Other Notes

Web Technology (Practice Set)

Syllabus of MTE

WEB SITE BASICS AND HTML5 Web Essentials: Clients, Servers, and Communication-The Internet – Basic Internet protocols – WWW – HTTP Request Message –HTTP Response Message – Web Clients-Web Servers-Web development strategies-

1. **Clients and Servers:**
   - **Clients:** These are devices like computers, smartphones, or tablets that request and consume resources (like web pages) from servers.
   - **Servers:** These are computers or systems that store and deliver resources in response to client requests.
2. **Basic Internet Protocols:**
   - **TCP/IP:** Transmission Control Protocol/Internet Protocol is a set of protocols that govern communication on the internet, ensuring data is reliably transmitted between devices.
   - **HTTP/HTTPS:** Hypertext Transfer Protocol (HTTP) is used for transmitting web pages, while HTTPS is the secure version that encrypts data for secure communication.
3. **World Wide Web (WWW):**
   - The WWW is a system of interlinked hypertext documents accessed via the internet. It's built on top of the internet and uses protocols like HTTP for communication.
4. **HTTP Request and Response Messages:**
   - **HTTP Request:** Sent by clients to request resources from servers. It contains information like the request method (GET, POST, etc.), headers, and optionally, a message body.
   - **HTTP Response:** Sent by servers in response to client requests. It includes status codes (e.g., 200 for OK, 404 for Not Found), headers, and the response body (e.g., HTML content).
5. **Web Clients and Web Servers:**
   - **Web Clients:** These are software applications (like web browsers) that access and display web content retrieved from servers.
   - **Web Servers:** These are software or hardware systems that store and serve web content to clients upon request.
6. **Web Development Strategies:**
   - **Frontend Development:** Focuses on the user interface and client-side functionality using technologies like HTML, CSS, and JavaScript.
   - **Backend Development:** Involves server-side programming and database management to handle requests, process data, and generate dynamic content.

Introduction to HTML5: Basic Elements, Form Elements, Media Elements, HTML5 Graphics (Canvas, SVG)-XHTML: Syntax and Semantics-Case Study: Create a static Website. FRONT END DESIGN USING CSS3 AND BOOTSTRAP FRAMEWORK CSS : Types of CSS, CSS Properties -CSS3: Selector String, Box Model, Text Properties, CSS 3D Transformation, CSS Animation- Bootstrap Framework: BS Grid, Tables, Images, Alerts, Form Elements. Representing Web Data: Basic XML- DTD- Namespaces-XML Schema, DOM, XSL and XSLT Transformation- Case study: Online Blog Creation

1. **Introduction to HTML5:**
   - **Basic Elements:** HTML5 introduces new semantic elements like **<header>**, **<nav>**, **<section>**, **<article>**, **<footer>**, etc., for better document structure.
   - **Form Elements:** HTML5 includes new form input types (`
- **Media Elements:** HTML5 provides **<audio>** and **<video>** elements for embedding media content directly into web pages.

- **HTML5 Graphics:** HTML5 offers two main graphics technologies: Canvas for dynamic, scriptable rendering of 2D shapes and graphics, and SVG (Scalable Vector Graphics) for defining vector-based graphics.

2. **XHTML:**
   - **Syntax and Semantics:** XHTML is a stricter, XML-based version of HTML. It enforces well-formedness and proper nesting of elements, making code more consistent and easier for parsers to understand.
3. **Front End Design using CSS3 and Bootstrap Framework:**
   - **Types of CSS:** CSS can be applied inline, internally within **<style>** tags, or externally through separate CSS files.
   - **CSS Properties:** CSS properties control the appearance and layout of HTML elements, including text styling, box model properties (like margin, padding, border), and advanced features like 3D transformations and animations in CSS3.
   - **Bootstrap Framework:** Bootstrap is a popular CSS framework that provides pre-designed UI components and responsive grid systems for building modern, mobile-friendly websites. It includes components like grids, tables, images, alerts, and form elements for easy customization and styling.
4. **Representing Web Data using XML:**
   - **Basic XML:** XML (eXtensible Markup Language) is a markup language used to structure and store data in a hierarchical format.
   - **DTD:** Document Type Definition (DTD) defines the structure and content of an XML document.
   - **Namespaces:** Namespaces in XML allow multiple XML vocabularies to be used within the same document without conflict.
   - **XML Schema:** XML Schema defines the structure, data types, and constraints of XML documents, providing a more robust validation mechanism.
   - **DOM:** Document Object Model (DOM) is a programming interface for XML and HTML documents, allowing dynamic access and manipulation of document content.
   - **XSL and XSLT Transformation:** XSL (eXtensible Stylesheet Language) and XSLT (XSL Transformations) are used for transforming XML documents into different formats, such as HTML or other XML structures, based on specified stylesheets.
5. **Case Study: Online Blog Creation:**
   - A case study involving the creation of an online blog would encompass various aspects of web development, including designing the user interface using HTML5 and CSS3, implementing dynamic features with JavaScript, integrating server-side functionality with technologies like PHP or Node.js, and managing data storage and retrieval using databases like MySQL or MongoDB.

DYNAMIC WEB PAGE DESIGN USING JAVA SCRIPT AND JQUERY Java Script: Data Types and Variables -Operators - Control Statements - Functions -Objects - Build in Objects - DOM - Java Script Event Handling - Form Handling and validations - AJAX & JQuery: IntroductionAjax Client Server Architecture- Ajax Client Server Architecture-XML Http Request Object-Call Back Methods-JQuery Selectors - JQuery Animations - Effects - Event Handling - JQuery DOM Traversing-JSON - JQuery AJAX-

1. **JavaScript:**
   - **Data Types and Variables:** JavaScript supports various data types like numbers, strings, booleans, arrays, objects, and more. Variables are used to store and manipulate data.
   - **Operators:** JavaScript includes arithmetic, comparison, logical, and assignment operators for performing operations on data.
   - **Control Statements:** Conditional statements (**if**, **else**, **switch**) and loops (**for**, **while**, **do-while**) control the flow of execution in JavaScript.
   - **Functions:** Functions in JavaScript are reusable blocks of code that can be called to perform specific tasks.
   - **Objects:** JavaScript is an object-oriented language where objects encapsulate data and behavior.
   - **Built-in Objects:** JavaScript provides built-in objects like **Math**, **Date**, **Array**, **String**, etc., for performing common operations.

- **DOM (Document Object Model):** The DOM represents the structure of an HTML document as a tree of objects, allowing JavaScript to interact with and manipulate HTML elements dynamically.
- **JavaScript Event Handling:** Events like clicks, keypresses, form submissions, etc., can be handled using event listeners in JavaScript.

2. **Form Handling and Validations:**
   - JavaScript is commonly used to handle form submissions, validate user inputs, and provide feedback to users based on validation rules.
3. **AJAX & jQuery:**
   - **Introduction to AJAX (Asynchronous JavaScript and XML):** AJAX is a technique for creating dynamic web pages by asynchronously sending and receiving data from a server without reloading the entire page.
   - **Ajax Client-Server Architecture:** AJAX involves the client (browser) making asynchronous requests to the server, which processes the request and sends back data (usually in JSON or XML format).
   - **XMLHttpRequest Object:** In JavaScript, the **XMLHttpRequest** object is used to make AJAX requests to the server.
   - **Callback Methods:** AJAX uses callback functions to handle responses from the server, ensuring that operations continue asynchronously.
   - **jQuery Selectors:** jQuery simplifies DOM manipulation and event handling with selectors that target HTML elements based on CSS selectors.
   - **jQuery Animations and Effects:** jQuery provides methods for creating animations, transitions, and effects on HTML elements.
   - **jQuery Event Handling:** Event handling in jQuery allows for easy attachment of event listeners to DOM elements.
   - **jQuery DOM Traversing:** jQuery offers methods for traversing and manipulating the DOM tree.
   - **JSON (JavaScript Object Notation):** JSON is a lightweight data interchange format often used in AJAX to send and receive structured data.
   - **jQuery AJAX:** jQuery provides convenient methods for making AJAX requests and handling responses, simplifying the process compared to raw JavaScript AJAX calls.

JavaScript Function Example:

javascript Copy code

```javascript
// Function to calculate the area of a rectangle
function calculateArea(length, width) {
    return length * width;
}

// Usage example
let rectangleLength = 5;
let rectangleWidth = 10;
let area = calculateArea(rectangleLength, rectangleWidth);
console.log("Area of the rectangle:", area);
```

2. Form Validation using JavaScript:

html Copy code

```html
<form id="myForm" onsubmit="return validateForm()">
    <input type="text" id="username" placeholder="Enter username">
    <input type="password" id="password" placeholder="Enter passwo
    <input type="submit" value="Submit">
</form>

<script>
    function validateForm() {
        let username = document.getElementById('username').value;
        let password = document.getElementById('password').value;

        if (username === '' || password === '') {
            alert('Please fill in all fields.');
            return false;
        }

        // Additional validation logic can be added here

        return true; // Form submission allowed
    }
</script>
```

3. AJAX Request using jQuery:

javascript Copy code

```javascript
// AJAX GET request using jQuery
$.ajax({
    url: 'https://jsonplaceholder.typicode.com/posts/1',
    method: 'GET',
    success: function(response) {
        console.log('AJAX response:', response);
        $('#result').html('Title: ' + response.title);
    },
    error: function(xhr, status, error) {
        console.error('AJAX error:', error);
        $('#result').html('Error fetching data.');
    }
});
```

4. jQuery Event Handling Example:

html Copy code

```
<button id="myButton">Click Me</button>
<div id="output"></div>

<script>
    // jQuery event handling
    $(document).ready(function() {
        $('#myButton').click(function() {
            $('#output').html('Button clicked!');
        });
    });
</script>
```

