A blue and white sign with black text

Description automatically generated

**Lab File**

**Web programming with Python and**

**JavaScript Lab**

**(SEC 035)**

**SUBMITTED BY:**

**VAIBHAV VAID**

**SUBMITTED TO:**

**DR. Tanvi Chawla**

**ROLL NUMBER: 2301010289**

**COURSE: BTECH CSE CORE**

**SECTION: E**

**School of Engineering & Technology**

**K. R. MANGALAM UNIVERSITY**

**Sohna, Haryana 122103, Indi**

**Experiment 1:** Creating a restaurant website with HTML and CSS.

**HTML CODE:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Document</title>

</head>

<body>

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

<h1> OM SWEETS</h1>

<p>Your favorite place for delicious food</p>

</header>

<div class="slide-right">

<h2>Welcome to the one of the famous resturant of NCR</h2>

</div>

<div class="container">

<h2><u>Order Your Choice</u> </h2>

<div class="menu-section">

<table>

<tr>

<div class="menu-item">

<img src="burger.jpg" alt="Dish 1" height="1000" width="300">

<a href="https://www.bk.com/"><h3>Burger</h3></a>

<ol>

<li>Veg Burger</li>

<li>Cheese Burger</li>

<li>Chicken Burger</li>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**A screenshot of a menu

AI-generated content may be incorrect.

**Experiment 2:-i) Simple Basic JavaScript Programs .**

**A white screen with black text

AI-generated content may be incorrect.program 1:**

**A white background with black text

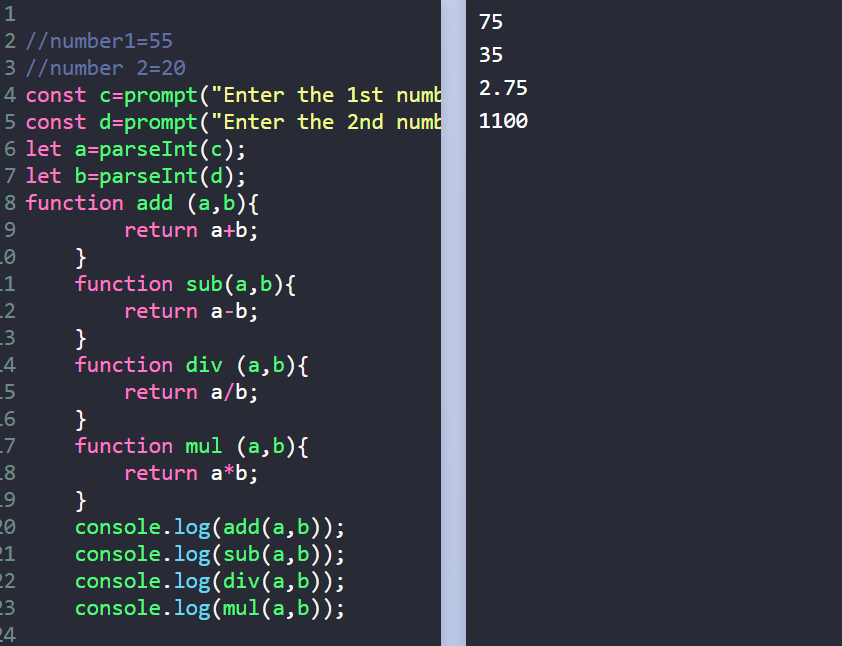
AI-generated content may be incorrect.program 2:**

**program** A screenshot of a computer

AI-generated content may be incorrect.**3:**

**A computer screen with text

AI-generated content may be incorrect.Program 4:**

**Program 5:**

**Experiment2:- ii) webpages using javascript .**

1. **Implement a Javascript function to change the background colour of a button on hover**
2. **Create a JS countdown timer that triggers and event when the timer reaches zero.**
3. **JS program to Reverse a string and check if the string is palindrome or not .**
4. **JS program to find number of occurrence of a character in the string.**

**Codes:**

**A)** <!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Hover Background Change with JavaScript</title>

    <style>

        /\* Button style \*/

        .hoverable-button {

            width: 200px;

            height: 200px;

            background-color: lightblue;

            align-content: center;

            color: white;

            border: none;

            cursor: pointer;

            font-size: 16px;

            transition: background-color 0.3s ease; /\* Smooth transition \*/

        }

    </style>

</head>

<body>

    <button class="hoverable-button" id="hoverButton">See the change in me : Hover over me!</button>

    <script>

        // Get the button element by ID

        const button = document.getElementById('hoverButton');

        // Function to change the background color on mouseenter

        function changeBackgroundColor() {

            button.style.backgroundColor = 'orange'; // Change color to orange

        }

        // Function to reset the background color on mouseleave

        function resetBackgroundColor() {

            button.style.backgroundColor = 'lightblue'; // Reset to original color

        }

        // Add event listeners for hover effect

        button.addEventListener('mouseenter', changeBackgroundColor);

        button.addEventListener('mouseleave', resetBackgroundColor);

    </script>

</body>

</html>

**Output:**

**A yellow square with white text

AI-generated content may be incorrect.A blue square with white text

AI-generated content may be incorrect.**

**c)** <!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

<title>Reverse String and Palindrome Check</title>

</head>

<body>

    <h2>Reverse a String and Check if Palindrome</h2>

    <input type="text" id="inputString" placeholder="Enter a string">

    <button onclick="reverseAndCheckPalindrome()">Check</button>

    <p id="reversed"></p>

    <p id="palindromeResult"></p>

    <script>

        // Function to reverse a string

        function reverseString(str) {

            return str.split('').reverse().join('');

        }        function isPalindrome(str) {

            const reversedStr = reverseString(str);

            return str === reversedStr;

        }

        function reverseAndCheckPalindrome() {

            const inputString = document.getElementById('inputString').value;

            const reversedString = reverseString(inputString);

            document.getElementById('reversed').textContent = `Reversed string: ${reversedString}`;

            const palindromeCheck = isPalindrome(inputString);

            if (palindromeCheck) {

                document.getElementById('palindromeResult').textContent = `"${inputString}" is a palindrome!`;

            } else {

                document.getElementById('palindromeResult').textContent = `"${inputString}" is not a palindrome.`;

