Type	No	Labs
Lab-1: Structuring Content with Table, Lists, and Div Tags		
A	1.	Create a weekly class timetable using HTML. Use the <table> tag to show the days and time slots clearly. Add some basic CSS to make the timetable look neat and easy to read.
A	2.	Create the HTML table given below that demonstrates the use of bold, italic, and underline text, along with rowspan and colspan. Apply basic CSS styling to enhance the table's appearance.
B	3.	Create an HTML document with an ordered list ( ), unordered list ( ), and description list (<dl>).

Data about Various Vehicles (Source: *Vehicle Data Source*).

	Name	Type	Price ($)	Year	Notes
Cars	Toyota Corolla	Compact Car	20,000	2024	**Reliable and fuel-efficient**
	Honda Accord	Midsize Car	25,000	2024	**Spacious and comfortable**
	Ford Mustang	Sports Car	35,000	2024	**Performance-oriented**
	Tesla Model S	Electric Car	95,000	2024	**High range and performance**
Motorcycles	Harley Davidson Iron 883	Cruiser	9,000	2024	**Classic design**
	Yamaha YZF-R6	Sportbike	12,000	2024	**Track-focused**
	Kawasaki KLR 650	*Adventure Bike*	7,500	2024	***Versatile***
Trucks	Ford F-150	Pickup Truck	30,000	2024	***Heavy-duty***
	Chevrolet Silverado 1500	Pickup Truck	32,000	2024	***Popular choice***
Summary by Category	Cars				**Total Vehicles: 4**
	Motorcycles				**Total Vehicles: 3**
	Trucks				**Total Vehicles: 2**
Overall Summary	**Total Vehicles**				**9**

Vehicle Overview

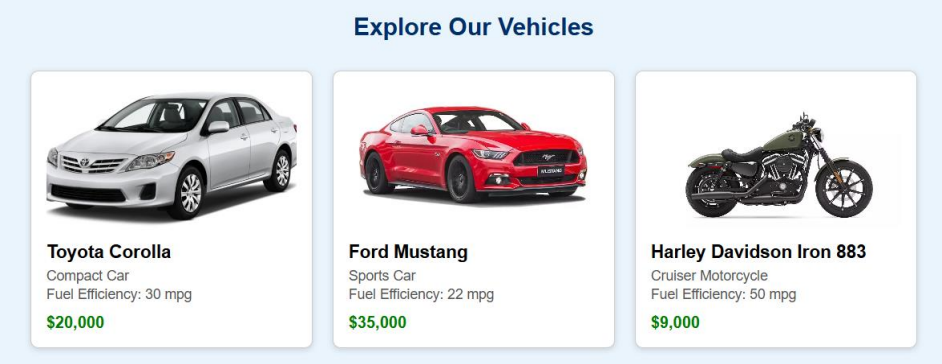
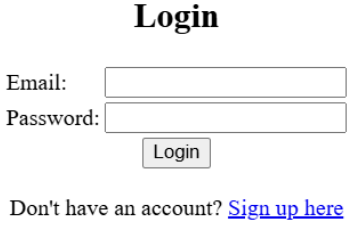
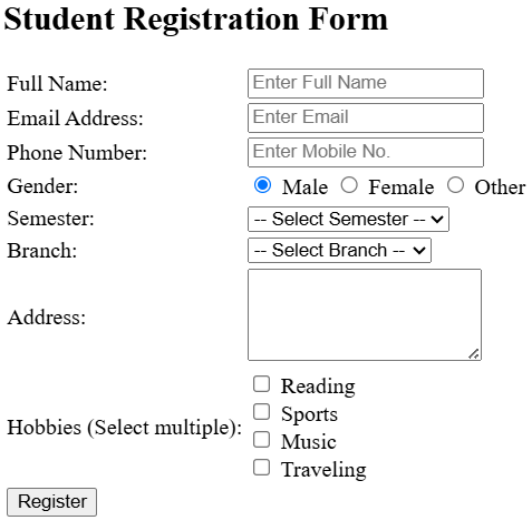
1. **Car**
   - Toyota Corolla
   - Honda Accord
   - Ford Mustang
2. **Motorcycle**
   - Harley Davidson Iron 883
   - Yamaha YZF-R6
3. **Truck**
   - Ford F-150