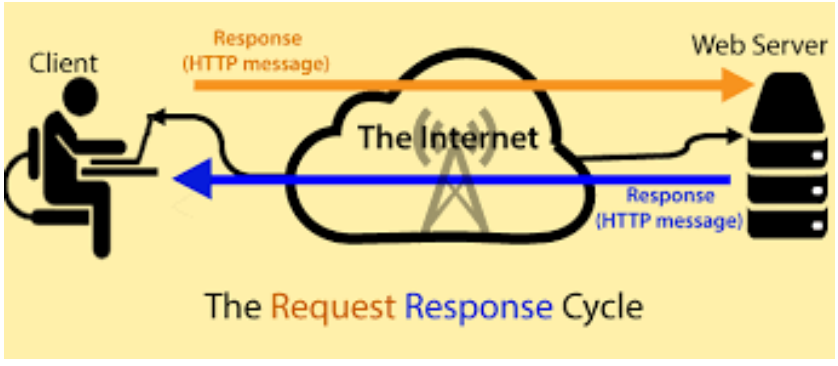
5. JSON Data Example:

javascript Copy code

```
// Sample JSON data
let jsonData = '{"name": "John Doe", "age": 30, "city": "New York"}';

// Parsing JSON string to object
let person = JSON.parse(jsonData);

// Accessing JSON properties
console.log('Name:', person.name);
console.log('Age:', person.age);
console.log('City:', person.city);
```

Practice Questions

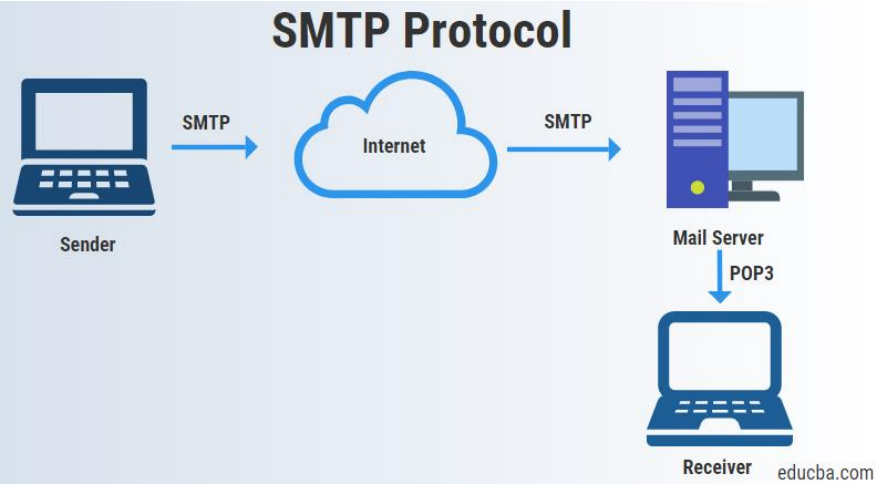
1. Explain the working of Web.



2. Mention the names of organizations which are controlling Internet.

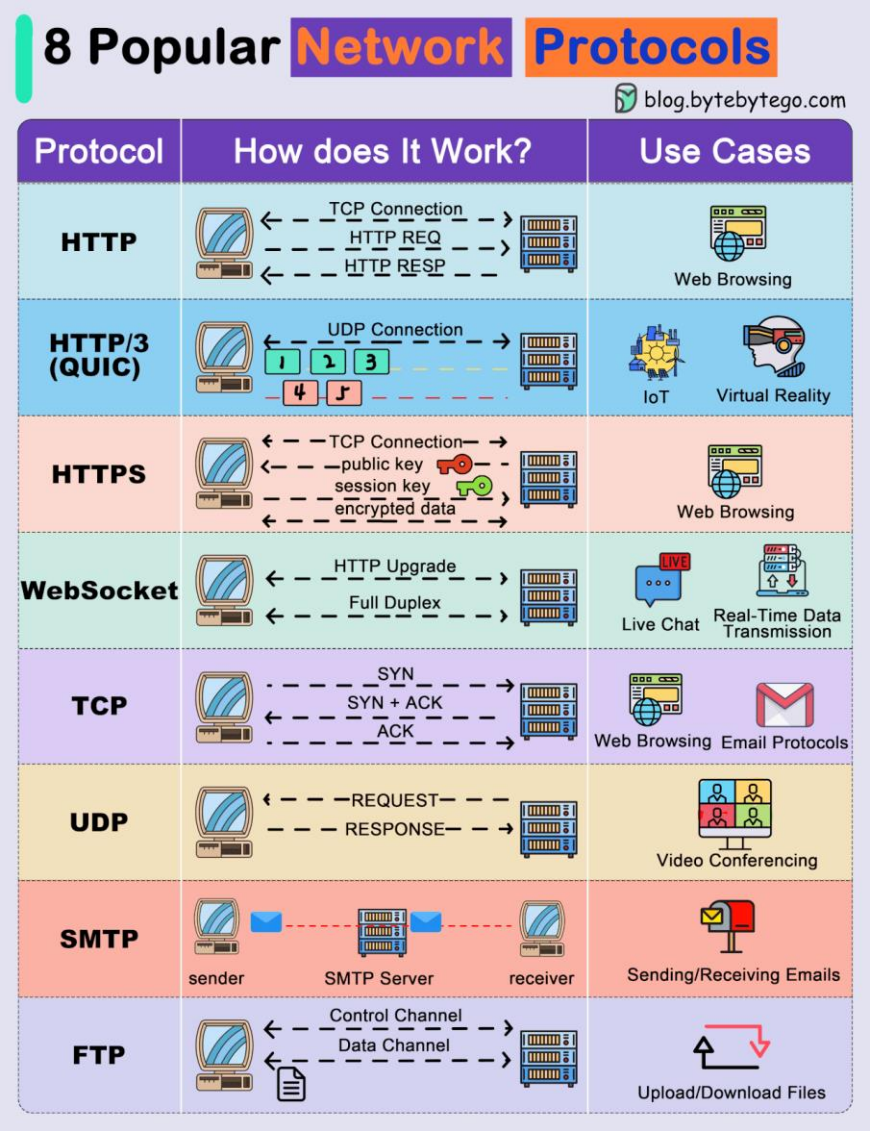
Organization	Role/Responsibility
ICANN (Internet Corporation for Assigned Names and Numbers)	Coordinates DNS, IP address allocation, and TLD management.
IETF (Internet Engineering Task Force)	Develops internet standards and protocols.
ISOC (Internet Society)	Advocates for an open and secure internet.
W3C (World Wide Web Consortium)	Develops and maintains web standards like HTML and CSS.
RIRs (Regional Internet Registries)	Manage IP address allocation within specific regions.
IGF (Internet Governance Forum)	Facilitates multi-stakeholder dialogue on internet governance.
NTIA (National Telecommunications and Information Administration)	Plays a role in internet policy and coordination in the US.
IANA (Internet Assigned Numbers Authority)	Historically managed key internet functions, now under ICANN.

3. Which protocol is used in Mail transfer?



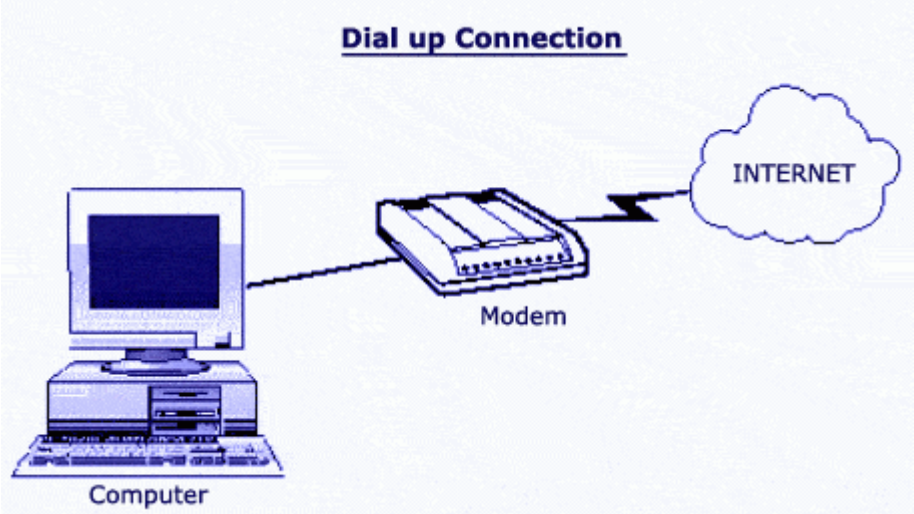
Simple Mail Transfer Protocol (SMTP)

4, What are the protocols used in Internet communication?



4. Give Examples of Internet browsers.

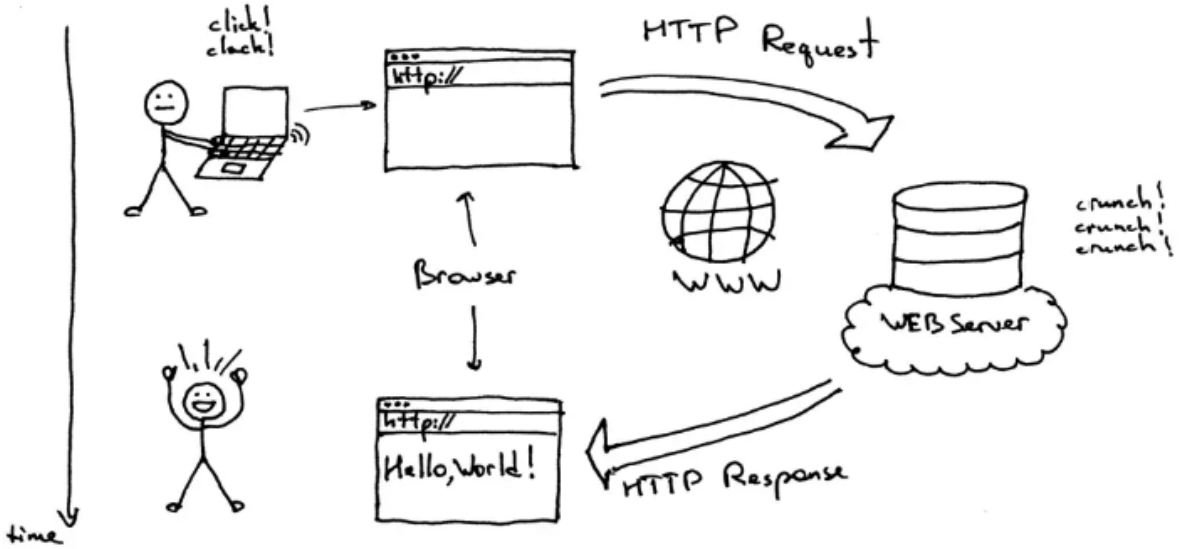
5. Explain Dial-Up Connectivity to start Internet.