            }

        }

    </script>

</body>

</html>

**A screenshot of a computer screen

AI-generated content may be incorrect.Output:**

**d)** <!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Count Character Occurrences</title>

</head>

<body>

    <h2>Find the Number of Occurrences of a Character in a String</h2>

    <input type="text" id="inputString" placeholder="Enter a string">

    <input type="text" id="charToFind" placeholder="Enter character to find">

    <button onclick="countCharacterOccurrences()">Count Occurrences</button>

    <h3><p id="result"></p>

    </h3>

    <script>

        // Function to count occurrences of a character in a string

        function countCharacterOccurrences() {

            const str = document.getElementById('inputString').value;

            const char = document.getElementById('charToFind').value;

            if (char.length !== 1) {

                document.getElementById('result').textContent = "Please enter a single character to find.";

                return;

            }

            // Count the occurrences of the character

            let count = 0;

            for (let i = 0; i < str.length; i++) {

                if (str[i] === char) {

                    count++;

                }}

document.getElementById('result').textContent = `The character '${char}' appears ${count} time(s) in the string.`;

        }

    </script>

</body>

</html>

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 3:**

**A)Create a registration / login page with the help of using html , CSS and Javascript.**

**Code:**

<!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>

  <style>

    body {

      font-family: Arial, sans-serif;

      display: flex;

      justify-content: center;

      align-items: center;

      height: 100vh;

      background-color:rgba(244, 244, 244, 0.51);

      margin: 0;

    }

    .login-container {

      background-color: rgb(80, 77, 77);

      padding: 50px;

      border-radius: 12px;

      box-shadow: 0 4px 8px rgba(18, 17, 17, 0.1);

      width: 300px;

      text-align: center;

    }

    h2 {

      margin-bottom: 30px;

**A login screen with green and black text

AI-generated content may be incorrect.*Output:***

**B)Create a To- Do List page with the help of using html , CSS and Javascript.**

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>To-Do List</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            display: flex;

            justify-content: space-between;

            padding: 20px;

            background-color: #f4f4f9;

        }

        .container {

            width: 45%;

        }

        ul {

            list-style-type: none;

            padding: 0;

        }

        li {

            padding: 8px;

            background-color: #e3e3e3;

            margin: 5px 0;

            cursor: pointer;

<div class="container">

    <h2>To-Do List</h2>

    <ul id="todoList">

        <!-- To-Do tasks will appear here -->

    </ul>

    <input type="text" id="taskInput" placeholder="Add a task" />

    <button onclick="addTask()">Add Task</button>

</div>

<div class="container">

    <h2>Completed Tasks</h2>

    <ul id="completedList">

        <!-- Completed tasks will appear here -->

    </ul>

</div>

<script>

    const todoList = document.getElementById("todoList");

    const completedList = document.getElementById("completedList");

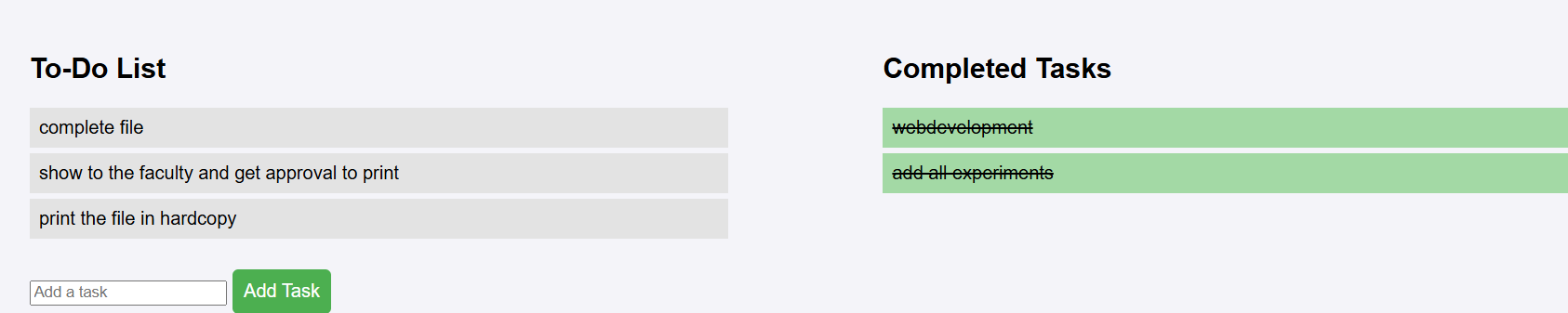
    const taskInput = document.getElementById("taskInput");

</script>

</body>

</html>

**Output:**

****

**C)Create a with the help of using html , CSS and Javascript.**

**Code:**

<div class="calculator">

<input type="text" class="display" id="display" disabled>

<div class="buttons">

<button class="clear" onclick="clearDisplay()">C</button>

<button class="operator" onclick="appendValue('/')">/</button>

<button class="operator" onclick="appendValue('\*')">\*</button>

<button class="operator" onclick="appendValue('-')">-</button>

<button class="number" onclick="appendValue('7')">7</button>

<button class="number" onclick="appendValue('8')">8</button>

<button class="number" onclick="appendValue('9')">9</button>

<button class="operator" onclick="appendValue('+')">+</button>

<button class="number" onclick="appendValue('4')">4</button>

<button class="number" onclick="appendValue('5')">5</button>

<button class="number" onclick="appendValue('6')">6</button>

<button class="equal" onclick="calculate()">=</button>

<button class="number" onclick="appendValue('1')">1</button>

<button class="number" onclick="appendValue('2')">2</button>

<button class="number" onclick="appendValue('3')">3</button>

<button class="number" onclick="appendValue('0')">0</button>

<button class="number" onclick="appendValue('.')">.</button>

</div>

</div>

<script>

function appendValue(value) {

document.getElementById('display').value += value;

}

function clearDisplay() {

document.getElementById('display').value = '';

}

function calculate() {

try {

document.getElementById('display').value = eval(document.getElementById('display').value);

} catch {

document.getElementById('display').value = 'Error';

}

}

</script>

</body>

</html>

**A black calculator with orange and green buttons

AI-generated content may be incorrect.Output:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 4:Create a portfolio using html, Css and javaScript.**

**Code:**

y>

<header>My Portfolio</header>

<div class="container">

<img src="https://via.placeholder.com/180" alt="Profile Image" class="profile-img">

