Course Outline

Course Name	Web System and Technology
Credit Hours	3
Instructor	M Abo Bakar Aslam muhammad.abobbakar@uog.edu.pk
Course Objectives	In a course focused on front-end web development using React, the primary goals are to equip students with a strong understanding of web technologies and React fundamentals. They should become proficient in creating component-based user interfaces, implementing client-side routing with React Router, managing state effectively, integrating with external APIs, and optimizing performance. Students will also learn to design responsive and accessible user interfaces and practice testing and debugging techniques. Practical project development is emphasized, and they should gain experience in version control, deployment, and building a professional portfolio. Collaboration and soft skills development are encouraged, preparing students for successful careers in web development.
Reference Materials	https://react.dev/learn https://www.w3schools.com/react/default.asp https://javascript.info/ https://sass-lang.com/documentation/ https://tailwindcss.com/docs https://getbootstrap.com/docs/
Solutions for given tasks in each lecture will be available on https://coursecs.wordpress.com/courses/web-development/	

Lectures and their Contents

Lecture	Contents and Practical Tasks
1	**Introduction about Course** • Motivation about Web application development • Introduction about different frameworks/techniques for web application development • Motivation about JavaScript and React based web application development • Introduction about tools used for JavaScript and React based web application development
2	**Setting up Environment for Implementation of JavaScript Code** • Install any code editor. In this course, we recommend Visual Studio (VS) Code. Download this tool by using https://code.visualstudio.com/download and install it. Moreover, install necessary extension in VS code. • Install "React Developer Tool" extension for your browser • Download and install node.js. Download it by using https://nodejs.org/en/download/current
JavaScript Basics	
3	**Variables, Data Types and Tasks-To-Do** • JavaScript Variables can be declared in 4 ways Automatically | Using var | Using let | Using const • Variable Declaration, Initialization and Display • Variable Reassignment and Display • JavaScript has 8 Datatypes String | Number | Bigint | Boolean Undefined | Null | Symbol | Object

	• The Object Datatypes Object | Array | date • Tasks-To-Do 1. Declare a variable called “name”, initialize it with your name and then display your name 2. Declare a variable called “age”, initialize it with your age and then display your age 3. Create an array called fruits having names of your favorite fruits and display each element without using loop 4. Declare a variable called currentYear, initialize it with the current year (e.g., 2023), reassign the value of currentYear to the next year and display it. 5. Create a variable called isRainyDay, set it to true or false based on the weather and display its value 6. Create an object called personInfo with properties like name, age, and city. Display all data of the object.
4	**Loops, Functions and Tasks-To-Do** • Basics for Loop o Write a for loop that counts from 1 to 10 and displays each number in the console. • Loop Through an Array: o Create an array of your favorite fruits. o Use a for loop to iterate through the array and display each fruit in the console. • While Loop: o Write a while loop that counts from 1 to 5 and displays each number in the console. • Looping Backwards: o Create an array of numbers from 10 to 1. o Use a for loop to iterate through the array in reverse order and display each number. • Loop with Conditional Statement: o Write a for loop that counts from 1 to 20. o For each number, check if it's even or odd and display a message in the console. • Function Basics: o Create a function named as “greet” that takes a name as a parameter and returns a greeting message (e.g., "Hello, [name]!"). o Call the function with different names and display the greetings in the console. • Function with Multiple Parameters: o Write a function called calculateArea that takes the length and width of a rectangle as parameters and returns the area. o Call the function with different values and display the results. • Function with Default Parameter: o Create a function called sayHello that takes a name as a parameter and displays a greeting message. o If no name is provided, it should default to "Guest." • Function as a Variable: o Declare a function called multiply that multiplies two numbers. o Create a variable called operation and assign the multiply function to it. o Use the operation variable to perform a multiplication. • Function within a Function: o Write a function called square that calculates the square of a number. o Create another function called cube that uses the square function to calculate the cube of a number. o Call the cube function with a number and display the result.
5	**Object’s Concepts and Tasks-To-Do (Part A)** • Object Creation o Create an object called person with properties such as name, age, city, and email. Fill in the values with your own information. • Accessing Object Properties o Write JavaScript code to access and display the name and age properties of the person object. • Object Methods o Add a method called greet to the person object. The greet method should display a greeting message using the name property • Object Iteration o Create an object called student with properties like name, age, grade, and subjects (an array of subjects the student is enrolled in). And now use a “for...in” loop to iterate through the properties of the student object and display them. • Object Manipulation o Add a new property to the person object called phoneNumber. Set it to your phone number. o Update the age property of the person object to reflect your current age. • Object Comparison o Create two objects, book1 and book2, each with properties like title, author, and year. Make sure they have some similar properties and some different ones. o Write a function that compares these two objects and checks if they are equal (have the same properties and values).
6	**Object’s Concepts and Tasks-To-Do (Part B)** • Nested Objects: o Create an object called school with properties like name, location, and students.

