height="match\_parent">

<TextView

android:text="☎ Library Book Entry" android:textSize="26sp" android:textColor="@color/deep\_pink" android:layout\_marginBottom="24dp" android:textStyle="bold" android:layout\_gravity="center" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" />

<EditText

android:id="@+id/bookName" android:hint="Book Name" android:background="@drawable/input\_bg" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" />

<EditText

android:id="@+id/author" android:hint="Author Name" android:background="@drawable/input\_bg" android:layout\_marginTop="10dp" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" />

<EditText

android:id="@+id/isbn" android:hint="ISBN" android:background="@drawable/input\_bg" android:layout\_marginTop="10dp" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" />

<EditText

android:id="@+id/category" android:hint="Category" android:background="@drawable/input\_bg" android:layout\_marginTop="10dp" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" />

<Button

android:id="@+id/submitBtn" android:text="Submit" android:layout\_marginTop="16dp" android:backgroundTint="@color/deep\_pink" android:textColor="#fff" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" />

<TextView

android:id="@+id/successMessage" android:text="" android:textSize="22sp" android:gravity="center" android:textStyle="bold"

android:textColor="@color/success\_green" android:layout\_marginTop="18dp" android:visibility="gone" android:layout\_width="match\_parent" android:layout\_height="wrap\_content"/>

</LinearLayout>

🎨 **colors.xml**

xml CopyEdit

<resources>

<color name="pink\_bg">#FFF0F5</color>

<color name="deep\_pink">#FF1493</color>

<color name="success\_green">#228B22</color>

</resources>

🎨 **input\_bg.xml (drawable folder)**

xml CopyEdit

<shape xmlns:android=["http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) android:shape="rectangle">

<solid android:color="#FFFFFF"/>

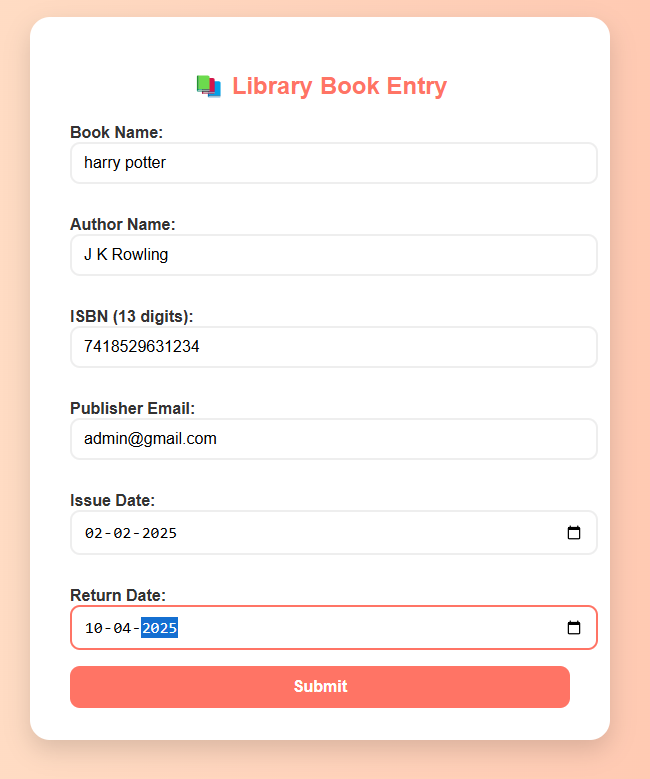
<corners android:radius="16dp"/>

<stroke android:width="2dp" android:color="#FFB6C1"/>

<padding android:left="12dp" android:right="12dp"/>

</shape>

**OUTPUT:**



****

✅ **RESULT**

When the user fills in all the fields and presses **Submit**, the screen displays:

###### 📚 Book Added Successfully!

With cute pink background, rounded input fields, toast messages for empty fields, and success UI.

**EXP 9 CALCULATOR APP**

🎯 **AIM**

To develop a **simple and cute Calculator App** in Android Studio using Kotlin, allowing the user to perform basic arithmetic operations (Addition, Subtraction, Multiplication, Division) with a mobile-friendly interface and clear input/output validation.

