Syllabus on Vocational Education and Training Course (VTC)

Paper Title	**: Web Designing -I**
CODE	**: VTC: 243.1**
Number of Credits	**: 4**
Semester	**: III**
No. of Theory Hours Per Week	**: One (1 hour)**
No. of Practical Hours per Week	**: Three (3 Hours)**

Outline of the Paper:

Type of Course	Units in the VTC	Hours	Credits	Total Marks	Distribution of Marks (as per OC-8)			
					In-Semester		End-Semester	
					Theory	Practical	Theory	Practical
Web Designing - I	Unit-I Theory (25 Marks)	15	4	100	25			
	Unit-II to IV Theory (75 Marks)	90				15		60

Marks Distribution	**: Internal Assessment: 40** **: External Assessment: 60**
Course Objectives	1. Develop skills in designing and creating websites using essential tools and technologies for web development. 2. Distinguish between front-end and back-end development roles and responsibilities. 3. Implement front-end development techniques using HTML, CSS, and JavaScript to build interactive and visually appealing web pages. 4. Manage server-side processes and database interactions to ensure robust and secure back-end functionality. the front-end.
Course Learning Outcome	After the completion of the course the students are able to: 1. use fundamental skills to maintain web server services required to host a website 2. select and apply mark-up languages for processing, identifying, and presenting information in web pages 3. use scripting languages and web services to transfer data and add interactive components to web pages 4. create and manipulate the web Employ media objects using editing software 5. combine multiple web technologies to create advanced web components 6. design websites using appropriate security principles, focusing specifically on the vulnerabilities inherent in common web implementations.

Unit I: (Theory) **15 Hours**	**Introduction to Web Technology and Web Designing** • **Web Technology**: HTTP; System Architecture of a Web server; Client-side Scripting versus Server-side Scripting. • **Introduction to HTML**: What is HTML-HTML Documents- Basic structure of an HTML document. CSS: What is CSS, Structure of CSS. Advantages of CSS. • **Javascripts:** What is JavaScript? -Client-Side JavaScript -Advantages of JavaScript-Limitations of JavaScript.
UNIT-II: (Practical) **30 Hours**	**Hyper Text Markup Language (HTML5)** **1. HTML5 Basics:** Structure of an HTML5 document (<!DOCTYPE html>, <html>, <head>, <title>, <body>), Semantic elements (<header>, <nav>, <section>, <article>, <footer>, etc.) **2. Text and Multimedia:** Text formatting (headings, paragraphs, emphasis, etc.),Adding images (<img> tag) and multimedia content (<video>, <audio> tags), Using HTML entities for special characters **3. Links, Lists, and Tables:** Creating hyperlinks (<a> tag) and anchor links,Lists (unordered , ordered , and definition <dl> lists),Creating tables (<table>, <tr>, <th>, <td>) **4. Forms and Input Elements:** Building forms (<form> tag) with various input types (text, password, email, etc.), Radio buttons, checkboxes, and dropdown lists,Form validation using HTML5 attributes (required, pattern, min/max, etc.) **5. Media and Embedding:** Embedding multimedia content (videos, audio) from external sources, Using the <iframe> tag for embedding content from other websites **6. HTML5 APIs :**Geolocation API for obtaining user location, Canvas API for drawing graphics and animations, Local Storage and Session Storage for client-side data storage **7. Accessibility and SEO:** Importance of semantic HTML for accessibility and SEO, Using ARIA attributes for enhancing accessibility, Optimizing HTML for search engines (meta tags, title tags, alt attributes) **8. Responsive Design and Mobile Compatibility:** Creating responsive layouts using HTML5 and CSS3, Meta viewport tag for mobile responsiveness, Mobile-friendly forms and input elements **9. Advanced HTML5 Features:** Web components and