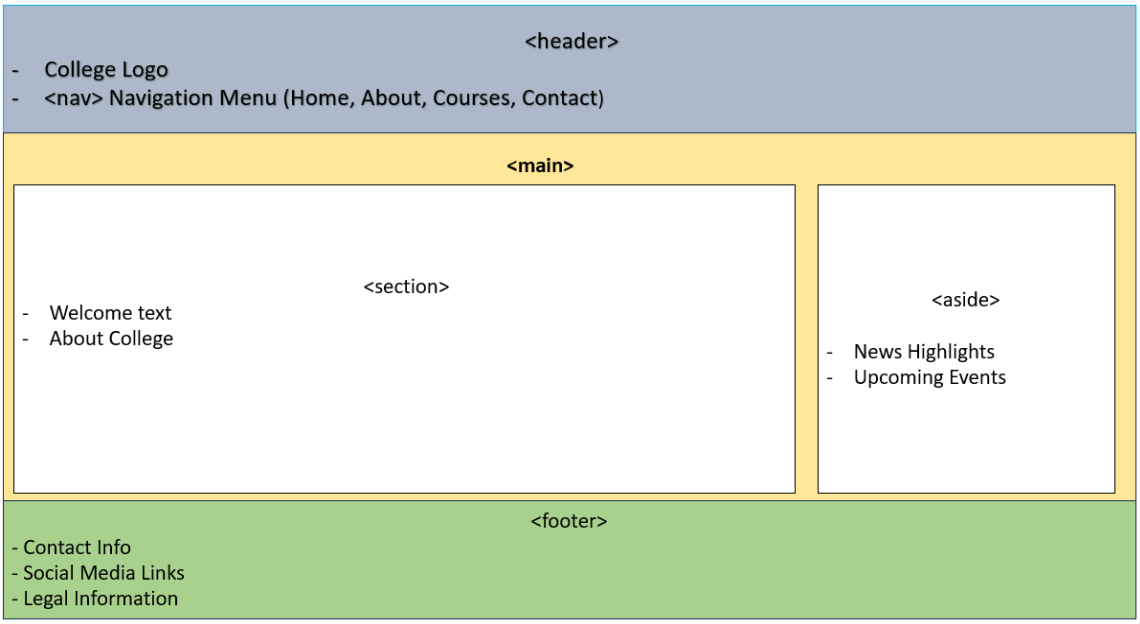
Vehicle Type Descriptions

Compact Car
Small and fuel-efficient, suitable for city driving.

Sportbike
High-performance motorcycle built for speed and agility.

Pickup Truck
Strong utility vehicle used for hauling and towing.

C	4.	Create an HTML document using the tag to organize content into sections. Apply basic CSS to style the sections. 
Lab-2: Design and Implementation of a Static Webpage Interface Using HTML and CSS		
A	1.	Design an HTML-based login interface that includes input fields for both username and password. 
B	2.	Create an HTML student registration page that includes following input fields such as name, email, phone number, gender, semester, branch, address and hobbies. 

B	3.	Create a basic layout for a static webpage using HTML5 tags, including <header>, <footer>, <nav>, <aside>, <main>, and <section>. Apply CSS Flexbox for the layout. 
C	4.	Implement the static webpage from Lab 2(3) with actual content, including a professional logo, images, and detailed information. Apply the necessary CSS to create an appealing webpage.
Lab-3: Implementation Static Blog Website Using Bootstrap		
A	1.	Create a Sign-In and Sign-Up pages using Bootstrap for the blog website.
A	2.	Design and implement the header and navigation bar using Bootstrap.
B	3.	Design an Add Post layout that includes input fields for the post title, description, file upload, and a submit button.
C	4.	Create the Blog Posts section that Include an image, publication date, author name, and a brief description for each post.
Lab-4: Demonstration of Basics of PHP Programming		
A	1.	WAP to display “Hello World”.
A	2.	WAP to display a message using variable.
A	3.	WAP to print value of variable of variables.
A	4.	WAP to print value of variable using string concatenation.