	○ The student's property should be an array of student objects (similar to the student object you created earlier). ○ Write code to access and display the information of a specific student within the school object. • Object Destructuring: ○ Given an object with properties like firstName, lastName, and email, use object destructuring to extract and display these properties. • Object Serialization: ○ Convert the person object into a JSON string using JSON.stringify(). ○ Then, parse the JSON string back into an object using JSON.parse(). • Object Cloning: ○ Create a new object called personCopy and copy the properties of the person object into it. Ensure that personCopy is a separate object and any changes made to one object do not affect the other.
7	**Array's Concepts and Tasks-To-Do (Part A)** • Array Initialization: ○ Create an array called fruits with the names of your favorite fruits. • Array Access: ○ Access and display the first and last elements of the fruits array. • Array Length: ○ Calculate and display the length of the fruits array. • Array Modification: ○ Add a new fruit to the end of the fruits array. ○ Remove the first fruit from the array. ○ Display the updated array. • Array Iteration: ○ Use a for loop to iterate through the fruits array and display each fruit in the console. • Array Sorting: ○ Create an array of numbers (e.g., [5, 2, 8, 1, 9]) and use the sort() method to sort it in ascending order. ○ Display the sorted array. • Array Filtering: ○ Create an array of numbers and use the filter() method to create a new array containing only even numbers. ○ Display the filtered array. • Array Mapping: ○ Create an array of numbers and use the map() method to double each number in the array. ○ Display the modified array. • Array Reduction: ○ Create an array of numbers and use the reduce() method to calculate the sum of all numbers in the array. ○ Display the sum.
8	**Array's Concepts and Tasks-To-Do (Part B)** • Array Searching: ○ Create an array of names and use the indexOf() method to check if a specific name exists in the array. ○ Display whether the name was found or not. • Multi-dimensional Arrays: ○ Create a multi-dimensional array to represent a tic-tac-toe board. Initialize it with empty spaces. ○ Modify elements in the array to simulate moves (e.g., "X" and "O"). • Array Sorting with Objects: ○ Create an array of objects, each representing a person with properties like name and age. ○ Use the sort() method to sort the array of objects based on the age property in ascending order. ○ Display the sorted array of objects. • Array Manipulation: ○ Create an array and use various array methods (e.g., push, pop, shift, unshift, splice) to add, remove, and modify elements in the array. • Array Concatenation: ○ Create two arrays and use the concat() method to combine them into a single array. ○ Display the concatenated array. • Array Copying: ○ Create an array and use different methods (e.g., slicing, spreading) to create a copy of the array without modifying the original.
9	Callbacks • Simple Callback Function ○ Create a function called sendMessage that takes a message and a callback function as parameters. The function should call the callback function with the message. • Error Handling with Callbacks: ○ Modify the sendMessage function from the first exercise to introduce error handling. If an error occurs, call a different callback function to handle the error.
10	Promises • Callback Promisify: ○ Take a callback-based function (e.g., fs.readFile for reading a file) and create a Promise-based version of it using a custom function.

	• Callback vs. Promise: o Create two versions of a function, one using callbacks and the other using Promises, to perform a simple asynchronous task (e.g., fetching data from an API). Compare the readability and ease of use of both versions.

11	Async/Await
	• Callback Hell (Callback Pyramid):
	  o Create a series of nested callback functions to simulate a callback hell scenario. For example, you can simulate asynchronous operations like fetching user data, their posts, and comments on each post, all nested within callbacks.
	• Parallel Callback Execution:
	  o Create an array of functions that simulate asynchronous tasks. Use a loop and callbacks to execute all tasks in parallel and collect the results in an array.