	custom elements,Drag and drop functionality, Web storage (local Storage, session Storage)
Suggested Practical Assignment:	**1. Create a Web Page Structure:** Design a web page structure using HTML5 semantic elements such as <header>, <nav>, <section>, <article>, <footer>, and <aside>. **2.** Create a web page for a cake shop to display all the different types of cakes and price to choose from. **3. Multimedia Embedding:** Embed an audio or video file using the <audio> or <video> tag with appropriate attributes like controls, autoplay, and loop. **4. Responsive Image Gallery:** Build a responsive image gallery using HTML5 <figure> and <figcaption> elements. Ensure that the gallery adjusts smoothly on different screen sizes. **5. Interactive Form Validation:** Develop an HTML5 form with input fields like text, email, password, and a submit button. Implement HTML5 form validation using attributes like required, pattern, and min/max. **6.** Create a HTML page with controls to take data for a College Admission with all the proper validations in the form. **7. Geolocation API Integration:** Implement the HTML5 Geolocation API to display the user's current location on a map or show nearby places based on latitude and longitude. **8. Local Storage Usage:** Create a web page that allows users to store data locally using HTML5 localStorage or session Storage. Develop functionality to add, edit, and delete stored items. **9.** Create a HTML Page to display the number of the times the web page was visited using local storage. **10. Semantic Markup for SEO:** Optimize an existing web page for search engines using semantic HTML5 tags. Use <header>, <nav>, <main>, <article>, <section>, <aside>, and <footer> tags appropriately.
UNIT-III: (Practical) **30 Hours**	**Cascading Style Sheets (CSS)** **1. Introduction to CSS:** What is CSS? Importance and benefits, CSS syntax: selectors, properties, and values, External, internal, and inline CSS **2. CSS Selectors and Specificity:** Basic selectors: element selectors, class selectors, ID selectors, Combinators: descendant, child, adjacent sibling, general sibling, Pseudo classes and pseudo-elements, CSS specificity and inheritance **3. CSS Box Model:** Understanding the box model:

	content, padding, border, margin, Box sizing: content-box vs. border-box, Margin collapsing **4. Layout and Positioning:** Display property: block, inline, inline-block, flex, grid, Position property: static, relative, absolute, fixed, sticky, Floats and clearing floats, CSS Grid and Flexbox layouts **5. Typography and Fonts:** Font properties: font-family, font-size, font-weight, font-style, line-height, Text properties: color, text-align, text-decoration, text-transform, letter spacing, word-spacing, Google Fonts and custom font usage **6. Colors and Backgrounds :**Color values: named colors, hexadecimal, RGB, RGBA, HSL, HSLA, Background properties: background-color, background-image, background repeat, background-position, background-size **7. Responsive Design and Media Queries:** Responsive design principles, Media queries syntax and usage, Designing responsive layouts for different screen sizes (mobile-first approach**)** **8. CSS Transitions and Animations:** Transition properties: transition-property, transition duration, transition-timing-function, transition-delay, CSS animations: keyframes, animation properties, animation-duration, animation-timing-function, animation-delay **9. Flexbox and Grid Layouts:** Flexbox properties: flex-direction, justify-content, align items, align-self, flex-grow, flex-shrink, CSS Grid properties: grid-template-columns, grid-template-rows, grid-gap, grid-template-areas **10. CSS Frameworks and Preprocessors:** Introduction to CSS frameworks (Bootstrap, Tailwind),Overview of CSS preprocessors (Sass): variables, mixins, nesting, inheritance **11. Advanced CSS Techniques:Transformations:** translate, rotate, scale, skew, CSS variables (custom properties), CSS gradients, shadows, and filters,Cross-browser compatibility and vendor prefixes
Suggested Practical on the topics	**1. CSS Selectors and Box Model:** ○ Create a webpage with different elements styled using basic selectors, class selectors, and ID selectors. Apply different properties such as background color, padding, border, and margin to understand the box model. **2. Layout and Positioning:** ○ Design a web page layout using CSS display properties (e.g., flexbox or grid) for header, navigation, content, and footer sections. Use positioning (static,