B	5.	Create a PHP script - Declare a variable $value and assign it an integer value (e.g., $value = 10;). - Use the gettype() function to display the type of $value. - Use settype($value, 'string') to change the type of $value to a string. - Use gettype() again to display the new type of $value. - Use var_dump($value) to display the value and type of $value after the conversion.
B	6.	Write a PHP program to demonstrate variable scopes: local, global, and static.
C	7.	WAP to swap values of two variables with the help of 3rd variable.
C	8.	WAP to swap values of two variables without using 3rd variable.
Lab-5: Implementation of Decision-Making Statements (Part – I)		
A	1.	WAP to check a person is eligible to vote.
A	2.	WAP to check whether the given number is odd or even.
A	3.	WAP to check whether the given number is positive, negative or zero.
A	4.	WAP to find greatest number from 2 numbers.
B	5.	WAP to find greatest number from 3 numbers.
B	6.	WAP to convert temperature from Fahrenheit to Celsius.
C	7.	WAP to print class of result based on percentage (i.e. less than 40% -> Fail, 40% to 50% -> Pass Class, 50% to 60% -> Second Class, 60% to 70% -> First Class, above 70% -> Distinction).
C	8.	WAP that reads a number in meters, converts it to feet, and displays the result.
Lab-6: Implementation of Decision-Making Statements (Part – II)		
A	1.	WAP to take a value from 1-7 and display current day using switch case. (If 1-Monday, 2-Tuesday, etc.).
A	2.	WAP to find a diameter from given area of circle.
B	3.	WAP to make a simple calculator using switch case.
B	4.	WAP to print class of result based on percentage using switch case (i.e. less than 40% -> Fail, 40% to 50% -> Pass Class, 50% to 60% -> Second Class, 60% to 70% -> First Class, above 70% -> Distinction).
B	5.	WAP to check to given year is a leap or not.

C	6.	WAP that check whether a letter is a vowel or consonants.
C	7.	WAP to check whether the three slide of triangle is isosceles, equilateral, scalene or right-angled triangle.
Lab-7: Implementation of Various Loops in PHP		
A	1.	WAP to print first n numbers using for, while and do while loop.
A	2.	WAP to print first n odd numbers using for, while and do while loop.
A	3.	WAP to demonstrate foreach loop.
B	4.	WAP to calculate and display sum and product of first N number.
B	5.	WAP to generate Fibonacci series of N number.
B	6.	WAP to check whether the given number is prime or not.
C	7.	WAP to check given number is Palindrome or not.
C	8.	WAP to check given number is Armstrong or not.
Lab-8: Implementation of Nested Loops in PHP		
A	1.	WAP to print following patterns: (a) * * * * * * * * * * * * * * * (b) 1 1 2 1 2 3 1 2 3 4 1 2 3 4 5 (c) 5 5 4 5 4 3 5 4 3 2 5 4 3 2 1 (d) 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
B	2.	WAP to print following patterns: (a) * ** *** **** ***** (b) * * * * * * * * * * * * * * * (c) ***** **** *** ** *

C	3.	WAP to print following patterns: 1 (a) 2 3 4 5 6 7 8 9 10 11 12 13 14 15 * (b) * * * * * * * * * * * * * * 1 2 3 4 5 (c) 2 3 4 5 3 4 5 4 5 5
Lab-9: Implementation of Array (Part – I)		
A	1.	WAP to create numeric array and print it.
A	2.	WAP to create associative array and print it.
B	3.	WAP to create multidimensional array and print it.
B	4.	Create a PHP script that performs the following tasks using array functions on a numeric array of student scores: $scores = [85, 78, 92, 67, 90]; - Shuffle the scores randomly and display the result. - Add a new score to the end of the list. - Remove the last score. - Remove the first score. - Add a new score to the beginning of the list.
C	5.	Create a PHP script that performs the following tasks using array functions on an associative array of student scores: $students = ["Alice" => 85, "Bob" => 78, "Charlie" => 92, "David" => 67, "Eve" => 90]; - Display the total number of students. - Print all student names and all scores separately. - Calculate and display the total and product of all scores. - Find the lowest and highest score in the list. - Check if any student has a score of 90. - Find the student who scored 92.