Assigning projects to student that will cover almost all concepts of JavaScript that you studied above.	
HMTL, CSS, Bootstrap and Tailwind	
12	HTML and Tasks-To-Do • Elements and Tags • Document Structure (head and body) • Comment • HTML Block and Inline Elements • Heading • Paragraph • Lists (ul, ol, li) • Table • Links and anchor tags **Tasks-To-Do:** o Practice above contents by creating a web page. Practice more content to do tasks for next lecture.
13	Tasks-To-Do associated with previous lectures only Create a web page on which you will display • Your Name • Introduction about yourself • An ordered list of your hobbies (at least 5 are listed) • An ordered list of awesome places of Pakistan you visited. Create a nice-looking CV in which you will include followings. In your CV, above all concepts of HTML must be implemented. But if you want to add more elements to make your CV more nicely, then it will be highly appreciated (Teacher will set some bonus marks depending upon added elements and your output result). • Full Name • Email • Profile picture • Educational history in tabular form • Professional skills in unordered list • List of projects you developed • Hobbies in form of ordered list • Your Life-Ambition • What you did for your country • What do you want to do for your country

14	CSS and Tasks-To-Do
	• CSS syntax and selectors
	• Inline, internal, and external CSS
	• background-color
	• color
	• font-family
	• font-size
	• text-align
	Tasks-To-Do:
	Update your CV created till previous Week such that you will add followings to make your CV more nicely. In this way, you will create the CV in form of a structured layout
	• Margin
	• Border
	  o Whole around the CV
	  o Border around any section (up to you) to make the CV awesome
	• Padding

<table>
<tr><td></td><td>

- Width for different sections

</td></tr>
<tr><td rowspan="2">15</td><td>Responsive Designing</td></tr>
<tr><td>

CSS Box Model and Layout

- Understanding the box model
- CSS layout properties (margin, padding, width)

CSS Styling and Positioning

- CSS layout techniques (float, display, position)

Responsive Web Design with CSS

- Introduction to responsive design
- Media queries and flexible layouts

Update your CV (created till previous task) in following context so that it will be converted a proper and basic website layout

- Your name should be in header section
- Your name and email should be displayed in footer section. Moreover, icons for Facebook, Instagram, LinkedIn and GitHub would also be shown in footer section. Name should be shown at top, email should be shown below name and all icons should be shown (in only one line) below email.
- Remaining content should be divided into three columns i.e., left, middle and right
  - Left column should have following properties
    - Width of 25%
    - This section will contain your profession skills
  - Right column should have following properties
    - Width of 25%
    - This section will contain your projects
  - Middle column should have following properties
    - Width of 50%
    - This section will contain all remaining content of the CV

Update your CV (created till previous task) such that its different sections would be styled and contained positions according to followings.

- Add just an icon for WhatsApp in your CV and fixed this icon to button right position.
- Set your header section with position as sticky so that your name will always be shown even if webpage is scroll down/up.
- Your profile picture should be right floated in middle column.

Update your CV (created till previous task) by using media queries to make to increase design-flexibility for all type of devices. For simplicity, just target three devices (mobile, tablet and desktop).

➢ Teacher will set some bonus marks depending upon added elements and your output result

</td></tr>
<tr><td rowspan="2">16</td><td>Bootstrap and Tasks-To-Do</td></tr>
<tr><td>

Add a horizontal navigation bar into your CV (created till previous Week) by using Bootstrap. It should contain following links (not active at this learning stage)

- Home
- Portfolio
- Projects
- JavaScript
- Result

Update your CV (created till previous task) such that it will be styles using Bootstrap. It is up to you that how much you style the CV by using Bootstrap classes (Teacher will set some bonus marks depending upon added classes and your output result)

</td></tr>
<tr><td colspan="2">Assigning projects to student that will cover concepts related to HTML, CSS and Bootstrap only.</td></tr>
<tr><td rowspan="2">17</td><td>Tailwind CSS and Tasks-To-Do</td></tr>
<tr><td>