	relative, absolute) to position elements within the layout. **3. Typography and Fonts:** ○ Style text on a webpage with different font families, sizes, weights, styles, colors, and text alignments. Experiment with line height, letter spacing, and text decorations. **4. Colors and Backgrounds:** ○ Create a webpage with various background colors, gradients, images, and patterns. Apply different background properties such as background-size, background-position, and background-repeat. **5. Responsive Design with Media Queries:** ○ Develop a responsive webpage that adjusts its layout and styling based on different screen sizes using media queries. Test the responsiveness on mobile devices and desktop screens. **6. CSS Transitions and Animations:** ○ Add transitions to elements (e.g., hover effects) using CSS transition properties (transition-duration, transition-property, transition-timing-function). simple animations using keyframes and animation properties. **7. Flexbox and Grid Layouts:** ○ Design a webpage layout using CSS Flexbox properties (flex-direction, justify content, align-items) for a navigation menu or card-based layout. Create a grid based layout using CSS Grid properties (grid-template-columns, grid-templaterows, grid-gap). **8. Customizing CSS Frameworks:** ○ Customize a CSS framework (e.g., Bootstrap) by modifying variables, adding custom styles, and overriding default styles to create a unique design. **9. Advanced CSS Techniques:** ○ Implement CSS transformations (translate, rotate, scale, skew) on elements to create interactive effects. Use CSS gradients, shadows, and filters to enhance visual elements. ○ Optimize CSS code by minifying, concatenating, and compressing stylesheets. Use browser developer tools to debug and optimize CSS for performance.
UNIT-IV: (Practical) **30 Hours**	Java Scripts 1. **JavaScript Basics** :JavaScript syntax: variables, data types, operators, expressions, statements, Functions: defining functions, function expressions, arrow functions, Control flow: if statements, switch statements, loops (for, while) **2. Arrays and Objects:** Arrays: creating arrays,

	accessing elements, array methods (push, pop, shift, unshift, slice, splice),Objects: creating objects, object properties, methods, constructor functions, prototypes **3. DOM Manipulation:** Accessing DOM elements: get Element ById, querySelector, querySelectorAll, Manipulating DOM elements: changing content, styles, attributes, adding/removing elements **4. Events and Event Handling:** click, mouseover, keydown, submit, etc. Event listeners: adding event listeners, event propagation (bubbling, capturing) Handling user interactions with events **5. Forms and Validation:** Working with HTML forms in JavaScript, Form validation: validating input fields, displaying error messages, preventing default form submission **6. Error Handling:** Handling errors in JavaScript: try-catch blocks, Debugging JavaScript code using browser developer tools
Suggested Practical Assignments	**1. Basic JavaScript Concepts:** ○ Write JavaScript code to declare variables of different data types (string, number, boolean). ○ Implement arithmetic operations, comparison operators, and logical operators in JavaScript. **2. Functions and Control Flow:** ○ Create a function to calculate the factorial of a number using recursion. ○ Write a JavaScript program to check if a number is prime or not using a function. ○ Write a Javascript program to print all the perfect numbers from 1 to n. **3. Arrays and Objects:** ○ Create an array of numbers and write JavaScript code to find the sum, average, maximum, and minimum value in the array. ○ Define an object representing a person with properties like name, age, and country. Use object methods to display information about the person. 4**. DOM Manipulation and Events:** ○ Build an HTML form with input fields for username and password. Use JavaScript to validate the form on submission and display appropriate messages. ○ Create a webpage with a button that changes the background color of a div element when clicked using event handling. **5. Project-Based Assignments:** ○ Choose a project idea (e.g., interactive quiz, weather app, budget tracker) and implement it using JavaScript. Use concepts learned throughout the syllabus to build the project.

Suggested Readings	1. David Flanagan, "JavaScript: The Definitive Guide" by, O'Reilly Media. 2022. 8th Edition 2. Elizabeth Castro and Bruce Hyslop, "HTML and CSS: Visual QuickStart Guide", Peachpit Press, 9th Edition 3. Jennifer Niederst Robbins, “Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics”. 4. Marijn Haverbeke, “Eloquent JavaScript: A Modern Introduction to Programming”.
Requirements	• Computers • Software • Internet Access • External Storage • Printers and Scanners • Projector and Screens • **Any other item as required**
Qualified Instructors	• Instructors with experience in Web Designing and teaching. • Certifications or relevant qualifications in Web Designing