# ALGORITHM

* 1. Start the app.
  2. Display two input fields for numbers.
  3. Show buttons: ➕ ➖ ✖◻ ➗ and ◻ (clear).
  4. User enters two numbers and taps a button.
  5. App checks if both inputs are valid numbers:
     + If not: show a toast message “Enter valid numbers”.
     + If valid:
       - Perform the selected operation.
       - Show the result on the screen.
  6. Clear button resets everything.
  7. End.

☎ **CODE**

✅ MainActivity.kt

kotlin CopyEdit

package com.example.calci

import android.os.Bundle import android.widget.\*

import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() { private lateinit var num1: EditText private lateinit var num2: EditText private lateinit var result: TextView private lateinit var addBtn: Button private lateinit var subBtn: Button private lateinit var mulBtn: Button private lateinit var divBtn: Button private lateinit var clearBtn: Button

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

num1 = findViewById(R.id.number1) num2 = findViewById(R.id.number2) result = findViewById(R.id.result) addBtn = findViewById(R.id.add) subBtn = findViewById(R.id.subtract) mulBtn = findViewById(R.id.multiply) divBtn = findViewById(R.id.divide) clearBtn = findViewById(R.id.clear)

addBtn.setOnClickListener { calculate("+") } subBtn.setOnClickListener { calculate("-") } mulBtn.setOnClickListener { calculate("\*") } divBtn.setOnClickListener { calculate("/") } clearBtn.setOnClickListener {

num1.text.clear() num2.text.clear() result.text = ""

}

}

private fun calculate(op: String) { val n1Text = num1.text.toString() val n2Text = num2.text.toString()

if (n1Text.isEmpty() || n2Text.isEmpty()) { Toast.makeText(this, "Enter valid numbers",

Toast.LENGTH\_SHORT).show()

return

}

val n1 = n1Text.toDouble() val n2 = n2Text.toDouble() val res = when (op) {

"+" -> n1 + n2 "-" -> n1 - n2

"\*" -> n1 \* n2 "/" -> {

if (n2 == 0.0) {

Toast.makeText(this, "Cannot divide by zero", Toast.LENGTH\_SHORT).show()

return

}

n1 / n2

}

else -> 0.0

}

result.text = "Result: $res"

}

}

* **activity\_main.xml (Cute Styling UI)**

xml CopyEdit

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

<LinearLayout

xmlns:android=["http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:orientation="vertical" android:background="#FFF1F8"

android:padding="24dp" android:gravity="center">

<TextView

android:text="Cute Calculator 💖" android:textSize="28sp" android:textColor="#E91E63" android:layout\_marginBottom="16dp" android:textStyle="bold" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" />

<EditText

android:id="@+id/number1" android:hint="Enter Number 1" android:inputType="numberDecimal" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:backgroundTint="#E91E63" android:padding="10dp" android:layout\_marginBottom="12dp"/>

<EditText

android:id="@+id/number2" android:hint="Enter Number 2" android:inputType="numberDecimal" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:backgroundTint="#E91E63" android:padding="10dp" android:layout\_marginBottom="24dp"/>

<LinearLayout

android:orientation="horizontal" android:layout\_width="match\_parent" android:layout\_height="wrap\_content" android:gravity="center" android:layout\_marginBottom="16dp">

<Button

android:id="@+id/add" android:text="➕" android:layout\_width="0dp" android:layout\_weight="1" android:layout\_height="wrap\_content" android:backgroundTint="#F8BBD0" />

<Button

android:id="@+id/subtract" android:text="➖" android:layout\_width="0dp" android:layout\_weight="1" android:layout\_height="wrap\_content" android:backgroundTint="#F8BBD0"

android:layout\_marginStart="8dp"/>

<Button

android:id="@+id/multiply" android:text="✖◻" android:layout\_width="0dp" android:layout\_weight="1" android:layout\_height="wrap\_content" android:backgroundTint="#F8BBD0" android:layout\_marginStart="8dp"/>

<Button

android:id="@+id/divide" android:text="➗" android:layout\_width="0dp" android:layout\_weight="1" android:layout\_height="wrap\_content" android:backgroundTint="#F8BBD0" android:layout\_marginStart="8dp"/>

</LinearLayout>

<Button

android:id="@+id/clear" android:text="◻ Clear" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:backgroundTint="#CE93D8" android:layout\_marginBottom="16dp" />

<TextView

android:id="@+id/result" android:textSize="22sp" android:textStyle="bold" android:textColor="#880E4F" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" />

</LinearLayout>

**OUTPUT:**

****

✅ **RESULT**

Once you run the app:

* You can enter two numbers.
* Tap any operation: ➕ ➖ ✖◻ ➗
* Result appears below in bold.
* Clear button resets the input.
* If input is missing or invalid, you’ll see a toast message.