- Introduction
- Benefits
- How to use it
- Setting up Tailwind CSS in a project
- Fonts and Sizing
- Margin, Border and Padding
- Creating responsive design

Recreate your CV by using Tailwind

Add more and more content to style your CV. Teacher will set some bonus marks depending upon added elements and your output result

</td></tr>
<tr><td rowspan="2">18</td><td>Completing Tasks-To-Do for Tailwind</td></tr>
<tr><td>

- Set some decent font in current project by using Tailwind CSS
- Set some font size for heading, paragraph and other content in current project by using Tailwind CSS
- Set margin, border and padding in current project by using Tailwind CSS
- Convert the project as responsive for all type of devices (mobile, tablet and desktop) by using Tailwind CSS.

➢ Add more elements to make your CV more nicely. Teacher will set some bonus marks depending upon added elements and your output result

</td></tr>
</table>

React	
19	React, Expression in JSX • Motivation about React based web application development • Introduction about tools used for React based web application development • Setting up environment for React based web application development • Understanding about file structure of your created-app in previous Week • Run the app and understand its basic working • Understand the basic code written in your created-app
20	React based Tasks-To-Do • Create a blank webpage • Create a variable in which you will store your name. Now display "Welcome YOUR-NAME" on the page. Your name will be displayed by using variable you created. • Create a variable in which you will store your marks in previous class. Now display "Marks: YOUR-MARKS" on the page. Your marks will be displayed by using variable you created. • Create five variables in which you will store marks (out of 100) for five different subjects. Now display marks for each subject along with subject name. Moreover, display total marks obtained in all subjects. • Regenerate task 1 in which you displayed your name by using string variable. Store your first, middle and last names in three different variables. Display your full name
21	React Components • Creating and rendering multiple components • Creating a component inside another component • Splitting component into different files and then render multiple components
22	Tasks-To-Do for Previous lecture At this stage, your project is contained multiple tabs in navigation bar. Therefore, create respective components i.e., one component for each tab. And now run the project. The result must be same as without using component-based structure.
23	React Router and Tasks-To-Do • Activate link in navigation bar • Render different components and show respective webpages Create following components having appropriate function relate to its working/nature. • Home; show detail about yourself, list of all concepts that your learnt in this course yet, list of your mini projects till now and detail about your expectations related to this course. • Portfolio • Projects; List of all projects developed by you • Contact Me; • JavaScript; a heading "JavaScript Concepts" only • Result; a form containing labels for name, email, name of five subjects, marks for each subject. Input fields for name, email, five subjects and marks for each subject. Button that will be used to submit the form. show result of a students in which student's name, his/her subjects, total marks for each subject, total obtained marks and percentage of marks. Activate links in navigation bar for navigating the above webpages.
Assign project related to react	
24	Props • Importance of props • How to pass props to a component • How to specify default values for props • How to pass an object as props
25	Tasks-To-Do for Previous Lecture • Create a component that will generate result of a student. We will have to give it name of student, email and marks of five subjects (max of 100 marks for each subject). The component will display student name, his/her email, marks w.r.t different subjects, actual total marks, obtained total marks and percentage of marks. • If only name and email of student is sent to the component, then marks 0 is used as by-default. • Create the component such that if either name or email is not sent to the component, then the component will generate an error instead of displaying any result sheet.

26	Handling Forms
	• Component state
	• Control changes in form-data by using event-handler
	• Hooks
	  ○ useState
	  ○ useEffect
	  ○ useCallback

27	Tasks-To-Do for Previous Lecture • Getting data from form you create in your project and display the data as it is received. • Perform some operations on data, then show result. For example, show percentage of marks with respect to received data from form.

28	JavaScript More
	• Asynchronous JavaScript
	  o Understanding asynchronous code
	  o Callback functions
	• Promises and Fetch API
	  o Promises and their use in asynchronous programming
	  o Fetching data from APIs using the Fetch API
	• Document Object Model (DOM)
	  o Manipulating the DOM with JavaScript
	  o Dynamic content and forms

29	Tasks-To-Do for Previous Lecture • Create an interactive webpage with event listeners • Build a webpage that dynamically updates content with DOM manipulation • Enhance a form on your webpage with JavaScript validation.
30	Completing your project based on react
31	Presenting your project
32	Presenting your project