<div class="about">

<h2>About Me</h2>

<p>Hello! I'm a web developer passionate about creating beautiful and functional websites.</p>

</div>

<div class="projects">

<h2>Projects</h2>

<p>Here are some of my recent works:</p>

<ul>

<li>Project 1 - Web Design</li>

<li>Project 2 - JavaScript App</li>

<li>Project 3 - Responsive Layout</li>

</ul>

</div>

<div class="contact">

<h2>Contact Me</h2>

<p>Email: myemail@example.com</p>

<a href="#" class="btn">Get in Touch</a>

</div>

</div>

</body>

</html>

**Output: A screenshot of a pink and white web page

AI-generated content may be incorrect.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 5: i) Create a Rock, Paper and Scissor game webpage using html, CSS and JavaScript.**

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Rock, Paper, Scissors Game</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css" rel="stylesheet">

    <style>

        /\* General Body \*/

        body {

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

            margin: 0;

            padding: 0;

            background-color: #f4f4f9;

            color: #333;

            text-align: center;

       }

    <!-- Header -->

    <header>

        Rock, Paper, Scissors Game

    </header>

    <!-- Game Container -->

    <div class="game-container" id="game-container">

        <h2>Let's Play!</h2>

        <!-- Instructions Section -->

        <div class="instructions">

            <h3>Rules:</h3>

            <ul>

                <li>Rock beats Scissors</li>

                <li>Scissors beats Paper</li>

                <li>Paper beats Rock</li>

            </ul>

        </div>

        <!-- Game Choices -->

        <div class="choices">

            <div class="choice" onclick="playGame('rock')">🪨 Rock</div>

            <div class="choice" onclick="playGame('paper')">📄 Paper</div>

            <div class="choice" onclick="playGame('scissors')">✂️ Scissors</div>

        </div>

        <!-- Result -->

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

            Choose your move!

</body>

</html>

**Output:**

A screenshot of a game

AI-generated content may be incorrect.

**ii) Create a movie and books library webpage using html, Css and javaScript.**

**Code:**

 <!-- Navbar -->

    <nav>

        <div class="logo">Library</div>

        <div>

            <a href="#movies">Movies</a>

            <a href="#books">Books</a>

            <a href="#contact">Contact</a>

        </div>

    </nav>

  <!-- Main content area -->

    <div class="container">

        <!-- Movies Section -->

        <div class="movie-section" id="movies">

            <h2>Movies</h2>

            <div class="movie-card">

                <img src="MOVIE1.jpg" alt="Movie 1" width="200" height="150">

                <h3>bECAME BY BRAGGING</h3>

                <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>

            </div>

            <div class="movie-card">

                <img src="MOVIE2.jpg" alt="Movie 2">

                <h3>PATHAAN</h3>

                <p>Curabitur pretium tincidunt lacus.</p>

            </div>

            <div class="movie-card">

                <img src="MOVIE3.jpg" alt="Movie 3"

            <div class="book-card">

                <img src="BOOK1.jpg" alt="Book 1">

                <h3>TOURS</h3>

                <p>THE AUTHORS GUIDE</p>

            </div>

            <div class="book-card">

                <img src="BOOK2.jpg" alt="Book 2">

                <h3>NOVEL </h3>

                <p>THE SOUND OF HOURS.</p>

            </div>

            <div class="book-card">

                <img src="BOOK3.jpg" alt="Book 3">

                <h3>STORY BOOK </h3>

                <p>THE SECRECT HOURS .</p>

            </div>

        </div>

    </div>

        <p>&copy; 2025 Movie & Book Library</p>

        <div class="social-icons">

            <i class="fab fa-facebook"></i>

            <i class="fab fa-twitter"></i>

            <i class="fab fa-instagram"></i>

        </div>

    </footer>

</body></html>

**Output:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**iii) Create a weatherportal webpage using html, Css and javaScript.**