Paper Title	**: Web Designing -II**
CODE	**: VTC: 263.1**
Number of Credits	**: 4**
Semester	**:IV**
No. of Theory Hours Per Week	**: One (1 hour)**
No. of Practical Hours per Week	**: Three (3 Hours)**

Outline of the Paper:

Type of Course	Units in the VTC	Hours	Credits	Total Marks	Distribution of Marks (as per OC-8)			
					In-Semester		End-Semester	
					Theory	Practical	Theory	Practical
Web Designing - II	Unit-I Theory (25 Marks)	15	4	100	25			
	Unit-II to IV Theory (75 Marks)	90				15		60

Marks Distribution	**: Internal Assessment: 40** **: External Assessment: 60**
Course Objectives	1. The course is designed to impart knowledge and skill to the students to learn and know the principles and techniques of programming for the Web. 2. This course aims to build in the students a web programming knowledge using PHP and MySQL and Laravel Framework.
Course Learning Outcome	After completion of the course students are able to: 1. Gain a strong foundation in PHP programming language, including variables, data types, operators, control structures, functions, and arrays. 2. Learn how to handle forms, process user input, validate data, and interact with databases using PHP. 3. Master the Laravel framework and its key components such as routing, controllers, views, models, migrations, and middleware. 4. Understand Laravel's MVC (Model-View-Controller) architecture and how to use it to build scalable and maintainable web applications. 5. Learn how to integrate PHP and Laravel applications with databases (e.g., MySQL) and perform CRUD operations using Laravel's 6. Gain proficiency in defining database relationships, designing schemas, and using migrations for database management.
Unit I: (Theory) **15 Hours**	**Introduction to Server side Technology (Theory)** • **Server-side technology**: Overview of server-side vs. client-side technologies, Common server side languages and frameworks, Introduction to PHP and its role in web development, Setting up a development environment

	(e.g., XAMPP, WAMP). • **Database:** Overview of relational databases and Mysql, Advantages to Mysql. • **Frameworks:** What is a web frameworks ,Introduction to Laravel framework,History and evolution of Laravel, Features and advantages of using Laravel
UNIT-II: (Practical) PHP **30 Hours**	• **PHP:** Conditions and Branches, Loops, Functions, Working with types,User-defined Functions, Arrays, Strings and Advanced Data Manipulation in PHP, Arrays, Strings, Regular Expressions, Dates and Times, Integers and Floats. • **Object-Oriented Programming with PHP :** Classes and Objects, Inheritance, Throwing and Catching Exceptions Advance Features of Object-Oriented Programming in PHP : Working with Class Hierarchies, Abstract Classes and Interfaces.Session and Cookies in PHP. • **File Handling:** File open,close,read and write, File Manipulation (Rename,delete copying file), Uploading of files.
UNIT-III: (Practical) **30 Hours**	**Introduction to MySQL with PHP** • Working with MySQL: Database Basics, MySQL Command Interpreter, Managing Databases and Tables, Inserting, • Updating and Deleting Data, Querying with SQL SELECT, Join Queries, Querying Web Databases, Querying a MySQL Database using PHP, • Processing User Input Writing to Web Databases, Database Inserts, Updates and Deletes, • Issues in Writing Data to Database
UNIT-IV: (Practical) **30 Hours**	**PHP Frameworks** • **Introduction to Laravel: Overview** of Laravel framework, Installation and setup, basic folder structure, Introduction to MVC architecture. • **Routing and Controllers:** Routing basics, Route parameters, Route naming, Creating and using controllers, Controller methods. • **Views and Blade Templating:** Views in Laravel, Introduction to Blade templating, Blade directives and control structures, Blade layouts and partials. • **Database Integration:** Database configuration in Laravel, Using Eloquent ORM for database operations, Defining models and relationships, Querying the database using Eloquent. **Forms and Validation:** Creating forms in Laravel, Form validation using Laravel's validation rules, Displaying validation errors, Custom validation rules.