Dial-up connectivity was one of the earliest methods used to access the internet before broadband technologies became prevalent. Here's how dial-up connectivity worked to start internet access:

- **Modem Setup:** Connect computer to a modem.
- **Telephone Line Connection:** Use a standard telephone line.
- **Dial-Up Software:** Install ISP-provided software.
- **Initiating Connection:** Enter ISP's phone number and credentials.
- **Handshake and Authentication:** Modems exchange signals; user credentials sent for authentication.
- **Internet Access:** Obtain IP address from ISP for internet access.
- **Data Transfer Speeds:** Typically 28.8 kbps to 56 kbps.
- **Usage Limitations:** Charged based on usage time.

6. How does web browser searches any web resource?



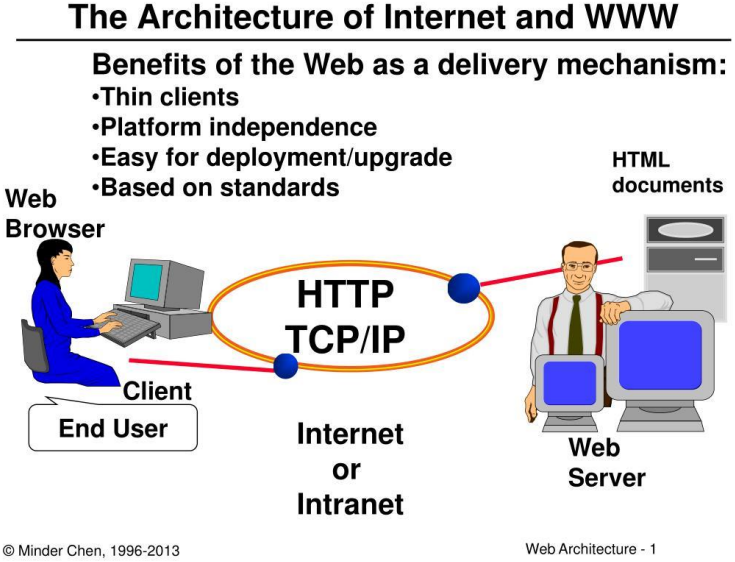
- **Browsers Display Web Content:**
  - Retrieve and show web content.
- **User Action:**
  - Enter URL or click link.
- **DNS Translation:**
  - Convert domain to IP.
- **HTTP Request:**
  - Browser asks server for content.
- **Server Response:**
  - Sends HTML, CSS, JavaScript.
- **Rendering and Styling:**
  - Browser displays and formats.
- **JavaScript Interactivity:**
  - Adds dynamic features.
- **Updates:**
  - Refreshes for changes.

8. What is the meaning of stateless protocol? Mention the name of an stateless protocol.

A stateless protocol is a type of communication protocol where each request from a client to a server is treated as an independent transaction. In other words, the server does not maintain any information or "state" about previous interactions with the client. Each request is processed based solely on the information provided in that specific request.

One example of a stateless protocol is the Hypertext Transfer Protocol (HTTP). In HTTP, each request made by a client (such as a web browser) to a server is handled independently. The server processes the request, generates a response, and sends it back to the client without retaining any information about past requests from the same client. This lack of state retention simplifies server management and scalability but may require additional mechanisms, such as cookies or session tokens, to manage user sessions and maintain continuity across multiple interactions.

9. What is the architecture of Internet? Explain its working.

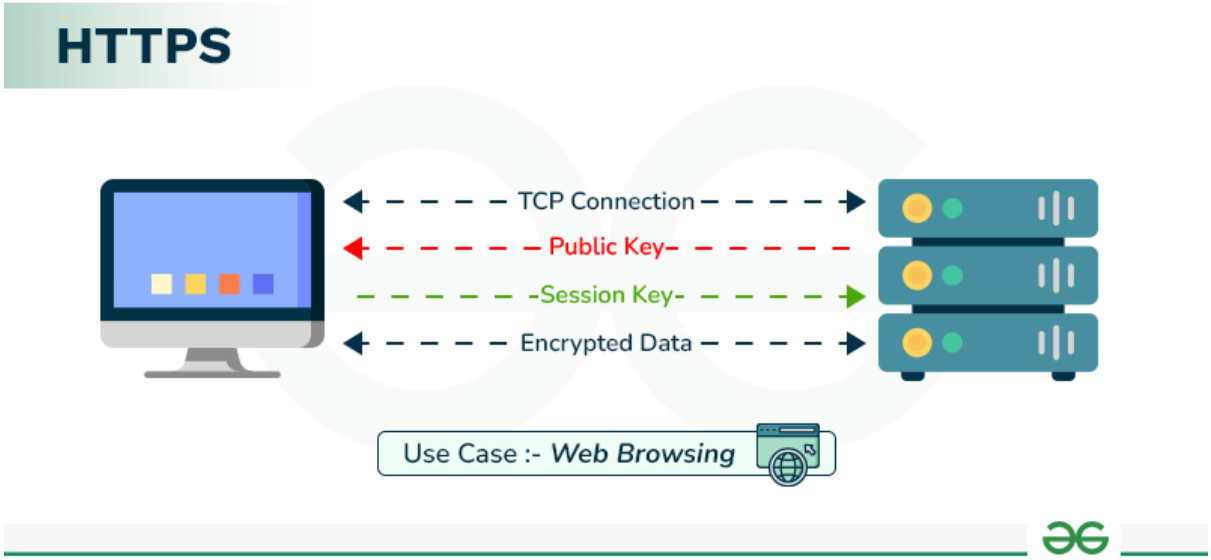


The internet's architecture, based on the TCP/IP model, operates through the following steps:

1. **Data Transmission:** Users send data packets from their devices.
2. **Routing:** Routers direct packets based on destination IP addresses.
3. **Transmission Control:** TCP ensures reliable delivery and error recovery.
4. **Internet Protocol (IP):** Assigns unique IP addresses and routes packets.
5. **Packet Switching:** Data packets switch between routers for efficient delivery.
6. **Packet Reassembly:** TCP reassembles packets at the receiving end.
7. **Response Delivery:** Servers process requests and send responses back through the same process.

This decentralized structure, with standardized protocols, enables global connectivity and data exchange across networks.

10. Explain the use of HTTPS used in the Internet. How is this related to security?



HTTPS, or Hypertext Transfer Protocol Secure, makes internet communication safer in these ways:

1. **Encryption:** It scrambles data, so if someone snoops, they can't understand it.
2. **Data Integrity:** It checks if data was changed during travel to prevent tampering.
3. **Authentication:** It confirms the website's identity, so you know it's real.
4. **Security Indicators:** Browsers show a lock icon or "HTTPS" to signal a safe connection.
5. **Protection:** It guards against hackers trying to steal your info while online.

11. What is the purpose of <head> tag in HTML What are different tags used in <head> tag? Explain with the help of a suitable example.

The `<head>` tag in HTML is used to provide metadata and settings for the webpage. Common tags within `<head>` include:

- `<title>`: Sets the title of the webpage.
- `<meta>`: Provides metadata like character encoding, viewport settings, and keywords.
- `<link>`: Links external resources like stylesheets and favicons.
- `<script>`: Includes scripts or links to external scripts.
- `<base>`: Specifies the base URL for relative URLs.

```html
<!DOCTYPE html>
<html>
<head>
    <title>Page Title</title>
    <meta charset="UTF-8">
    <meta name="description" content="Description of the page">
    <link rel="stylesheet" href="styles.css">
    <script src="script.js"></script>
</head>
<body>
    <!-- Content of the webpage -->
</body>
</html>
```

12. What are different ways to add CSS in a web page? Write a program to add CSS using all the possible ways.

There are several ways to add CSS styles to a web page:

1. **Inline CSS:** Apply styles directly to HTML elements using the `style` attribute.
2. **Internal CSS:** Define styles within the `<style>` tag in the `<head>` section of the HTML document.
3. **External CSS:** Link an external CSS file to the HTML document using the `<link>` tag.

```html
<!DOCTYPE html>
<html>
<head>
    <!-- Internal CSS -->
    <style>
        h1 {
            color: red;
        }
    </style>

    <!-- External CSS -->
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <!-- Inline CSS -->
    <h1 style="text-align: center;">Heading with Inline CSS</h1>
</body>
</html>
```

13. How can we add image in a web page? Write the names of four attributes used for image show.

html Copy code

```html
<img src="example.jpg" alt="Example Image" width="300" height="200">
```

In this example:

- `src`: Specifies the image file's URL.
- `alt`: Provides alternative text for the image.
- `width`: Sets the image's width.
- `height`: Sets the image's height.

14, How do we move text in the direction bottom to up in a web page? Explain with a suitable example in which text direction will be bottom to top and image from up to down.

```html
<!DOCTYPE html>
<html>
<head>
    <style>
        /* Define animation for text moving from bottom to top */
        @keyframes moveUp {
            from { transform: translateY(100%); }
            to { transform: translateY(0); }
        }

        /* Apply animation to the text */
        .moving-text {
            animation: moveUp 2s ease-in-out forwards;
        }
    </style>
</head>
<body>
    <div class="moving-text">Text moving from bottom to top</div>
</body>
</html>
```

15. What is the use of box properties in a web page?

- **Width and Height:** Define the size of an element.
- **Padding:** Adds space inside the element.
- **Margin:** Adds space outside the element.
- **Border:** Defines the border around an element.
- **Box Model:** Determines how total width and height are calculated.
- **Display:** Specifies how an element is displayed.
- **Positioning:** Controls the positioning of elements.
- **Overflow:** Specifies behavior when content overflows the element's box.