**Code:**

    <!-- Navbar -->

    <nav>

        <div class="logo">Weather Portal</div>

        <div>

            <a href="#home">Home</a>

            <a href="#about">About</a>

            <a href="#contact">Contact</a>

        </div>

    </nav>

    <!-- Main Content -->

    <div class="main-container">

        <h2>Get the Latest Weather Information</h2>

        <!-- City Selection and Search Section -->

        <div class="search-container">

            <select id="citySelect">

                <option value="New Delhi">New Delhi</option>

                <option value="Mumbai">Mumbai</option>

                <option value="Kolkata">Kolkata</option>

                <option value="Chennai">Chennai</option>

                <option value="Bangalore">Bangalore</option>

                <option value="Hyderabad">Hyderabad</option>

            </select>

            <button onclick="getWeather()">Get Weather</button>

        </div>

        <!-- Weather Information -->

        <div class="weather-info" id="weather-info">

            <p>Loading weather data...</p>

        </div>

        <!-- Weather Season Information -->

        <div class="season-info">

            <p id="season-info"></p>

        </div>

        <!-- 5 Day Weather Forecast -->

        <div id="forecast" class="forecast">

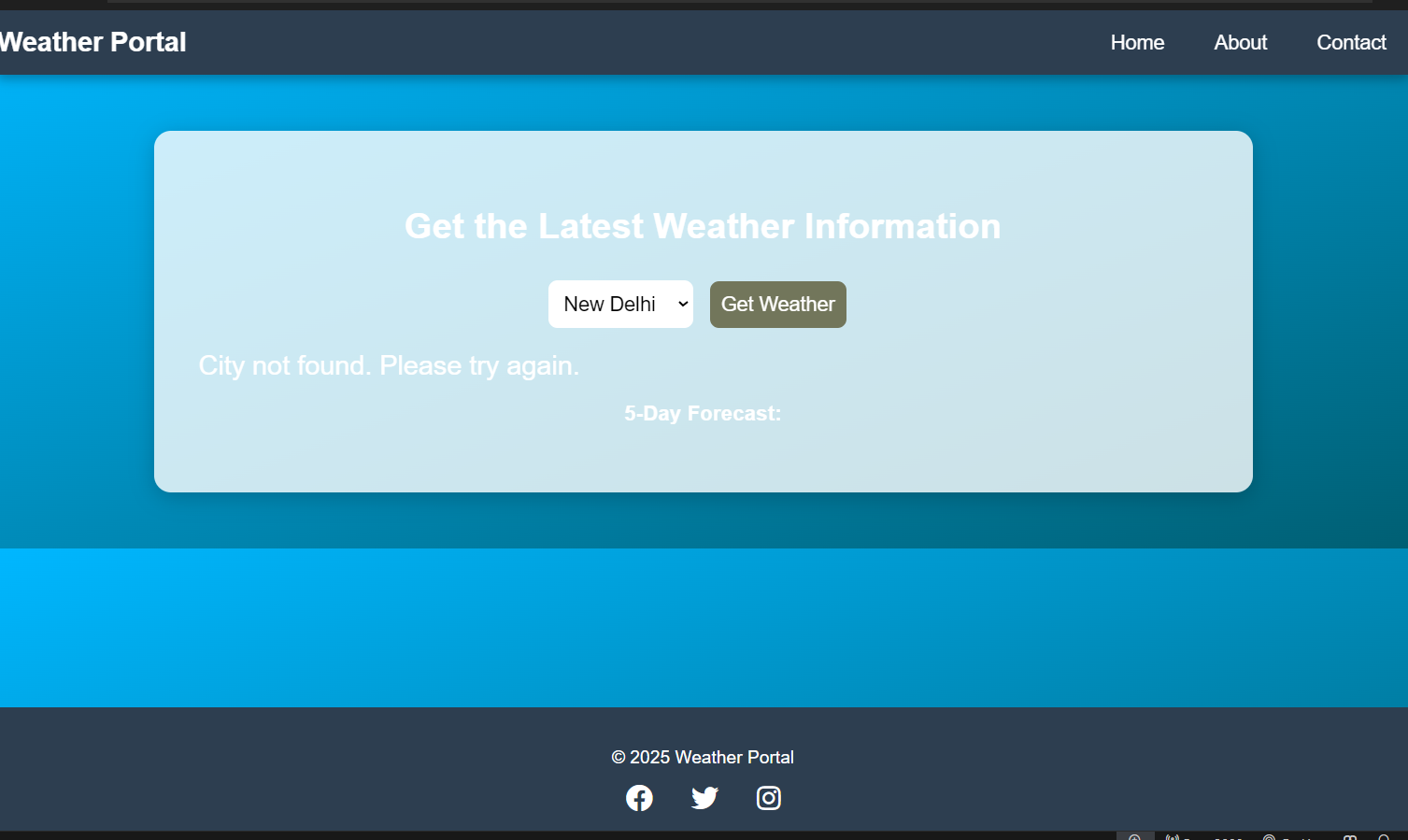
            <h3>5-Day Forecast:</h3>

            <div id="forecast-details"></div>

        </div>

    </div>

**Output:**

****

**Experiment 6: Create a webpage using if/for and other loops.**

**Code:**

 // Function to add a task to the list

        function addTask() {

            const taskInput = document.getElementById('taskInput');

            const timerInput = document.getElementById('timerInput');

            const taskText = taskInput.value.trim();

            const timerValue = parseInt(timerInput.value);

            if (taskText !== "" && !isNaN(timerValue) && timerValue > 0) {

                // Create a task with a timer

                const task = {

                    text: taskText,

                    timer: timerValue, // Expiration timer for the task

                    id: tasks.length // Unique task ID

                };

                tasks.push(task); // Add task to the array

                taskInput.value = ''; // Clear the task input field

                timerInput.value = ''; // Clear the timer input field

                displayTasks(); // Refresh the task list

                startTimer(task.id, timerValue); // Start the countdown for the task

            } else {

                alert("Please enter a valid task and timer.");

            }

        }

        // Function to display tasks

        function displayTasks() {

            const taskList = document.getElementById('taskList');

            const message = document.getElementById('message');

            // Clear the current list before adding new items

            taskList.innerHTML = "";

            // Check if the task list is empty

            if (tasks.length === 0) {

                message.textContent = "No tasks to display!";

            } else {

                message.textContent = ""; // Clear message if tasks exist

            }

                    timeLeft--;

                    timerElement.textContent = `${timeLeft} seconds left`;

                } else {

                    clearInterval(timerInterval);

                    timerElement.textContent = "Expired!";

                    timerElement.style.color = "gray";

                }

            }, 1000);