**EXP 10 FONT COLOUR CHANGER**

🎯 **AIM**

To develop an Android application that **changes the font and color of a TextView** and shows a **toast message** when the user clicks a button.

# ALGORITHM

* 1. Start the app.
  2. Show a TextView with default text.
  3. Display a Button: Change Style.
  4. When the button is clicked:
     + Change the font style (bold/italic/custom).
     + Change the text color (to a vibrant color).
     + Show a Toast: "Style Changed!"
  5. End.

☎ **CODE**

✅ MainActivity.kt

kotlin CopyEdit

package com.example.textstyler

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

import android.widget.\*

import androidx.appcompat.app.AppCompatActivity class MainActivity : AppCompatActivity() {

lateinit var myText: TextView lateinit var styleButton: Button

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

myText = findViewById(R.id.myText) styleButton = findViewById(R.id.styleButton)

styleButton.setOnClickListener {

pink

myText.setTextColor(Color.parseColor("#E91E63")) // Vibrant

myText.setTypeface(null, Typeface.BOLD\_ITALIC) myText.textSize = 24F

Toast.makeText(this, "Style Changed!",

Toast.LENGTH\_SHORT).show()

}

}

}

🎨 activity\_main.xml

xml CopyEdit

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

<LinearLayout

xmlns:android=["http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) android:orientation="vertical"

android:gravity="center" android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:background="#FFF8E1" android:padding="24dp">

<TextView

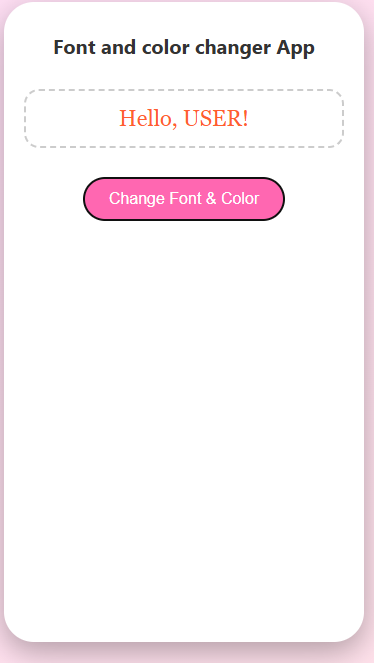
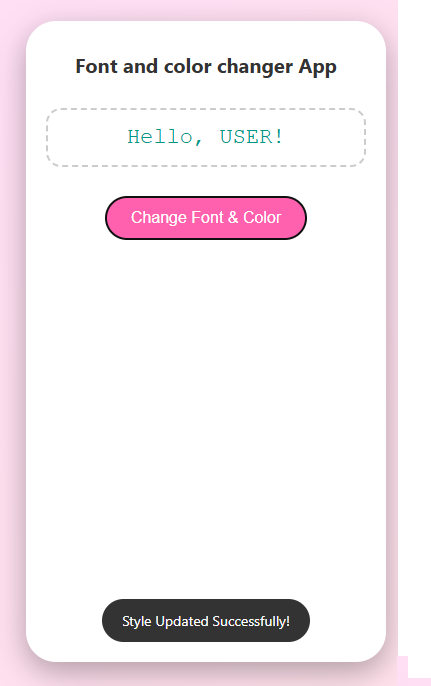
android:id="@+id/myText" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="Welcome to My App!" android:textSize="20sp" android:textColor="#333" android:padding="16dp" />

<Button

android:id="@+id/styleButton" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="Change Style" android:textColor="#fff" android:backgroundTint="#F48FB1" android:layout\_marginTop="20dp" android:padding="10dp"/>

</LinearLayout>

## OUTPUT:

****

✅ **RESULT**

* When the app runs, a TextView displays the message.
* Pressing the **Change Style** button:
  + Changes the **text color to pink**, makes it **bold italic**, and increases the size.