		- Sort the scores in ascending and descending order. - Sort the student names in alphabetical and reverse alphabetical order.
Lab-10: Implementation of Array (Part – II)		
A	1.	WAP to count number of even or odd number from an array of n number.
A	2.	WAP to find the average of n number using array.
B	3.	WAP to accept n numbers in an array. Display the sum of all the numbers which are divisible by either 3 or 5.
C	4.	WAP to accept n numbers in an array. Now, enter a number and search whether the number is present or not in the list of array elements by using linear search.
C	5.	WAP to sort given array in descending order without using inbuilt function.
Lab-11: Implementation of User Define Function (Part – I)		
A	1.	Write a PHP program to create a user-defined function that prints your name and call it.
A	2.	WAP to create user define function for adding two numbers and display the result.
B	3.	WAP to create calculator using all four types of UDF.
B	4.	WAP to calculate simple interest using method.
C	5.	WAP to generate Fibonacci series of N given number using method.
Lab-12: Implementation of User Define Function (Part – II)		
A	1.	WAP that calculates area of circle, triangle and square using user defined function.
A	2.	WAP to find maximum number from given three numbers using user defined function.
B	3.	WAP to accept a number and check whether the number is prime or not. Use user defined function name as check (int n). The method returns 1, if the number is prime otherwise, it returns 0.
C	4.	WAP to take two values as an input from the user and display all the prime numbers between the two given numbers using function.
Lab-13: Implementation of Recursion		
A	1.	WAP to print numbers from N to 1 using recursion.

A	2.	WAP to calculate sum of first n numbers using recursion.
B	3.	WAP to calculate factorial of a number using recursion.
B	4.	WAP to generate Fibonacci series of N number using recursion.
C	5.	WAP to check whether the number is prime or not using recursion.
Lab-14: Demonstration of Web Page Partition and Form Attributes		
A	1.	WAP to demonstrate the use of include, require, include_once and require_once.
B	2.	Demonstrate the use of PHP's include, require, include_once, and require_once functions to partition a webpage into different sections in Lab 2(4).
C	3.	Write a PHP program that creates an HTML form to do the following: - Place a text input field outside the <form> tag and associate it with the form using the input's form attribute and the form's id. - Use the required attribute on one or more input fields to ensure the user must fill them before submitting. - Add the novalidate attribute to the <form> tag while keeping the required attributes on inputs to observe the behaviour. - Use the autocomplete attribute on the <form> tag. - Add the accept-charset attribute to the <form> tag. - Utilize the target attribute with values like _self, _blank, _parent, _top, and a custom iframe name. - Set action="" or action="#" to submit the form to the same page. - Set action to a different PHP page to handle submission. - Test the form with different target values for each action.
Lab-15: Implementation of Form Processing		
A	1.	Design a student registration form and retrieve data in controller page using following Method: GET, POST, and REQUEST.
B	2.	Design an employee registration form and retrieve data in controller page using following Method: GET, POST, and REQUEST.
C	3.	Create form with all input types and retrieve data in controller page using following Method: GET, POST, and REQUEST.
Lab-16: Implementation of String Function		
A	1.	WAP to explore the string functions.

B	2.	Create a PHP script that performs the following tasks using given a string with some text, and you need to use the string functions listed below to perform various operations on it. $text = " PHP is a powerful scripting language for web development. "; - Find the length of the string. - Convert the string to uppercase and display the result. - Convert the string to lowercase and display the result. - Find the position of the word "powerful" in the string. - Replace the word "scripting" with the word "programming". - Extract the word "PHP" from the string. - Reverse the string and display the result. - Trim the leading and trailing spaces from the string. - Repeat the string "PHP " 5 times and display the result. - Implode the array of words ["PHP", "is", "a", "powerful", "language"] into a string, with a space separating each word. - Explode the string into an array using spaces as the delimiter, and display the array. - Shuffle the characters of the string randomly and display the result. - Convert newlines in a string to tags. Use the string "Hello\nWorld" for this task.
C	3.	Create a PHP script that performs the following operations on an array of names (strings). Use the array_map() function wherever applicable, and string functions to manipulate the array values: $names = [" Alice ", "JessiCA", " charlie", "DAVID", "eVa", "AleXa", "Olivia", " Levi "]; - Trim leading and trailing whitespace from each name. - Convert all names to lowercase. - Capitalize the first letter of each name. - Find and display the length of each name. - Reverse each name and display the reversed names. - Check if any name contains the substring "vi" (case-insensitive). Print those names. - Join all names into a single string separated by a comma and a space.
Lab-17: Implementation of Regular Expression and Server-Side Validation		
A	1.	Implement server-side validation on student registration form using PHP.
B	2.	Implement server-side validation on employee registration form using PHP.
C	3.	Implement server-side validation on form with all input types using PHP.
Lab-18: Implementation of File Upload		
A	1.	Create a webpage which accepts a file and upload it in specified folder on server.