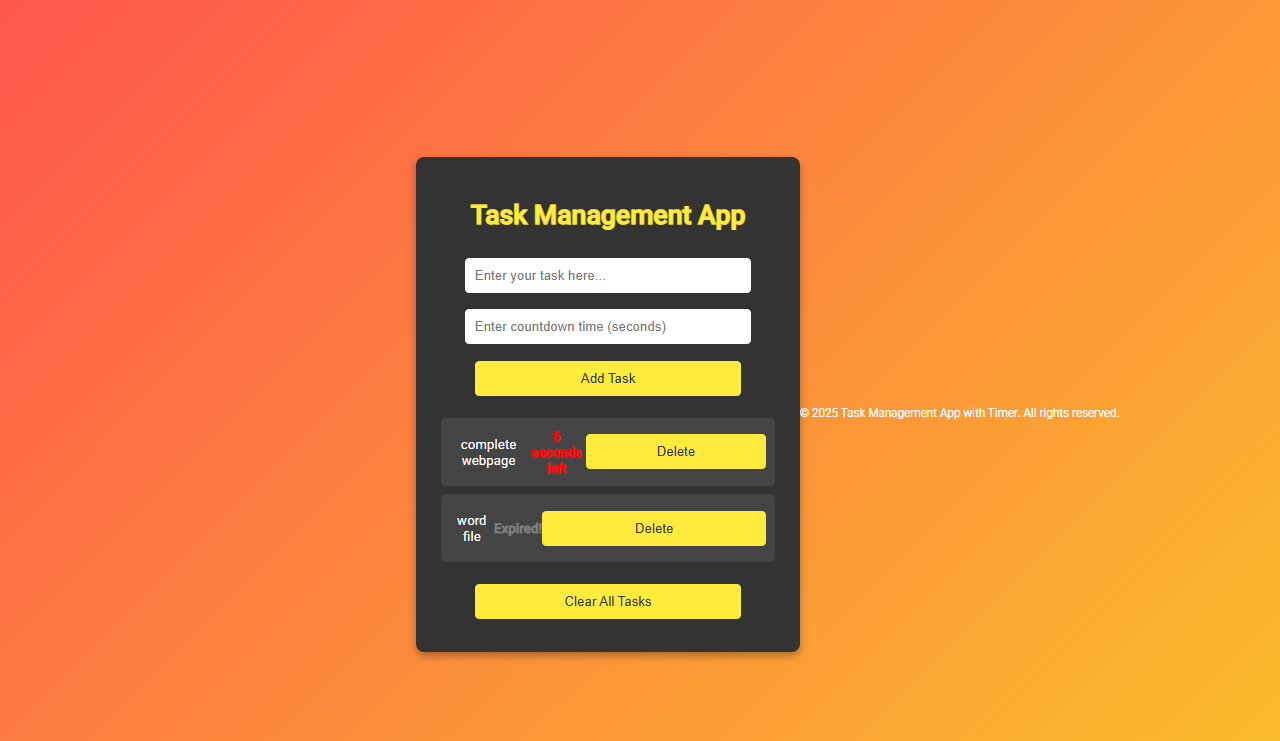
        }

    </script>

</body>

</html>

**Output:**



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 7: Create a javaScript Program to show the implementation of arrays,if ,loops.**

**Code :**

// Function to generate the star pattern

function generatePattern() {

// Loop to create rows

for (let i = 1; i <= 5; i++) {

let row = "";

// Loop to add stars in each row

for (let j = 1; j <= i; j++) {

row += "\* "; // Add a star to the row

}

console.log(row); // Print the row to the console

}}

generatePattern();

**A black screen with yellow text

AI-generated content may be incorrect.**

// Function to generate the number pattern

function generatePattern() {

// Loop to create rows

for (let i = 1; i <= 4; i++) {

let row = "";

// Loop to add numbers in each row

for (let j = 1; j <= i; j++) {

row += j + " "; // Add the number to the row

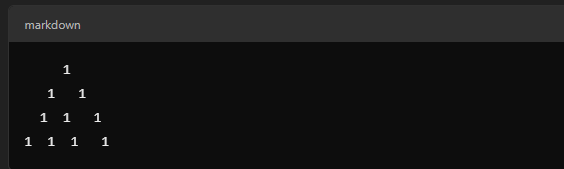
}

console.log(row); // Print the row to the console

}}

// Call the function to generate the pattern

generatePattern();

//

//Function to generate the pattern

function generatePattern() {

// Loop to create rows

for (let i = 1; i <= 4; i++) {

let row = "";

// Add spaces before the stars for each row

for (let j = 1; j <= 4 - i; j++) {

row += " "; // Add leading spaces

}

// Add stars in the row

for (let k = 1; k <= i; k++) {

row += "\* "; // Add stars with spaces in between

}

console.log(row); // Print the row to the console

}}

A black screen with yellow text

AI-generated content may be incorrect.generatePattern();

function generatePattern() {

// Loop to create rows

for (let i = 1; i <= 4; i++) {

let row = "";

// Add spaces before the 1s for each row

for (let j = 1; j <= 4 - i; j++) {

row += " "; // Add leading spaces

}

// Add 1s in the row

for (let k = 1; k <= i; k++) {

row += "1 "; // Add 1s with spaces in between

}

console.log(row); // Print the row to the console

}}

A screen shot of a computer

AI-generated content may be incorrect.generatePattern();

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 8:** **Create an ecommerce website with the help of html,css and javascript .**

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>My Ecommerce Store</title>

    <header>My Ecommerce Store</header>

    <div class="container">

        <div class="product">

            <img src="Hawkins cooker.jpg" alt="cookware">

            <h3>Hawkins cooker 5ltr</h3>

            <p>Rs. 2150</p>

            <a href="#" class="btn">Add to Cart</a>

        </div>

        <div class="product">

            <img src="frypan prestige.jpg" alt="cookware" height="200">

            <h3>Frypan</h3>

            <p>Rs.1450</p>

            <a href="#" class="btn">Add to Cart</a>

        </div>

        <div class="product">

            <img src="mixer grinder.jpg" alt="Kitchenware">

            <h3>Mixer Grinder</h3>

            <p>Rs. 27560</p>

            <a href="#" class="btn">Add to Cart</a>

        </div>

**Output:** A screenshot of a phone

AI-generated content may be incorrect.**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 9:Create a webpage showing the “Dom Implentation”**

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>DOM Example</title>

    <style>

        #message { font-size: 20px; color: blue; }

    </style>

</head>

<body>

    <p id="message">Hello, World!</p>

    <button onclick="changeText()">Click Me</button>

    <script>

        function changeText() {

            let msg = document.getElementById("message");

            msg.innerHTML = "Hello, DOM!";

            msg.style.color = "red";

        }

    </script>

</body>

</html>

**A white background with red text

AI-generated content may be incorrect.A close up of a message

AI-generated content may be incorrect.Output:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 10: A)Document Object Model using Javascript.**