16. Write down the names of technologies used in client side and server side programming.

Client-Side Programming	Server-Side Programming
HTML	Node.js
CSS	PHP
JavaScript	Python (Django, Flask)
TypeScript	Ruby (Ruby on Rails)

Client-Side Programming	Server-Side Programming
XML	Java (Spring Boot)
JSON	C# (ASP.NET)
AJAX	Express.js
React	Ruby on Rails
Angular	ASP.NET Core
Vue.js	Flask
Bootstrap	Django
Sass	Laravel (PHP)
	MySQL (Database)
	PostgreSQL (Database)
	MongoDB (NoSQL Database)

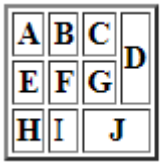
17. Find out the measures differences between HTML and XHTML?

Feature	HTML	XHTML
Based on	SGML	XML
Case-sensitive	No	Yes
Syntax	Loose	Strict
End tags and attributes	Optional for some elements	Required for all elements
Empty elements	Do not require a closing tag	Require a closing tag
Extensibility	Less extensible	More extensible
Suitability for use with other data formats	Less suitable	More suitable
Browser support	Widely supported	Less widely supported than HTML, but still supported by all major browsers

18. Draw the following figure using <table > tag

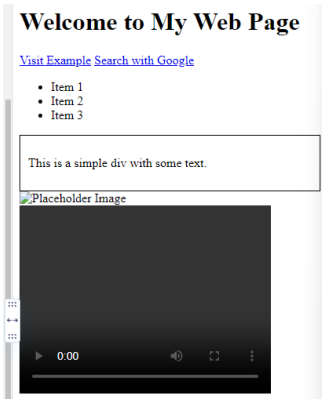
<table>
<tr><td>A</td><td>B</td><td>C</td><td rowspan="2">D</td></tr>
<tr><td>E</td><td>F</td><td>G</td></tr>
<tr><td>H</td><td>I</td><td colspan="2">J</td></tr>
</table>

```html
<table border='2'>
    <tr>
        <td>A</th>
        <td>B</th>
        <td>C</th>
        <td rowspan="2">D</th>
    </tr>
    <tr>
        <td>E</td>
        <td>F</td>
        <td>G</td>
    </tr>
    <tr>
        <td>H</th>
        <td>I</th>
        <td colspan='2'>J</th>
    </tr>
</table>
```



19. Create a web page using links, list, div, images and videos.

```html
<body>
    <h1>Welcome to My Web Page</h1>

    <!-- Links -->
    <a href="https://example.com">Visit Example</a>
    <a href="https://google.com">Search with Google</a>

    <!-- List -->
    <ul>
        <li>Item 1</li>
        <li>Item 2</li>
        <li>Item 3</li>
    </ul>

    <!-- Div -->
    <div style="border: 1px solid black; padding: 10px;">
        <p>This is a simple div with some text.</p>
    </div>

    <!-- Image -->
    <img src="https://via.placeholder.com/150" alt="Placeholder Image">

    <!-- Video -->
    <video width="320" height="240" controls>
        <source src="https://www.example.com/video.mp4" type="video/mp4
        Your browser does not support the video tag.
    </video>
</body>
</html>
```



20. Write an XML file which will display the Book information which includes the following: Title of the book, Author Name, ISBN number, Publisher name, Edition and Price. Validate the above document using DTD and XML Schema.

Here's an XML file representing book information:

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<library>
  <book>
    <title>The Lord of the Rings</title>
    <author>J.R.R. Tolkien</author>
    <isbn>978-0261102694</isbn>
    <publisher>Houghton Mifflin Harcourt</publisher>
    <edition>50th Anniversary</edition>
    <price>27.99</price>
  </book>
  </library>
```

Use code with caution.

This file defines a root element `<library>` which contains multiple `<book>` elements. Each book element holds details like title, author, ISBN, publisher, edition, and price.

2. Validation with DTD

book.dtd:

XML

```xml
<!ELEMENT library (book+)>
<!ELEMENT book (title, author, isbn, publisher, edition, price)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT isbn (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT edition (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

3. Validation with XML Schema

book.xsd:

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="library">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="book" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" type="xs:string"/>
        <xs:element name="author" type="xs:string"/>
        <xs:element name="isbn" type="xs:string"/>
        <xs:element name="publisher" type="xs:string"/>
        <xs:element name="edition" type="xs:string"/>
        <xs:element name="price" type="xs:decimal"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

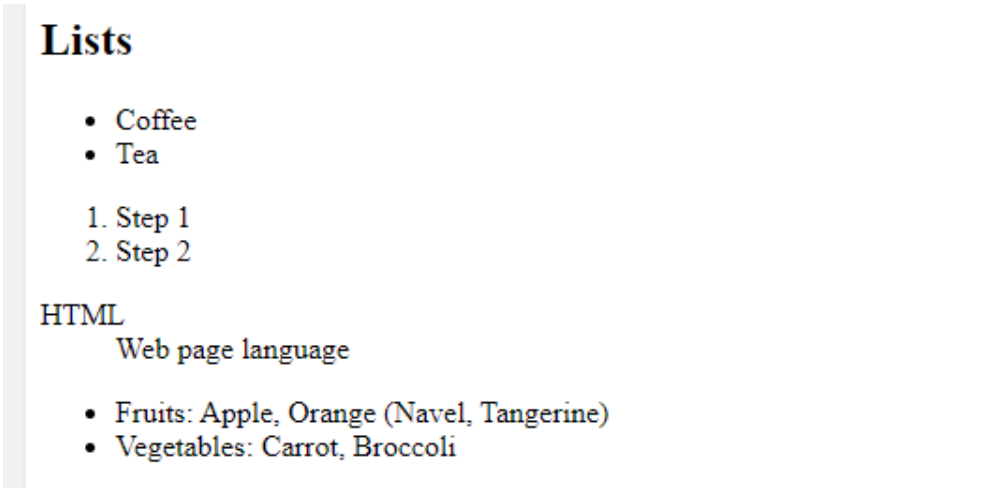
21. Create two links using HTML in which first link will connect to another page the second linl to heading in the same page.

HTML

```html
<p>
  <a href="otherpage.html">Click here to go to another page</a><br>

  <a href="#second_heading">Jump to the second heading</a>
</p>

<h2 id="second_heading">This is the second heading</h2>
```

22. Demonstrate the following using HTML codes.
    a. Unordered List
    b. Ordered List
    c. Definition List
    d. Nested List

```html
<!DOCTYPE html>
<html lang="en">
<body>
  <h2>Lists</h2>
  <ul>
    <li>Coffee</li>
    <li>Tea</li>
  </ul>
  <ol>
    <li>Step 1</li>
    <li>Step 2</li>
  </ol>
  <dl>
    <dt>HTML</dt>
    <dd>Web page language</dd>
  </dl>
  <ul>
    <li>Fruits: Apple, Orange (Navel, Tangerine)</li>
    <li>Vegetables: Carrot, Broccoli</li>
  </ul>
</body>
</html>
```



Lists

- Coffee
- Tea

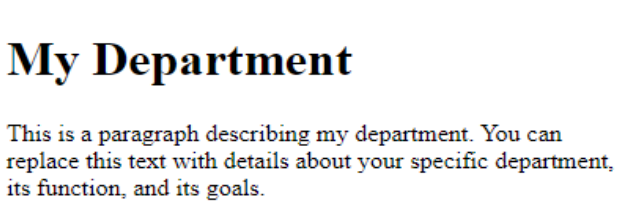
1. Step 1
2. Step 2

HTML
: Web page language

- Fruits: Apple, Orange (Navel, Tangerine)
- Vegetables: Carrot, Broccoli

23. Create a webpage using HTML to describe your department using paragraph and use innerHTML,

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device
      -width, initial-scale=1.0">
  <title>My Department</title>
</head>
<body>
  <h1>My Department</h1>
  <p id="departmentDescription"></p>

  <script>
    const departmentDescription = document
        .getElementById("departmentDescription"
        );
    departmentDescription.innerHTML = "This is
        a paragraph describing my department.
        You can replace this text with details
        about your specific department, its
        function, and its goals.";
  </script>
</body>
</html>
```



24. Explain the structure of XML. What is the use of DTD in XML?

XML Structure (Super Short):

- **Building blocks:** Elements (`<tag>content</tag>`), define data and hierarchy.
- **Attributes:** Extra info for elements (`<tag name="value">`).
- **Nesting:** Elements can hold other elements for complex data.
- **Well-formed:** Follows basic syntax rules (closed tags, nesting).
- **Valid:** Conforms to a defined structure (DTD or XML Schema).

DTD (in a nutshell):

- Defines the legal structure of an XML document.
- Like a blueprint, specifying allowed elements, attributes, and nesting.
- Benefits: Validation (data consistency), Documentation (clear structure).
- Limitations: Basic data types, less readable syntax.
- Alternative: XML Schema (more powerful, user-friendly).

25. Write an XML document to store the details of Employees. Also design a web page to get the input of Employee id and the full details from the XML document.

1. XML document for Employee details (emp_data.xml)

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<employees>
  <employee id="1001">
    <name>
      <first_name>John</first_name>
      <last_name>Doe</last_name>
    </name>
    <department>IT</department>
    <position>Software Engineer</position>
    <salary>80000</salary>
  </employee>
  </employees>
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Employee Search</title>
  <script>
    function searchEmployee() {
      const id = document.getElementById("empId").value;
      fetch("emp_data.xml")
        .then(response => response.text())
        .then(data => {
          const parser = new DOMParser();
          const xmlDoc = parser.parseFromString(data, "text/xml");
          const employee = xmlDoc.querySelector(`employee[id="${id}"]`);
          if (employee) {
            const name = employee.querySelector("name").textContent.split(" ");
            document.getElementById("result").innerHTML = `
              <p>Employee ID: ${id}</p>
              <p>Name: ${name[0]} ${name[1]}</p>
              <p>Department: ${employee.querySelector("department").textContent}</p
              <p>Position: ${employee.querySelector("position").textContent}</p>
              <p>Salary: ${employee.querySelector("salary").textContent}</p>
            `;
          } else {
            document.getElementById("result").textContent = "Employee not found!";
          }
        })
        .catch(error => console.error(error));
    }
  </script>
</head>
<body>
  <h1>Employee Search</h1>
  <label for="empId">Employee ID:</label>
  <input type="text" id="empId" name="empId">
  <button onclick="searchEmployee()">Search</button>
  <br>
  <div id="result"></div>
</body>
</html>
```