B	2.	Design a profile page which allows changing profile picture dynamically.
C	3.	Create a webpage which accepts an image file in .jpg or .jpeg or .png format only and that to maximum of 1MB.
Lab-19: Implementation of File Handling		
A	1.	WAP to check the given file name is exits or not if exits than print the size of the give file.
A	2.	Write a program to create file named “students.txt” to store names of the students.
A	3.	Write a program to read names of the students from “students.txt” file.
B	4.	Write a program to create file named “employee.txt” to store empno, name, gender and mobileno. Open the same file again to display the content of the file.
B	5.	Write a file named “teachers.txt” and copy the content of file named “employee.txt” in it.
C	6.	Write a program to append the file named “students.txt” to add details of 3 new students in a file.
C	7.	Write a program to read a csv file named “result.csv”.
Lab-20: Implementation of Session Management using COOKIES		
A	1.	Create a PHP script that sets a cookie named username with the value (your name), and make it last for 1 hour.
B	2.	Make a script that counts how many times a user has visited the page by using a cookie.
B	3.	Create static login application using COOKIE in PHP.
C	4.	Create a webpage showing multiple items with add to cart and purchase functionality and on click of purchase it will redirect user to bill page that will show the detailed bill using COOKIES.
Lab-21: Implementation of Session Management using SESSION		
A	1.	Write a PHP script that starts a session and sets a session variable called username with the value (your name).
B	2.	Store multiple user details like username, email, and role in the session and display them on the profile page.
B	3.	Create static login application using SESSION in PHP.

C	4.	Create a webpage showing multiple items with add to cart and purchase functionality and on click of purchase it will redirect user to bill page that will show the detailed bill using SESSION.
Lab-22: Implementation of Basic SQL Commands and PHP Functions for Database Connectivity		
A	1.	Understanding and working with basic SQL commands on a sample database.
B	2.	Understanding and working with PHP functions for database connectivity.
C	3.	WAP to check database connectivity on a sample database.
Lab-23: Implementation of CRUD Operation using PHP		
A	1.	Implement CRUD database operation on student table from webpage using PHP.
B	2.	Implement CRUD database operation on employee table from webpage using PHP.
Lab-24: Implementations of Fully Functional Login Page with Remember Me Functionality		
A	1.	Implement server-side validation on login form using PHP.
B	2.	Apply database connectivity and verify the login credentials on the login page created in Lab 24(1).
C	3.	Implement cookie to provide remember me functionality in login page created in Lab 24(1).
Lab-25: Implementation of CRUD Operation on Registration Form		
A	1.	Create a student registration form to implement CRUD database operation.
B	2.	Create an employee registration form to implement CRUD database operation.
Lab-26: Implementation of Prepared Statement		
A	1.	Implement CRUD operations on students table using prepared statement in PHP.
B	2.	Modify the program Lab 25(1) to implement prepared statement in it.
Lab-27: Implementation of Stored Procedure		
A	1.	Implement CRUD operations on employee table using stored procedure in PHP.
B	2.	Modify the program Lab 25(1) to implement stored procedure in it.
Lab-28: Implementation of WordPress Theme, Post and Pages		
A	1.	Download and install WordPress on the XAMPP.

B	2.	Create a WordPress website for blogging, activate a suitable theme, and explore each customization setting appropriate for website.
B	3.	Create at least 3 WordPress posts with text, images, and different post formats, and 3 professional pages for your blog website.
C	4.	Explore and implement a comment section with options to approve, reject, and delete comments. Apply this functionality to both the post and page sections of a blog website.
Lab-29: Implementation of WordPress Categories, Tags and Widgets		
A	1.	Create, edit, and change categories and tags for blog website.
B	2.	Set the core settings of WordPress for the given content and add a new user.
C	3.	Use different Widgets depending on the posts and pages in blog website.
Lab-30: Implementation of E- Commence Website using WooCommerce Plugin		
A	1.	Create a WordPress website for e-commerce purpose, activate a suitable e-commerce theme, and explore each customization setting appropriate for blogs and install and active the WooCommerce plugin for e-commerce website.
A	2.	Add minimum 5 products and explore customized option setting while adding products.
A	3.	Create categories and tags for products of e-commerce website.
B	4.	Explore and customize settings for the product filter.
C	5.	Design coupon code for selling product and upload stock of item on site.