**A screenshot of a computer

AI-generated content may be incorrect.1)**

**A white rectangular object with a black border

AI-generated content may be incorrect.2)**

**A white background with colorful text

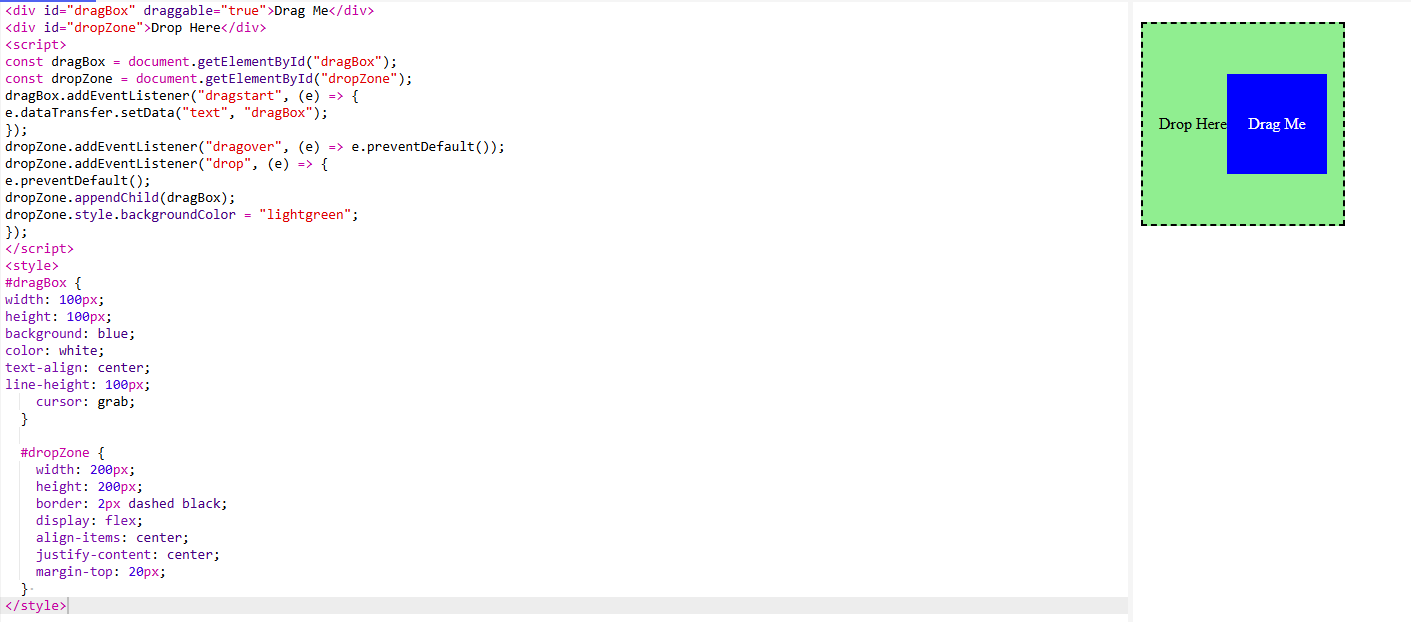
AI-generated content may be incorrect.3)**

**A white rectangular object with pink and purple text

AI-generated content may be incorrect.4)**

**A screen shot of a computer code

AI-generated content may be incorrect.5)**

**6)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**B)Project based on “Custom Content Menu ” Document Object Model using Javascript.**

**Code :**

<div id="content">

Right-click anywhere on this page to see the custom context menu.

</div>

<p>

This is a custom context menu demo. Right-click anywhere on this page to see it in action. You can customize the content and actions within the context menu.

</p>

<p>

The context menu is typically used to provide additional options when the user interacts with elements of the page. In this case, you can see it showing up dynamically based on the mouse position. Give it a try!

</p>

<ul id="customMenu" class="custom-menu">

<!-- Menu items will be inserted here dynamically -->

</ul>

<script>

// script.js

// Get references to the custom menu and the content area

const customMenu = document.getElementById('customMenu');

const content = document.getElementById('content');

// Menu items (can be dynamically changed)

const menuItems = [

{ name: "Option 1", action: () => alert('you choosed Option 1 ') },

{ name: "Option 2", action: () => alert('you choosed Option 2 ') },

{ name: "Option 3", action: () => alert('you choosed Option 3 ') },

];

// Function to populate the custom menu

function populateMenu() {

customMenu.innerHTML = ''; // Clear any previous items

menuItems.forEach(item => {

const li = document.createElement('li');

li.classList.add('menu-item');

li.textContent = item.name;

li.addEventListener('click', item.action);

customMenu.appendChild(li);

});

}  
  
**Output:**

A red and white rectangle with black text

AI-generated content may be incorrect.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 11: Basic Python Codes :**

# print("Hello world!")

**# 2 Variables and data types**

name="Alice"

age=25

height=5.6

is\_student =True

A screenshot of a computer

AI-generated content may be incorrect.print(name,age,height,is\_student)

**# 3 taking user input**

name=input("Enter your name: ")

print ("hello, "+ name+ "!")

**# 4 Conditional statements**

num=int(input("Enter a number: "))

if num>0:

    print("positive number")

elif num<0:

    print("negative number")

else:

    print("zero")

**# 5  loops**

for i in range(5):

    print(i)

count=0

while count<5:

    print(count)

    count+=1

---------------------------------------------------------------------------------------------------------------------------

**#6 Functions**

def greet(name):

    return "hello, "+ name +"!"

print (greet("vaibhav"))

**# 7 lists**

fruits =["apple","banana","cherry "]

fruits.append("orange")

print(fruits)

**# 8 Dictionaries**

person={"name": "vaibhav", "age": 25, "city": "New York"}

A screen shot of a computer screen

AI-generated content may be incorrect.print(person["name"])

**# 9 classes and objects**

class Person:

    def \_\_init\_\_(self,name,age):

        self.name =name

        self.age =age

    def greet(self):

        return f"hello, my name is {self.name} and i am {self.age} years old."

p = Person("vaibhav",21)

print (p.greet())

---------------------------------------------------------------------------------------------------------------------------

**# 10 calculator**

def calculator():

    num1 = float(input("Enter first number: "))

    operator = input("Enter operator (+, -, \*, /): ")

    num2 = float(input("Enter second number: "))

    if operator == '+':

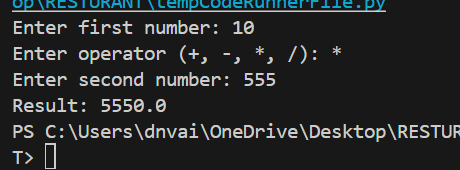
        print("Result:", num1 + num2)

    elif operator == '-':

        print("Result:", num1 - num2)

    elif operator == '\*':

        print("Result:", num1 \* num2)

    elif operator == '/':

        if num2 != 0:

            print("Result:", num1 / num2)

        else:

            print("Cannot divide by zero!")

    else:

        print("Invalid operator!")

calculator()

**# 11 to do list**

def add\_task(task):

    tasks.append(task)

def remove\_task(task):

A screen shot of a computer

AI-generated content may be incorrect.    if task in tasks:

        tasks.remove(task)

def show\_tasks():

    print("To-Do List:")

    for i, task in enumerate(tasks, start=1):

        print(f"{i

                 }. {task}")

while True:

    choice = input("Choose: add/remove/show/quit: ").lower()

    if choice == "add":

        task = input("Enter task: ")

        add\_task(task)

    elif choice == "remove":

        task = input("Enter task to remove: ")

        remove\_task(task)

    elif choice == "show":

        show\_tasks()

    elif choice == "quit":

        break

    else:

        print("Invalid choice!")