Suggested PHP Practicals	1. Declare variables for name, age, and email address. Print these variables with appropriate labels. 2. Write PHP code to perform arithmetic operations (addition, subtraction, multiplication, division) on two numbers. 3. Create a PHP script that checks if a user's age is greater than or equal to 18. If true, display "You are an adult"; otherwise, display "You are a minor". 4. Use a loop (for or while) to print numbers from 1 to 10 on separate lines. 5. Write a PHP program to print all the fibonacci series from 1 to n. 6. Create an array of fruits and use a loop to print each fruit on a new line. 7. Write a PHP function that takes two parameters (length and width) and calculates the area of a rectangle. Call the function with different values to test it 8. Create an HTML form with input fields for name and email. Write a PHP script to process the form data and display the submitted values. 9. Write PHP code to read the contents of a text file and display them on the screen. 10. Implement a PHP script that sets a session variable (e.g., username) when a user logs in. Display a personalized message using the session variable on subsequent visits.
Suggested Laravel Practical	1. **Routing and Controllers:** a. Create a new route that points to a controller method. The controller method should return a simple message or view. b. Implement route parameters in Laravel and use them to fetch data from a database in the controller method. 2. **Views and Blade Templating**: a. Create a new Blade template that includes a header, footer, and a content section. Extend this template in multiple views. b. Use Blade directives (if, foreach) to conditionally display content or loop through data in a view. 3. **Database Operations:** a. Create a migration to add a new table to the database schema. Run the migration to apply the changes. b. Implement CRUD operations (Create, Read, Update, Delete) for a resource (e.g., articles, users) using Laravel's Eloquent ORM. 4. **Form Handling and Validation:** a. Create a form to add new data to the database. Implement form validation using Laravel's validation rules. b. Display validation errors in the form and repopulate form fields with old input on validation failure. 5. **File Upload and Storage**: a. Build a form for users to upload files to the server. b. Implement functionality to store uploaded files in Laravel's storage system

Suggested Minor Project work (Students are to undertake one minor project for internal assesment)	1. **To-Do List Application:** Create a basic to-do list application where users can add, edit, delete, and mark tasks as completed. Implement user authentication so that each user has their own set of tasks. 2. **E-commerce Store:** Develop a simple ecommerce store with product listings, product details, shopping cart functionality, and checkout process. Implement user authentication and roles for customers and administrators. 3. **Contact Management System:** Build a contact management system where users can add, edit, delete, and search for contacts. Implement features like contact groups, import/export contacts, and contact sharing. 4. **Event Management System:** Develop an event management system for organizing and managing events. Include features such as event creation, registration, ticketing, attendee management, and event analytics 5. **Task Management System:** Create a task management system with task lists, task details, deadlines, priorities, and task assignments. Implement notifications and reminders for upcoming tasks.
Suggested Readings	1. DuBoi Paul s, MySQL Cookbook. 2. DuBois Paul, My SQL Cookbook 3. Pecoraro,Christopher John Mastering Laravel: A Comprehensive Guide to Laravel's Best Practices. 4. Stauffer,Matt Laravel: Up & Running. 5. Welling Luke and Laura Thomson, PHP and MySQL Web Development. 6. Welling Luke and Laura Thomson, PHP and MySQL Web Development.
Requirements	1. Computers 2. Desktop computers or laptops with adequate RAM and processing power. 3. Required software installed (e.g., PHP development environment, MySQL, Laravel, text editors). 4. Software: XAMPP, WAMP, or similar for local server setup (Apache,MySQL, PHP). 5. IDEs (Integrated Development Environments) such as Php Storm, Visual Studio Code, or Sublime Text for PHP and Laravel development. 6. Database Management MySQL or Maria DB for database management php MyAdmin or MySQL Workbench for database administration 7. Internet Access 8. External Storage 9. Printers and Scanners 10. **Any other item as required**

Qualified Instructors	• Instructors with experience in Web Designing and teaching. • Certifications or relevant qualifications in Web Designing

Paper Title	**: Web Designing -III**
CODE	**: VTC: 363.1**
Number of Credits	**: 4**
Semester	**:VI**
No. of Theory Hours Per Week	**: One (1 hour)**
No. of Practical Hours per Week	**: Three (3 Hours)**

Outline of the Paper:

Type of Course	Units in the VTC	Hours	Credits	Total Marks	Distribution of Marks (as per OC-8)			
					In-Semester		End-Semester	
					Theory	Practical	Theory	Practical
Web Designing - III	Unit-I Theory (25 Marks)	15	4	100	25			
	Unit-II to IV Theory (75 Marks)	90				15		60

Marks Distribution	**: Internal Assessment: 40** **: External Assessment: 60**
Course Objectives	1. Understand the fundamentals of front-end development and the role of React in modern web development. 2. Learn how to set up a development environment for React applications using tools like Node.js and npm. 3. Gain proficiency in creating reusable components, managing state, and handling events in React. 4. Explore advanced React concepts such as hooks, context API, and component lifecycle methods. 5. Develop skills in building responsive and interactive user interfaces using React and CSS.
Course Learning Outcome	After completion of the course students are able to: 1. design and develop dynamic and interactive user interfaces using React components and JSX syntax 2. implement state management solutions using React's built-in state management and context API 3. utilize React Router for handling navigation and creating multi-page applications 4. integrate third-party libraries, APIs, and services into React applications for enhanced functionality 5. apply responsive design principles and CSS techniques to create visually appealing and mobile-friendly interfaces.
Unit I: (Theory) 15 Hours	• **Introduction to React:** Understanding the React ecosystem, Virtual DOM, and JSX, React • **Components and Props:** Creating components, passing props, and composing UIs, • **State Management:** Understanding state, setState, and component re-rendering. React Lifecycle • **Methods:**Exploring component DidMount, component DidUpdate.

UNIT-II: **(Practical)** **30 Hours**	**Concepts Covered** • Component creation and rendering • State management with React hooks • Handling user interactions and events • Asynchronous data fetching with APIs • Form handling, validation, and dynamic updates • React Router for client-side navigation
	Sample exercises and assignments • **Exercise:** Build a simple To-Do list application using React components, state management, and event handling. • **Assignment:** Create a weather forecast application using React components to display weather data fetched from an API. • **Exercise:** Develop a user registration form with form validation using React hooks and controlled components. • **Assignment:** Build a movie search application using React and an external API to search and display movie details.
UNIT-III: **(Practical)** **30 Hours**	**Concepts Covered** • Structuring components for scalability • State management for dynamic data with Redux or Context API • Implementing CRUD operations for data manipulation • Utilizing React Router for multi-page applications • Filtering and sorting data for enhanced user experience
	Sample exercises and assignments • **Exercise:** Implement a simple e-commerce product listing page with React components and state management. • **Assignment:** Create a blog application where users can view, create, and delete blog posts using React Router for navigation. • **Exercise:** Build a chat application using React components and state to display messages in real-time. • **Assignment:** Develop a recipe finder application using React to search and display recipes fetched from an API.
UNIT-IV: **(Practical)** **30 Hours**	**Concepts Covered**: • Building multi-step forms with conditional rendering • Designing responsive layouts for various devices • Implementing user authentication and authorization with JWT tokens • Securing routes and resources based on user roles
	Sample exercises and assignments • **Exercise:** Create a multi-step form using React Router for

	navigation between form sections. • **Assignment:** Build a portfolio website using React to showcase projects and skills with responsive design. • **Exercise:** Implement user authentication and authorization in a React application using JWT tokens. • **Assignment:** Develop a social media dashboard application with React components to display user posts and interactions.
Suggested Readings	1. Kirupa Chinnathambi, Learning React: A Hands-On Guide to Building Web Applications Using React and Redux, 2nd Edition, Addison-Wesley Professional, 2018 2. Michele Riva, React Design Patterns and Best Practices: Build easy to scale modular applications using the most powerful components and design patterns, 2nd Edition, Packt Publishing, 2019
Requirements	**Materials:** 1. Computers 2. Node.js for running JavaScript on the server (needed for npm or yarn) 3. phpMyAdmin or MySQL Workbench for database administration 4. Internet Access 5. External Storage 6. Printers and Scanners 7. Any other item as required
Qualified Instructors	• Instructors with experience in Web Designing and teaching. • Certifications or relevant qualifications in Web Designing