27. Create a sample code to illustrate types of Style sheets for your web page. Explain with an example.

Ans 12

style.css (External Style Sheet):

CSS

```css
body {
  font-family: Arial, sans-serif;
  background-color: #f5f5f5;
}

p {
  color: #333;
  line-height: 1.5;
}
```

29. Write an XML for person information and access the data using XSL. How do we write XML schema?

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<people>
  <person id="1">
    <name>John Doe</name>
    <email>john.doe@example.com</email>
    <age>30</age>
  </person>
  <person id="2">
    <name>Jane Smith</name>
    <email>jane.smith@example.com</email>
    <age>25</age>
  </person>
</people>
```

2. Accessing Data with XSL:

people.xsl (XSL Stylesheet):

XML

```xml
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <h2>People List</h2>
    <ul>
      <xsl:for-each select="people/person">
        <li>
          <xsl:value-of select="name"/> - <xsl:value-of select="email"/> (Age: <xsl
        </li>
      </xsl:for-each>
    </ul>
  </xsl:template>
</xsl:stylesheet>
```

3. Writing XML Schema:

person.xsd (XML Schema):

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="people">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="person" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="person">
    <xs:complexType>
      <xs:attribute name="id" type="xs:string" use="required"/>
      <xs:sequence>
        <xs:element name="name" type="xs:string"/>
        <xs:element name="email" type="xs:string"/>
        <xs:element name="age" type="xs:integer"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

30. Create a web page in which use <DIV> tag, CSS classes and CSS box properties.

```html
<!DOCTYPE html>
<html>
<head>
<title>Boxes</title>
<style>
  .box {
    width: 150px;
    padding: 10px;
    margin: 10px;
    border: 1px solid #ddd;
    background: #eee;
  }
</style>
</head>
<body>
  <div class="box">Box 1</div>
  <div class="box">Box 2</div>
  <div class="box">Box 3</div>
</body>
</html>
```

31. What are different popups in JavaScript? Explain the use of all with the help of suitable example.

JavaScript Popups (Short & Simple)

Popup Type	Use Case	Example	Return Value
Alert	Simple notification	`alert("Message here!")`	None
Confirm	Confirmation prompt	`const answer = confirm("Delete item?");`	`true` for OK, `false` for Cancel
Prompt	User input (text)	`const name = prompt("Enter your name:");`	Entered text (string) or `null` for Cancel

32. How can we print the content of web page visible on the window screen?

2. Using JavaScript (Limited Use Cases):

While not ideal for all scenarios, JavaScript can be used to trigger the browser's print functionality. However, it requires user interaction and might not work consistently across all browsers.

JavaScript

```javascript
function printVisibleContent() {
  window.print();
}
```

33. Develop a web page and add JavaScript using all the possible ways.

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>JavaScript Demo</title>
  <style>
    /* Increase width of the body element or container element to fit content,
       if needed, adjust margins/paddings for better layout */
    body {
      width: 100%; /* Example: Set body width to 100% */
    }
  </style>
  <script>
    alert("Inline Script!");
  </script>
</head>
<body>
  <h1>JavaScript Demo</h1>
  <button onclick="changeColor()">Change Color (Inline)</button>
  <p id="message">This is a message.</p>
  <script>
    function changeColor() {
      document.getElementById("message").style.color = "red";
    }
  </script>
  <script src="script.js"></script>
</body>
</html>
```

script.js (External Script):

JavaScript

```
alert("External Script!")  // Call directly (optional)
```

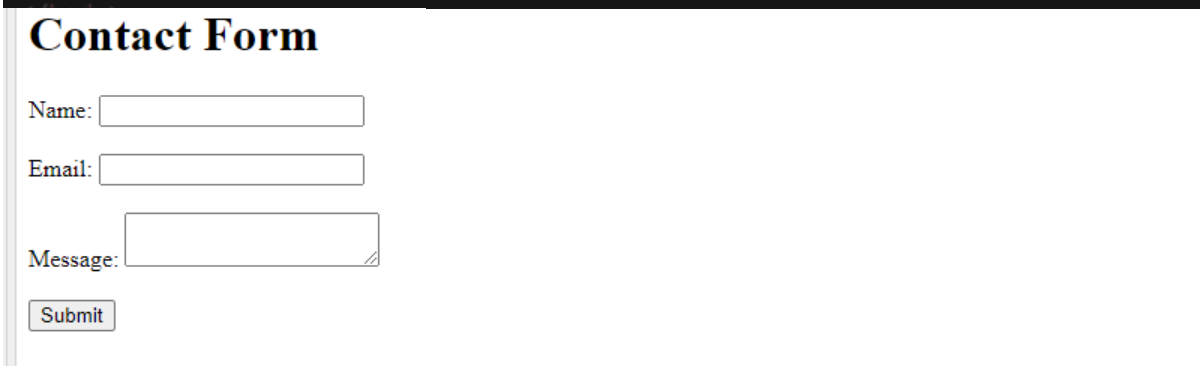
34. Write a JavaScript code to validate an email id using JavaScript. There should be one @ and a dot after the @ symbol.

JavaScript

```
function validateEmail(email) {
  const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  return regex.test(email);
}

// Example usage
const email1 = "johndoe@example.com";
const email2 = "invalidEmail";
const email3 = "john.doe@"; // Missing domain after dot

console.log(validateEmail(email1)); // true
console.log(validateEmail(email2)); // false (no @ symbol)
console.log(validateEmail(email3)); // false (missing domain after dot)
```

35. Develop an HTML form and stop user in submitting blank form using JavaScript.

```html
<body>
  <h1>Contact Form</h1>
  <form id="myForm">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required><br><br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>
    <label for="message">Message:</label>
    <textarea id="message" name="message" required></textarea><br><br>
    <button type="submit">Submit</button>
  </form>

  <script>
    const form = document.getElementById("myForm");

    form.addEventListener("submit", function(event) {
      // Prevent default form submission
      event.preventDefault();

      // Get form elements
      const name = document.getElementById("name").value;
      const email = document.getElementById("email").value;
      const message = document.getElementById("message").value;

      // Check if any field is empty
      if (name === "" || email === "" || message === "") {
        alert("Please fill out all fields!");
        return false; // Explicitly return false to prevent submission
      }

      // Additional validation can be added here (e.g., email format check)

      // If all fields are filled, submit the form using your preferred method (o
      // You can use techniques like AJAX or redirect to a processing page
      console.log("Form submitted successfully!"); // Example (replace with your
    });
  </script>
```



36. Construct a Dropdown list using HTML Can we select more than one value at a time?

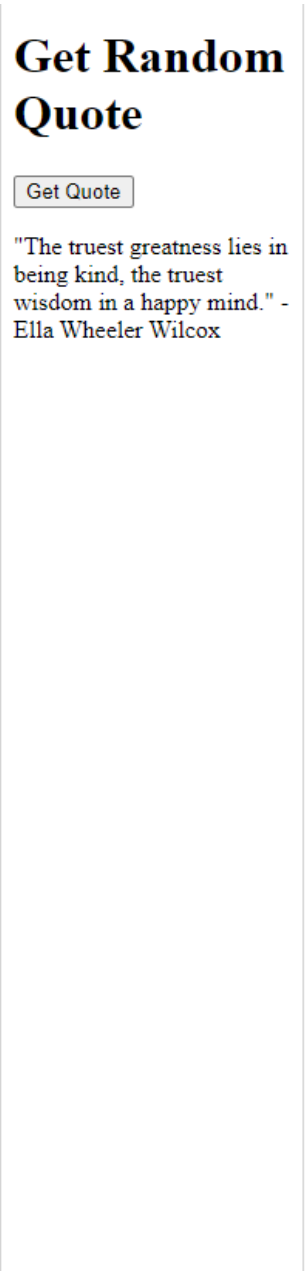
```
<body>
    <h1>Select Colors</h1>
    <select id="colors" multiple>
    <option value="red">Red</option>
    <option value="green">Green</option>
    <option value="blue">Blue</option>
    <option value="yellow">Yellow</option>
  </select>
```

37. Write a java script program which shows history and other DOM objects.

```
// Display the browser's history
console.log('Browser History:');
console.log(window.history);

// Display the document object
console.log('Document Object:');
console.log(document);

// Display the window object
console.log('Window Object:');
console.log(window);
```

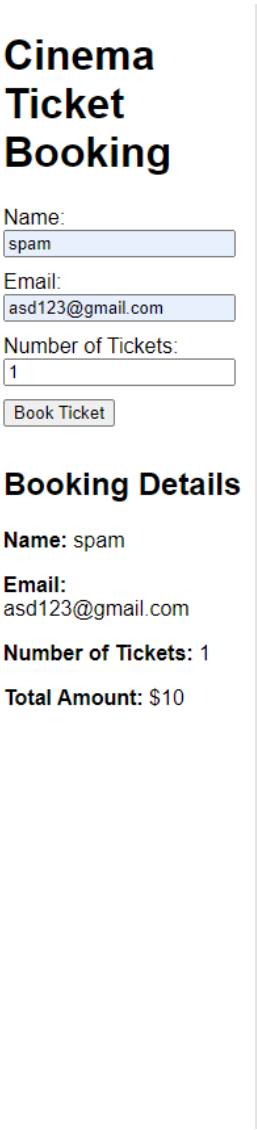
38. Make use of AJAX in web programming with a suitable program.