**# 12 to print area of a circle ,square, and rectangle**

import math

# Function to calculate area of a circle

def circle\_area(radius):

    return math.pi \* radius \*\* 2

# Function to calculate area of a square

def square\_area(side):

    return side \*\* 2

A computer screen shot of a black screen

AI-generated content may be incorrect.# Function to calculate area of a rectangle

def rectangle\_area(length, width):

    return length \* width

# Taking user inputs

radius = float(input("Enter the radius of the circle: "))

side = float(input("Enter the side of the square: "))

length = float(input("Enter the length of the rectangle: "))

width = float(input("Enter the width of the rectangle: "))

# Printing the areas

print(f"Area of the circle: {circle\_area(radius):.2f}")

print(f"Area of the square: {square\_area(side):.2f}")

print(f"Area of the rectangle: {rectangle\_area(length, width):.2f}")

**#13 WAP to print fibonacci**

#Function to print Fibonacci series

A computer screen shot of a black background

AI-generated content may be incorrect.def fibonacci(n):

    a, b = 0, 1

    for \_ in range(n):

        print(a, end=" ")

        a, b = b, a + b

# Taking user input

n = int(input("Enter the number of terms: "))

# Printing Fibonacci series

print("Fibonacci Series:")

fibonacci(n)

**# 14 wap to reverse a string and print palindrome or not**

**# Function to reverse a string**

def reverse\_string(s):

    return s[::-1]A screen shot of a computer

AI-generated content may be incorrect.

# Function to check palindrome

def is\_palindrome(s):

    return s == s[::-1]

string = input("Enter a string: ")

# Reversing the string

reversed\_string = reverse\_string(string)

# Checking if palindrome

if is\_palindrome(string):

    print(f"The string '{string}' is a palindrome.")

else:

    print(f"The string '{string}' is NOT a palindrome.")

# Printing reversed string

print(f"Reversed string: {reversed\_string}")

A screenshot of a computer program

AI-generated content may be incorrect.**# 15 Wap to print patterns**

**# a Taking user input for number of rows**

n = int(input("Enter the number of rows: "))

# Printing the pattern

for i in range(1, n + 1):

    for j in range(1, i + 1):

        print(j, end="")

    print()  # Move to the next line

# b Taking user input for number of rows

n = int(input("Enter the number of rows: "))

# Printing the pattern

for i in range(1, n + 1):

    for j in range(1, i + 1):

        print("\*", end="")

    print()  # Move to the next line

**Experiment 12: FLASK Programs**

**1. Basic "Hello, World!" Flask App**

A simple Flask app to display "Hello, World!" on the homepage.

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route('/')

def hello\_world():

return "Hello, World! Welcome to Flask."

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**How to run:  
Save as app.py and run using:**

**python app.py**

**Visit http://127.0.0.1:5000/ in your browser.**

**2. Flask App with Dynamic Routes**

This example shows how to create dynamic routes that take input from the URL.

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route('/user/<username>')

def show\_user\_profile(username):

return f"Hello, {username}! Welcome to your profile."

@app.route('/square/<int:number>')

def square\_number(number):

return f"The square of {number} is {number\*\*2}"

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**URL Examples:**

* **http://127.0.0.1:5000/user/John → *Hello, John!***
* **http://127.0.0.1:5000/square/4 → *The square of 4 is 16.***

**3. Flask App with HTML Templates (Jinja2)**

This program shows how to use **Jinja2 templates** to render dynamic HTML.

**Folder Structure:**

project/

│

├── app.py # Flask app file

└── templates/

└── home.html # HTML template

**HTML Template (templates/home.html):**

<!DOCTYPE html>

<html lang="en">

<head>

<title>{{ title }}</title>

</head>

<body>

<h1>Welcome to {{ title }}!</h1>

<p>This is a dynamic page rendered using Flask and Jinja2.</p>

</body>

</html>

**Python Code (app.py):**

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def home():

return render\_template('home.html', title="Flask Template Example")

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**4. Flask App with Forms**

This app accepts user input through an HTML form and displays the result.

**Folder Structure:**

project/

│

├── app.py # Flask app

└── templates/

└── form.html # HTML template for the form

**HTML Form (templates/form.html):**

<!DOCTYPE html>

<html lang="en">

<head>

<title>Flask Form Example</title>

</head>

<body>

<h2>Enter your name:</h2>

<form action="/greet" method="post">

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

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

</form>

</body>

</html>

**Python Code (app.py):**

from flask import Flask, request, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def form():

return render\_template('form.html')

@app.route('/greet', methods=['POST'])

def greet():

name = request.form['name']

return f"Hello, {name}! Welcome to Flask."

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**Experiment 13: Flask programs that render HTML pages using different techniques.**

**1. Basic HTML Rendering**

from flask import Flask, render\_template\_string

app = Flask(\_\_name\_\_)

@app.route('/')

def home():

html = """

<html>

<head><title>Home</title></head>

<body>

<h1>Welcome to Flask</h1>

<p>This is a basic HTML page.</p>

</body

</html> **Output:-**

A black and white image of a person's face

AI-generated content may be incorrect. """

return render\_template\_string(html)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

✅ **Explanation:**

* Uses render\_template\_string to render an HTML page directly.

**2. Rendering HTML from a Template File**

**Project Structure:**

bash

CopyEdit

/flask\_app

│── app.py

│── templates/

│ ├── index.html

**app.py (Flask Code)**

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def home():

return render\_template("index.html")

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**templates/index.html**

html

CopyEdit

<!DOCTYPE html>

<html>

<head>

<title>Flask Template</title>

</head>

<body>

<h1>Welcome to Flask!</h1>

<p>This is a page rendered from an HTML template.</p>

</body>

</html>

✅ **Explanation:**

* Uses render\_template() to serve an HTML file from the templates/ directory.

