CMR INSTITUTE OF TECHNOLOGY

USN | | | | | | | | | | |

Internal Assesment Test – I

Answer Key

Sub:	**Web Development using Full Stack Open – MMCB311A**							**Code:**	**MMCB311A**
Date:	**05.11.25**	**Duration:**	90 mins	**Max Marks:**	50	**Sem:**	I	**Branch:**	MCA

Answer Any One FULL Question from each part.

PART I

		Marks	OBE CO	OBE RBT
1.	Discuss the importance of forms in HTML and form validation using JavaScript.	10	CO1	
2	Describe JavaScript fundamentals with suitable examples.	10	CO1	
3	Explain different CSS units and color models.	10	CO1	
4	Compare inline CSS, internal CSS, and external CSS with examples.	10	CO1	
5	Write code to develop a small webpage integrating HTML, CSS, and JavaScript.	10	CO2	
6	Explain the concept of responsive web design.	10	CO2	
7	Explain the key features of React and why it is preferred for modern web development.	10	CO2	
8	What is JSX? How does it differ from HTML?	10	CO2	
9	Describe conditional rendering in React with examples.	10	CO2	
10	How are user events handled in React?	10	CO2	

Solution

1) Discuss the importance of forms in HTML and form validation using JavaScript.

HTML forms are used to collect user input and send it to the server for processing.
Key points:
- HTML provides tags like <form>, <input>, <select>, <textarea>, <button>.
- Attributes like action, method, and required define form behavior.
- JavaScript enables client-side validation to ensure correct data entry before submission.
Example:

```
<form onsubmit="return validateForm()">
  <input type="text" id="name" required>
  <input type="email" id="email" required>
  <button type="submit">Submit</button>
</form>
<script>
function validateForm(){
  let email = document.getElementById("email").value;
  if(!email.includes("@")){
    alert("Invalid Email!");
    return false;
  }
  return true;
}
</script>
```

Explanation: The script checks email validity before submitting the form.

2) Describe JavaScript fundamentals with suitable examples.

JavaScript is a scripting language for web interactivity.
- Variables: Declared using var, let, const.
- Data Types: number, string, boolean, object, array.
- Operators: arithmetic, logical, comparison.
- Control Structures: if, for, while, switch.
- Functions: Modular code units.
Example:

```
let name = "CMR";
for (let i = 0; i < 3; i++) {
  console.log("Hello " + name);
}
```

Output: Prints “Hello CMR” three times.

3) Explain different CSS units and color models.

- Absolute Units: px, cm, mm, pt (fixed size).
- Relative Units: %, em, rem, vw, vh (scale with context).
- Color Models:
  - RGB: rgb(255, 0, 0)
  - RGBA: rgba(0, 0, 255, 0.5)
  - HEX: #FF6600
  - HSL: hsl(120, 100%, 50%)

Example:
h1 { color: #ff6600; }
p { color: hsl(200, 100%, 40%); }
Explanation: Different color models give design flexibility.

4) Compare inline CSS, internal CSS, and external CSS with examples.

Type | Syntax | Advantage | Disadvantage
Inline | <h1 style="color:red;">Hello</h1> | Quick and simple | Hard to maintain
Internal | <style>p{color:blue;}</style> | Good for single page | Not reusable
External | <link rel="stylesheet" href="style.css"> | Reusable across pages | Extra file dependency
Example:
- Inline: Text
- Internal:
<style> body { background: lightgray; } </style>
- External: style.css linked using <link> tag.

5) Write code to develop a small webpage integrating HTML, CSS, and JavaScript.

```
<!DOCTYPE html>
<html>
<head>
  <title>Integration Example</title>
  <style>
    body { background:#f0f0f0; font-family:Arial; }
    button { background:blue; color:white; padding:10px; }
  </style>
</head>
<body>
  <h1>Welcome to Web Development</h1>
  <button onclick="alert('Hello from JavaScript!')">Click Me</button>
</body>
</html>
```

Explanation: HTML defines structure, CSS handles styling, JavaScript adds interactivity.

6) Explain the concept of responsive web design.

Responsive design makes websites adaptable to various devices.
Techniques include:
- Fluid Grids: Layout adjusts proportionally.
- Flexible Images: Resize automatically.
- Media Queries: Apply styles based on screen width.
Example:

```
@media (max-width:600px) {
  body { background: lightblue; }
}
```

Explanation: Layout and design change dynamically on smaller screens.

7) Explain the key features of React and why it is preferred for modern web development.

- Component-based Architecture: Reusable UI components.
- Virtual DOM: Efficient rendering.
- JSX: Simplified syntax for defining UI.

- Unidirectional Data Flow: Predictable state management.
- Advantages: Speed, modularity, and performance make React a top choice.

8) What is JSX? How does it differ from HTML?

JSX (JavaScript XML) is a syntax extension that allows writing HTML-like code inside JavaScript.
Differences from HTML:
- Uses className instead of class.
- JavaScript expressions inside {}.
- Must return a single parent element.
Example:
const App = () => <h1>{'Hello, React!'}</h1>;
Explanation: JSX simplifies UI declaration and compiles to JavaScript.

9) Describe conditional rendering in React with examples.

Conditional rendering means displaying components based on conditions.
Techniques:
- Using if-else
- Using ternary (? :) operator
- Using logical AND (&&)
Example:
{isLoggedIn ? <h1>Welcome</h1> : <LoginForm />}
Explanation: Displays “Welcome” if user is logged in, else shows login form.

10) How are user events handled in React?

React uses synthetic events for consistent event handling.
- Common handlers: onClick, onChange, onSubmit.
- Event binding methods: arrow functions or .bind(this).
Example:
<button onClick={() => alert('Clicked!')}>Click Me</button>
Explanation: Executes alert on button click using React’s synthetic event system.