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>AJAX Example</title>
</head>

<body>
    <h1>Get Random Quote</h1>
    <button id="getQuoteButton">Get Quote</button>
    <p id="quoteDisplay"></p>

    <script>
        const getQuoteButton = document.getElementById("getQuoteButton");
        const quoteDisplay = document.getElementById("quoteDisplay");

        getQuoteButton.addEventListener("click", function() {
            // AJAX request using Fetch API
            fetch("https://api.quotable.io/random")
                .then(response => response.json()) // Parse JSON response
                .then(data => {
                    const quoteText = data.content;
                    const quoteAuthor = data.author;
                    quoteDisplay.textContent = `"${quoteText}" -
                        ${quoteAuthor}`;
                })
                .catch(error => {
                    quoteDisplay.textContent = "Error fetching quote: " +
                        error;
                });
        });
    </script>
</body>

</html>
```



39. Construct a system in which book a cinema ticket with the help of HTML, CSS & JavaScript.

```
<style>
  body {
    font-family: Arial, sans-serif;
  }
  input, button {
    margin-bottom: 10px;
  }
</style>
</head>
<body>
  <h1>Cinema Ticket Booking</h1>
  <label for="name">Name:</label>
  <input type="text" id="name" required><br>
  <label for="email">Email:</label>
  <input type="email" id="email" required><br>
  <label for="tickets">Number of Tickets:</label>
  <input type="number" id="tickets" min="1" required><br>
  <button onclick="bookTicket()">Book Ticket</button>
  <div id="booking-details"></div>

  <script>
    function bookTicket() {
      const name = document.getElementById('name').value;
      const email = document.getElementById('email').value;
      const tickets = parseInt(document.getElementById('tickets').value
          );
      const totalAmount = tickets * 10; // Assuming each ticket costs
          $10

      const bookingDetails = document.getElementById('booking-details');
      bookingDetails.innerHTML = `
        <h2>Booking Details</h2>
        <p><strong>Name:</strong> ${name}</p>
        <p><strong>Email:</strong> ${email}</p>
        <p><strong>Number of Tickets:</strong> ${tickets}</p>
        <p><strong>Total Amount:</strong> $${totalAmount}</p>
      `;
    }
  </script>
```



40. Write a program to add two numbers using “Prompt box” in JavaScript.

JavaScript

```
// Get the first number from the user using prompt
const firstNumber = parseFloat(prompt("Enter the first number:"));

// Check if the input is a valid number (not NaN)
if (isNaN(firstNumber)) {
  alert("Please enter a valid number!");
} else {
  // Get the second number from the user using prompt
  const secondNumber = parseFloat(prompt("Enter the second number:"));

  // Check if the second input is also a valid number
  if (isNaN(secondNumber)) {
    alert("Please enter a valid number!");
  } else {
    // Calculate the sum
    const sum = firstNumber + secondNumber;

    // Display the result using alert
    alert("The sum of the two numbers is: " + sum);
  }
}
```

41. Why is JavaScript Object Notation (JSON) important in web application? Explain with the help of a suitable program.

JSON (JavaScript Object Notation) is crucial in web applications for data exchange. Here's why:

1. **Easy Data Interchange:** JSON provides a lightweight, human-readable format for sending data between servers and web apps. This allows different parts of your application or even separate applications to communicate seamlessly.
2. **Language Independence:** Unlike complex formats like XML, JSON is understandable by most programming languages. This makes it ideal for building web APIs (Application Programming Interfaces) that can be accessed by various clients regardless of their language.
3. **Simplicity:** JSON's syntax is similar to JavaScript object literals, making it familiar and easy to work with for web developers.
4. **Efficiency:** JSON is lightweight and requires less processing power compared to XML. This translates to faster data transfer and better performance for web applications.

```
fetch("https://api.example.com/data") // Replace with your API endpoint
  .then(response => response.json())
  .then(data => {
    // Access data properties here (e.g., data.name, data.value)
  })
  .catch(error => {
    console.error("Error fetching data:", error);
  });
```

42 What is the utility of Jquery? Explain with the help of a suitable example.

• **DOM Manipulation:** jQuery offers easier ways to select, modify, and interact with HTML elements compared to raw JavaScript.

• **Simplified AJAX:** jQuery simplifies making asynchronous requests (AJAX) for fetching data without full page reloads.

• **Cross-Browser Compatibility:** jQuery helps ensure code works consistently across different browsers.

Example (Fade Effect):

HTML

```
<button id="fadeButton">Fade</button>
<p id="message">This is a message.</p>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
  $(document).ready(function() {
    $("#fadeButton").click(function() {
      $("#message").fadeToggle(); // Toggles fade in/out
    });
  });
</script>
```

43. Can we reduce the length of JavaScript using JQuery? Explain with a suitable example.

Absolutely! jQuery can significantly reduce JavaScript code length for common DOM manipulation tasks. Here's an example:

Without jQuery (More Code):

JavaScript

```javascript
document.getElementById("myButton").addEventListener("click", function() {
  const element = document.getElementById("myElement");
  if (element.classList.contains("hidden")) {
    element.classList.remove("hidden");
  } else {
    element.classList.add("hidden");
  }
});
```

With jQuery (Less Code):

JavaScript

```javascript
$("#myButton").click(function() {
  $("#myElement").toggle();
});
```

44. Write a program using Java script which gives browsers information and history object.

JavaScript

```javascript
const getInfo = () => {
  const { appName, appVersion, userAgent } = navigator;
  const { width, height } = screen;
  const { title, href } = document.location;
  const historyLength = window.history.length;
  const referrer = document.referrer || "Referrer information not available.";
  console.log("Browser:", appName, appVersion, userAgent);
  console.log("Screen:", width, "x", height);
  console.log("Document:", title, href);
  console.log("History:", historyLength);
  console.log("Referrer:", referrer);
};

getInfo();
```

45. Create a webpage containing 3 overlapping images using HTML, CSS and JavaScript. Further when the mouse is over any image, it should be on the top and fully displayed.

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Overlapping Images</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="image-container">
    <img src="image1.jpg" alt="Image 1" class="image">
    <img src="image2.jpg" alt="Image 2" class="image">
    <img src="image3.jpg" alt="Image 3" class="image">
  </div>

  <script src="script.js"></script>
</body>
</html>
```

style.css:

CSS

```css
.image-container {
  position: relative;
  width: 300px; /* Adjust width as needed */
  height: 300px; /* Adjust height as needed */
}

.image {
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  opacity: 0.7; /* Set initial opacity */
  transition: opacity 0.3s ease-in-out; /* Smooth transition */
}

.image:hover {
  opacity: 1; /* Full opacity on hover */
  z-index: 1; /* Bring hovered image to the top */
}
```

47. Create a web page which divides the page in two equal frames and place the audio and video clips in frame-1 and frame-2 respectively. Execution will be on the basis of user input: (Sad / Happy).

```html
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
  initial-scale=1.0">
  <title>Happy Mood</title>
</head>
<frameset cols="50%,50%">
  <frame src="sad.html" name="sadFrame">
    <frame src="happy.html" name="happyFrame">
</frameset>

</html>
```

sad_audio.html (Audio Player Frame Content):

HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Sad Audio</title>
</head>
<body>
  <audio controls autoplay>
    <source src="sad_audio.mp3" type="audio/mpeg"> Your browser does not support th
  </audio>
</body>
</html>
```

happy_video.html (Video Player Frame Content):

HTML

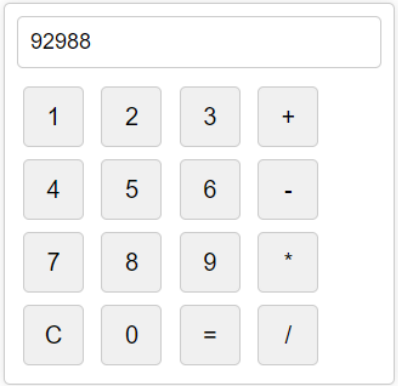
```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Happy Video</title>
</head>
<body>
  <video controls autoplay>
    <source src="happy_video.mp4" type="video/mp4"> Your browser does not support t
  </video>
</body>
</html>
```

48. Create a calculator using HTML, CSS and JavaScript.

```html
<body>
  <div class="calculator">
    <input type="text" id="display" readonly>
    <input type="button" value="1" onclick="appendNumber('1')">
    <input type="button" value="2" onclick="appendNumber('2')">
    <input type="button" value="3" onclick="appendNumber('3')">
    <input type="button" value="+" onclick="appendOperator('+')">
    <input type="button" value="4" onclick="appendNumber('4')">
    <input type="button" value="5" onclick="appendNumber('5')">
    <input type="button" value="6" onclick="appendNumber('6')">
    <input type="button" value="-" onclick="appendOperator('-')">
    <input type="button" value="7" onclick="appendNumber('7')">
    <input type="button" value="8" onclick="appendNumber('8')">
    <input type="button" value="9" onclick="appendNumber('9')">
    <input type="button" value="*" onclick="appendOperator('*')">
    <input type="button" value="C" onclick="clearDisplay()">
    <input type="button" value="0" onclick="appendNumber('0')">
    <input type="button" value="=" onclick="calculate()">
    <input type="button" value="/" onclick="appendOperator('/')">
  </div>
```

```html
  <script>
    function appendNumber(number) {
      document.getElementById('display').value += number;
    }

    function appendOperator(operator) {
      const display = document.getElementById('display');
      const lastChar = display.value.slice(-1);

      if (!isNaN(lastChar) || display.value === '') {
        display.value += operator;
      }
    }

    function clearDisplay() {
      document.getElementById('display').value = '';
    }

    function calculate() {
      const display = document.getElementById('display');
      const result = eval(display.value);
      display.value = result;
    }
  </script>
```

```
<style>
  body {
    font-family: Arial, sans-serif;
  }
  .calculator {
    width: 300px;
    margin: 20px auto;
    border: 1px solid #ccc;
    padding: 10px;
    border-radius: 5px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  }
  .calculator input[type="text"] {
    width: 100%;
    padding: 10px;
    margin-bottom: 10px;
    font-size: 18px;
    border-radius: 5px;
    border: 1px solid #ccc;
    box-sizing: border-box;
  }
  .calculator input[type="button"] {
    width: 50px;
    height: 50px;
    font-size: 20px;
    margin: 5px;
    border-radius: 5px;
    border: 1px solid #ccc;
    cursor: pointer;
  }
</style>
```



49. Write a JavaScript program to input name and address of a visitor and display a greeting message.