**A computer screen with text and words

AI-generated content may be incorrect.Output:-**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**3. Passing Data to HTML Template**

**app.py**

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def home():

user = "John Doe"

return render\_template("welcome.html", username=user)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**templates/welcome.html**

<!DOCTYPE html>

<html>

<head>

<title>Flask Variables</title>

</head>

<body>

<h1>Hello, {{ username }}!</h1>

<p>Welcome to your personalized Flask page.</p>

</body>

</html>

✅ **Explanation:**

* Sends data (username) from Flask to the HTML template using Jinja2 templating

**Output:-**

**A screenshot of a computer

AI-generated content may be incorrect.**

**4. Rendering a List in an HTML Template**

**app.py**

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route('/')

def home():

items = ["Apple", "Banana", "Cherry", "Date"]

return render\_template("list.html", items=items)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**templates/list.html**

<!DOCTYPE html>

<html>

<head>

<title>Flask List</title>

</head>

<body>

<h1>Fruit List</h1>

<ul>

{% for item in items %}

<li>{{ item }}</li>

{% endfor %}

</ul>

</body>

</html>

✅ **Explanation:**

* Passes a **list** from Flask to the HTML template and loops through it using Jinja2.

**Output:-**

**A screenshot of a computer

AI-generated content may be incorrect.**

**5. Rendering a Form and Processing Input**

**app.py**

from flask import Flask, render\_template, request

app = Flask(\_\_name\_\_)

@app.route('/', methods=['GET', 'POST'])

def form():

if request.method == 'POST':

name = request.form['name']

return render\_template("greeting.html", name=name)

return render\_template("form.html")

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**templates/form.html**

<!DOCTYPE html>

<html>

<head>

<title>Flask Form</title>

</head>

<body>

<h1>Enter Your Name</h1>

<form method="post">

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

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

</form>

</body>

</html>

**templates/greeting.html**

<!DOCTYPE html>

<html>

<head>

<title>Greeting</title>

</head>

<body>

<h1>Hello, {{ name }}!</h1>

<p>Welcome to Flask!</p>

</body>

</html>

✅ **Explanation:**

* **Displays a form** that allows the user to enter their name.
* **Handles form submission** and renders a personalized greeting page.

**Output:-**

A white and black text

AI-generated content may be incorrect.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Experiment 14: Implementing Flask Projects.**

**A)** 1. User Authentication System

Features:

• Sign up, login, logout

• Password hashing (use Werkzeug)

• Role-based access (admin vs. regular user)

Tools: Flask, SQLite, Flask-WTF, Flask-Login

**Code:-**

from flask import Flask, render\_template\_string, redirect, url\_for, request, flash

from flask\_login import LoginManager, login\_user, logout\_user, login\_required, UserMixin, current\_user

from flask\_wtf import FlaskForm

from wtforms import StringField, PasswordField, SubmitField

from wtforms.validators import DataRequired

from werkzeug.security import generate\_password\_hash, check\_password\_hash

import sqlite3

app = Flask(\_name\_)

app.secret\_key = 'secret123'

# Setup Login Manager

login\_manager = LoginManager()

login\_manager.init\_app(app)

login\_manager.login\_view = 'login'

# Initialize Database

def init\_db():

with sqlite3.connect("users.db") as conn:

conn.execute('''CREATE TABLE IF NOT EXISTS users (

id INTEGER PRIMARY KEY AUTOINCREMENT,

username TEXT UNIQUE NOT NULL,

password TEXT NOT NULL,

role TEXT NOT NULL

)''')

init\_db()

# WTForms

class LoginForm(FlaskForm):

username = StringField("Username", validators=[DataRequired()])

password = PasswordField("Password", validators=[DataRequired()])

submit = SubmitField("Login")

class SignupForm(FlaskForm):

username = StringField("Username", validators=[DataRequired()])

password = PasswordField("Password", validators=[DataRequired()])

submit = SubmitField("Sign Up")

# User class for Flask-Login

class User(UserMixin):

def \_init(self, id, username, role):

self.id = id\_

self.username = username

self.role = role

def get\_id(self):

return str(self.id)

@login\_manager.user\_loader

def load\_user(user\_id):

with sqlite3.connect("users.db") as conn:

user = conn.execute("SELECT \* FROM users WHERE id = ?", (user\_id,)).fetchone()

if user:

return User(user[0], user[1], user[3])

return None

# Routes

@app.route('/')

def home():

return redirect(url\_for('login'))

@app.route('/signup', methods=['GET', 'POST'])

def signup():

form = SignupForm()

if form.validate\_on\_submit():

hashed = generate\_password\_hash(form.password.data)

try:

with sqlite3.connect("users.db") as conn:

conn.execute("INSERT INTO users (username, password, role) VALUES (?, ?, ?)",

(form.username.data, hashed, "user"))

flash("User created! Now login.", "success")

return redirect(url\_for('login'))

except:

flash("Username already exists!", "danger")

return render\_template\_string(SIGNUP\_TEMPLATE, form=form)

@app.route('/login', methods=['GET', 'POST'])

def login():

form = LoginForm()

if form.validate\_on\_submit():

with sqlite3.connect("users.db") as conn:

user = conn.execute("SELECT \* FROM users WHERE username = ?", (form.username.data,)).fetchone()

if user and check\_password\_hash(user[2], form.password.data):

login\_user(User(user[0], user[1], user[3]))

return redirect(url\_for('dashboard'))

flash("Invalid username or password", "danger")

return render\_template\_string(LOGIN\_TEMPLATE, form=form)

@app.route('/dashboard')

@login\_required

def dashboard():

if current\_user.role == "admin":

return f"<h2>Welcome Admin: {current\_user.username}</h2> <a href='/logout'>Logout</a>"

return f"<h2>Welcome User: {current\_user.username}</h2> <a href='/logout'>Logout</a>"

@app.route('/logout')

@login\_required

def logout():

logout\_user()

return redirect(url\_for('login'))

# HTML Templates (inline)

LOGIN\_TEMPLATE = '''

<h2>Login</h2>

<form method="POST">

{{ form.hidden\_tag() }}

{{ form.username.label }} {{ form.username() }}<br>

{{ form.password.label }} {{ form.password() }}<br>

{{ form.submit() }}

</form>

<a href="/signup">Sign up</a>

'''

SIGNUP\_TEMPLATE = '''

<h2>Sign Up</h2>

<form method="POST">

{{ form.hidden\_tag() }}

{{ form.username.label }} {{ form.username() }}<br>

{{ form.password.label }} {{ form.password() }}<br>

{{ form.submit() }}

</form>

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

'''

if \_name\_ == "\_main\_":

app.run(debug=True)

**Output:-**