```html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Greeting Message</title>
</head>
<body>
  <script>
    // Prompt the user to input name and address
    const name = prompt('Enter your name:');
    const address = prompt('Enter your address:');

    // Construct the greeting message
    let greeting = `Hello, ${name}!`;
    if (address) {
      greeting += ` Your address is ${address}.`;
    } else {
      greeting += ` Your address is not provided.`;
    }

    // Display the greeting message
    alert(greeting);
  </script>
</body>
</html>
```

50. Can we copy content of an array into the other array? Write a program in this support.

Yes, you can copy the content of one array into another array in JavaScript. Here's an example program to demonstrate this:

javascript Copy code

```javascript
// Original array
const originalArray = [1, 2, 3, 4, 5];

// Create a new array and copy the content of the original array into
const newArray = [...originalArray];

// Display the original and new arrays
console.log('Original Array:', originalArray);
console.log('New Array:', newArray);
```

We use the spread syntax (`...`) to create a new array `newArray` and copy the content of `originalArray` into it.

51. Create an enquiry form using Java script and Why must one use JSON over XML? Explain with the help of a suitable example.

```
<body>
  <h1>Enquiry Form</h1>
  <form id="enquiryForm">
    <label for="name">Name:</label>
    <input type="text" id="name" required><br><br>
    <label for="email">Email:</label>
    <input type="email" id="email" required><br><br>
    <label for="message">Message:</label><br>
    <textarea id="message" rows="4" cols="50" required></textarea><br
    <button type="submit">Submit</button>
  </form>
```

JSON (JavaScript Object Notation) and XML (eXtensible Markup Language) are both used for structuring and exchanging data. JSON has become more popular than XML in many web applications due to its simplicity, readability, and ease of use with JavaScript.

One of the key reasons to use JSON over XML is its compactness and simplicity. JSON data is represented as key-value pairs, making it lightweight and easier to parse compared to XML. Here's an example to illustrate this:

JSON Example:

json Copy code

```
{
  "name": "John Doe",
  "age": 30,
  "email": "johndoe@example.com",
  "address": {
    "city": "New York",
    "country": "USA"
  }
}
```

XML Equivalent:

xml Copy code

```xml
<person>
  <name>John Doe</name>
  <age>30</age>
  <email>johndoe@example.com</email>
  <address>
    <city>New York</city>
    <country>USA</country>
  </address>
</person>
```

In this example, the JSON representation is more concise and easier to read compared to the XML representation. JSON is also more natural to work with in JavaScript, making it a preferred choice for data interchange in web development.

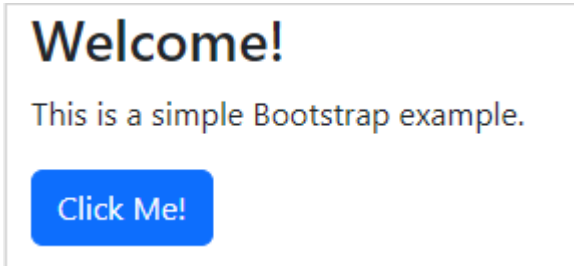
53. Explain the importance of Boot Strap. Write a program to show the utility of Boot Strap. Book a railway ticket using web pages. One page for user registration, one for source and destination selection and the third page will be for different ways of payment.

Importance of Bootstrap:

Bootstrap is a free and open-source CSS framework that provides a collection of pre-designed components and utilities for building responsive websites and web applications. Here's why it's important:

- **Simplified Development:** Bootstrap offers pre-built components and utility classes for common UI elements like buttons, forms, grids, navigation bars, modals, etc. This reduces the amount of custom CSS code you need to write, saving time and effort.
- **Responsiveness:** Bootstrap components are designed to adapt automatically to different screen sizes and devices, ensuring your website looks good on desktops, tablets, and smartphones.
- **Cross-browser Compatibility:** Bootstrap ensures your website displays consistently across different web browsers, minimizing compatibility issues.
- **Rapid Prototyping:** Bootstrap's components and utilities can be used to quickly create basic layouts and mockups, facilitating faster development iteration.
- **Large Community & Ecosystem:** Bootstrap benefits from a large and active community of developers who contribute extensions, themes, and ongoing maintenance.

```
  <title>Bootstrap Example</title>
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/cs
</head>
<body>
  <div class="container">
    <div class="row">
      <div class="col-md-6">
        <h1>Welcome!</h1>
        <p>This is a simple Bootstrap example.</p>
      </div>
      <div class="col-md-6">
        <button type="button" class="btn btn-primary">Click Me!</button>
      </div>
    </div>
  </div>

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundl
</body>
</html>
```



54. To create a web page which includes a map and display the related information when a hot spot is clicked in the map.

55. Develop and interpret a XHTML file that includes Javascript script for the following problems:
a) Input: A number n obtained using prompt box
Output: The first n Prime numbers
b) Input: A number n obtained using prompt
Output: A table of numbers from 1 to n and their squares using alert

```
// Function to handle user input and display results
function handleInput() {
  const num = parseInt(prompt('Enter a number:'));
  if (!isNaN(num) && num > 0) {
    const primes = generatePrimes(num);
    alert(`The first ${num} prime numbers are:\n${primes.join(',

    const table = generateTable(num);
    alert(table);
  } else {
    alert('Invalid input. Please enter a positive number.');
  }
}
</script>
```

```
function isPrime(num) {
  if (num <= 1) return false;
  for (let i = 2; i <= Math.sqrt(num); i++) {
    if (num % i === 0) return false;
  }
  return true;
}

// Function to generate the first n prime numbers
function generatePrimes(n) {
  const primes = [];
  let num = 2;
  while (primes.length < n) {
    if (isPrime(num)) primes.push(num);
    num++;
  }
  return primes;
}
```

```
// Function to generate a table of numbers and their squares
function generateTable(n) {
  let table = '<table border="1">';
  table += '<tr><th>Number</th><th>Square</th></tr>';
  for (let i = 1; i <= n; i++) {
    table += `<tr><td>${i}</td><td>${i * i}</td></tr>`;
  }
  table += '</table>';
  return table;
}
```

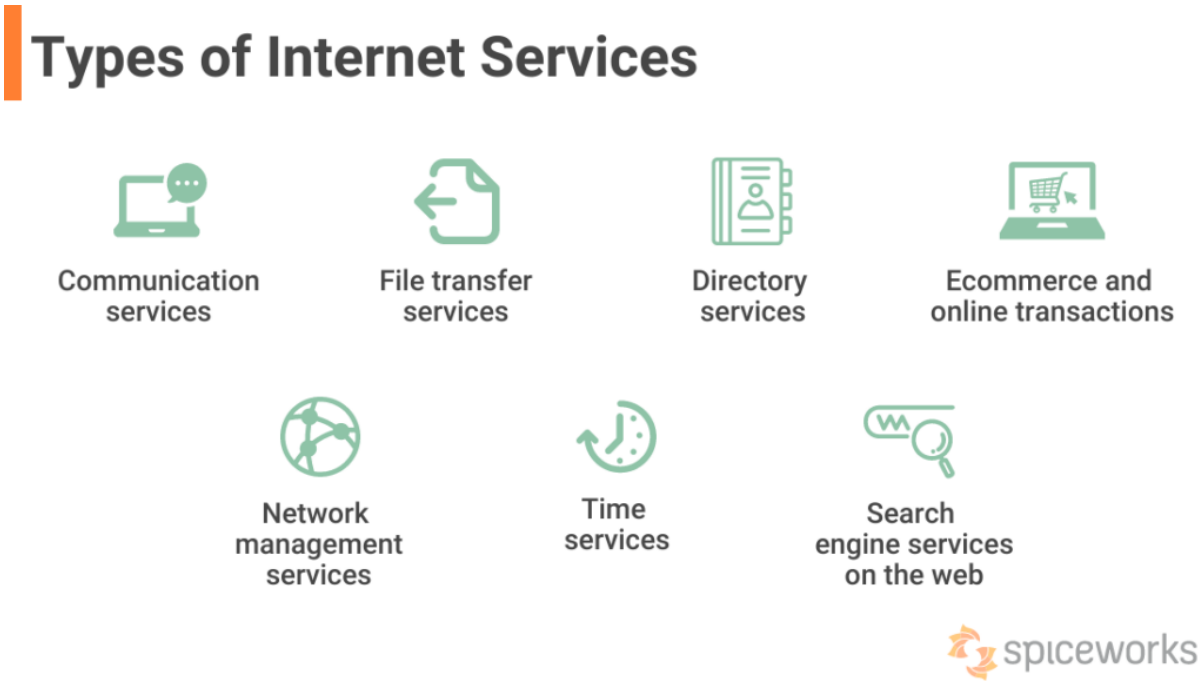
56. Create a web page using HTML, CSS and javascript ii which there will be 4 pages related to Flight booking. First page will show the general information about the Airlines including pictures and management. Second page will show about the food items available. Third will contain the expenditure information in which user fills the requirement using form and java script will show the expenditure and the fourth page is for user registration.

57. Why do we use AJAX in web programming? Write a AJAX program which print table of a given number.

1. **Asynchronous Requests:** AJAX allows us to make asynchronous requests to a web server. This means the browser can continue to interact with the user while waiting for the server's response, improving user experience and responsiveness.
2. **Partial Page Updates:** With AJAX, we can update parts of a web page dynamically without reloading the entire page. This is useful for creating interactive and dynamic user interfaces.
3. **Reduced Bandwidth Usage:** AJAX requests typically send and receive smaller amounts of data compared to full page loads. This can lead to reduced bandwidth usage and faster loading times for web applications.
4. **Improved Performance:** By fetching data asynchronously and updating the page dynamically, AJAX can improve the performance of web applications, especially for tasks that require frequent data updates.

```
<body>
  <h1>AJAX Multiplication Table</h1>
  <input type="number" id="numberInput" placeholder="Enter a number">
  <button onclick="generateTable()">Generate Table</button>
  <div id="tableContainer"></div>

  <script>
    function generateTable() {
      const number = document.getElementById('numberInput').value;
      if (!number) {
        alert('Please enter a number.');
        return;
      }

      let tableHTML = '<table border="1">';
      for (let i = 1; i <= 10; i++) {
        tableHTML += `<tr><td>${number} x ${i}</td><td>${number * i}<
      }
      tableHTML += '</table>';

      document.getElementById('tableContainer').innerHTML = tableHTML
    }
  </script>
```

58. What is internet? Discuss the various internet services in brief.



The internet is a global network that connects millions of devices worldwide. It enables various internet services, including:

1. **World Wide Web (WWW):** Allows access to websites and information via web browsers.
2. **Email:** Facilitates electronic messaging and communication.
3. **File Transfer Protocol (FTP):** Enables file sharing and transfer between computers.
4. **Instant Messaging (IM):** Real-time text-based communication over the internet.
5. **Voice over IP (VoIP):** Voice communication using internet protocols.
6. **Social Media:** Platforms for sharing content and connecting with others.
7. **Online Gaming:** Multiplayer gaming and interaction over the internet.
8. **Video Streaming:** On-demand video content delivery.
9. **Cloud Storage:** Online storage and access to files and data.
10. **Online Banking and E-commerce:** Digital financial transactions and online shopping.

59. What do you mean by web projects?

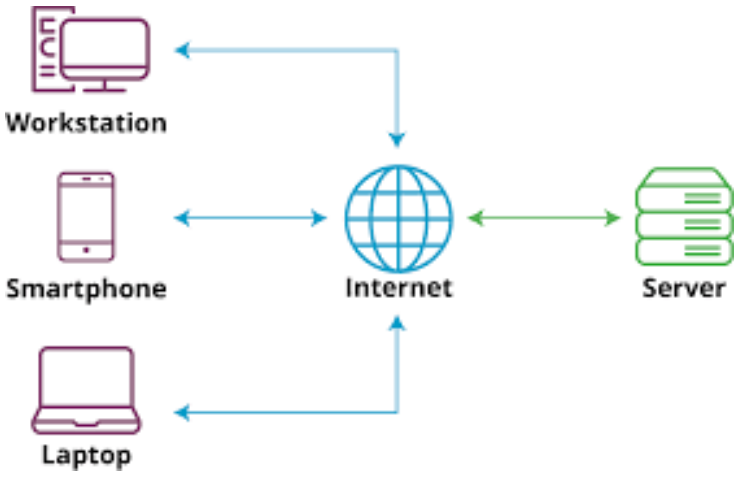
Web projects involve creating applications, websites, or services that run on the internet. These projects use web technologies like HTML, CSS, JavaScript, and server-side languages to build interactive and functional solutions.

Types of Web Projects:

- **Static Websites:** Simple websites with fixed content, often for information or portfolios.
- **Dynamic Websites:** Content changes based on user interaction or data input (databases, server-side scripting).
- **Web Applications:** Interactive applications accessed through web browsers (online tools, e-commerce platforms).
- **Content Management Systems (CMS):** Platforms for creating, managing, and publishing content online (WordPress, Drupal).
- **E-commerce Platforms:** Websites dedicated to buying and selling products or services online.
- **Social Media Platforms:** Platforms for social networking, content sharing, and community building.
- **Web APIs:** Interfaces for communication and data exchange between different systems or applications over the internet.
- **Web Services:** Services or functionalities accessible over the web, often used for integration with other applications or systems.
- **Responsive Design Projects:** Projects that adapt website layout and content to different devices (phones, tablets, desktops).
- **SEO Optimization Projects:** Efforts to improve a website's search engine visibility and ranking.

Overall: Web projects cover a wide range of activities that create and maintain web-based solutions for various needs on the internet.

60. Explain the Client-Server architecture with diagram.



- **Client-Server Architecture:**
  - Clients request services or data from servers over a network.
  - Clients initiate requests, while servers process them and send back responses.
  - Servers handle tasks like data storage, computation, and generating responses based on client requests.
  - Communication between clients and servers occurs over protocols like HTTP, TCP/IP, or WebSocket.

- This architecture enables efficient communication and resource sharing between devices.

61. Write a HTML page with JavaScript that accepts student's information (RollNo, Name, Address, Email) from user and performs following validation:

(i) RollNo field should have only digit (ii) Name field should have only character
(iii) Email should be valid email
Explain the program in brief.

```
<form id="studentForm" onsubmit="return validateForm()">
  <label for="rollNo">Roll Number:</label>
  <input type="text" id="rollNo" name="rollNo" required><br><br>

  <label for="name">Name:</label>
  <input type="text" id="name" name="name" required><br><br>

  <label for="address">Address:</label>
  <input type="text" id="address" name="address"><br><br>

  <label for="email">Email:</label>
  <input type="email" id="email" name="email" required><br><br>

  <input type="submit" value="Submit">
</form>
```

```
<script>
  function validateForm() {
    const rollNo = document.getElementById('rollNo').value;
    const name = document.getElementById('name').value;
    const email = document.getElementById('email').value;

    // Validation for RollNo (only digits)
    if (!/^\d+$/.test(rollNo)) {
      alert('Roll Number should have only digits.');
      return false;
    }

    // Validation for Name (only characters)
    if (!/^[A-Za-z ]+$/.test(name)) {
      alert('Name should have only characters.');
      return false;
    }

    // Validation for Email (valid email format)
    if (!/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/.test(ema
      alert('Email should be a valid email address.');
      return false;
    }

    // Form submission if all validations pass
    alert('Form submitted successfully!');
    return true;
  }
```

62. Write a XML file for Book details (BookNo, Title, Publisher, Price) with one attribute 'Qty' in 'Title' element that shows quantity of the book. Write corresponding DTD ***or*** XML Schema to validate the XML file of Book details. Explain the program in brief.

1. XML File (book_details.xml):

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book Qty="5">
    <BookNo>101</BookNo>
    <Title>The Lord of the Rings</Title>
    <Publisher>HarperCollins</Publisher>
    <Price>25.99</Price>
  </book>
  <book Qty="3">
    <BookNo>102</BookNo>
    <Title>The Hitchhiker's Guide to the Galaxy</Title>
    <Publisher>Pan Macmillan</Publisher>
    <Price>14.99</Price>
  </book>
  </bookstore>
```

2. DTD (book_details.dtd):

XML

```xml
<!ELEMENT bookstore (book+)>
<!ELEMENT book (BookNo, Title, Publisher, Price)>
<!ELEMENT BookNo (#PCDATA)>
<!ELEMENT Title (#PCDATA) >
<!ATTLIST Title Qty CDATA #REQUIRED>
<!ELEMENT Publisher (#PCDATA)>
<!ELEMENT Price (#PCDATA)>
```

3. XML Schema (book_details.xsd):

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.example.com/book_store">

  <xs:element name="bookstore">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="book" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="BookNo" type="xs:string"/>
        <xs:element name="Title" type="xs:string">
          <xs:attribute name="Qty" type="xs:integer" use="required"/>
        </xs:element>
        <xs:element name="Publisher" type="xs:string"/>
        <xs:element name="Price" type="xs:decimal"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

63. Describe benefit of AJAX in a web site. How will you specify handler in AJAX? What are the values of readyState property and their purpose?

Benefits of AJAX in a website:

- Improved user experience with faster and more responsive interactions.
- Reduced server load and bandwidth usage due to asynchronous communication.
- Enhanced interactivity with features like auto-complete and real-time updates.
- Bandwidth efficiency and faster loading times, especially on slower networks.
- Asynchronous processing for concurrent requests without blocking the UI.

To specify a handler in AJAX, you can use the `onreadystatechange` event of the XMLHttpRequest object.

The values of the `readyState` property in AJAX are:

1. 0 (UNSENT): The request is uninitialized.
2. 1 (OPENED): The request is set up.

3. 2 (HEADERS_RECEIVED): The request headers are received.
4. 3 (LOADING): The response is being received.
5. 4 (DONE): The request is complete, and the response is ready.

64. Explain class and id selctors used in CSS with examples of each. What is the purpose of Inline CSS? Explain Inline CSS with example.

Selector Type	Syntax	Example	Purpose
Class Selector	`.className`	`.highlight { color: red; }`	Targets elements with specific class attribute
ID Selector	`#idName`	`#header { background-color: blue; }`	Targets elements with specific id attribute
Inline CSS	`style="..."`	`<div style="color: red;">Text</div>`	Applies styles directly to individual elements

65. Write an XML program for three employee information (empno,empname,desgination,salary) using corresponding DTD or XSD. Explain the program. Include attribute (type) at the element empno and attribute (mode) on element "salary".

1. XML File (employees.xml):

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<employees>
  <employee empno="EN001" type="int">
    <empname>John Doe</empname>
    <designation>Manager</designation>
    <salary mode="monthly">5000.00</salary>
  </employee>
  <employee empno="EN002" type="int">
    <empname>Jane Smith</empname>
    <designation>Developer</designation>
    <salary mode="hourly">30.00</salary>
  </employee>
  <employee empno="EN003" type="int">
    <empname>Alice Brown</empname>
    <designation>Tester</designation>
    <salary mode="contract">10000.00</salary>
  </employee>
</employees>
```

2. DTD (employees.dtd):

XML

```
<!ELEMENT employees (employee+)>
<!ELEMENT employee (empno, empname, designation, salary)>
<!ELEMENT empno (#PCDATA) >
<!ATTLIST empno type CDATA #REQUIRED>
<!ELEMENT empname (#PCDATA)>
<!ELEMENT designation (#PCDATA)>
<!ELEMENT salary (#PCDATA)>
<!ATTLIST salary mode CDATA #REQUIRED>
```

3. XML Schema (employees.xsd):

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.example.com/employees">

  <xs:element name="employees">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="employee" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="employee">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="empno" type="xs:int" use="required">
          <xs:attribute name="type" type="xs:string" use="required"/>
        </xs:element>
        <xs:element name="empname" type="xs:string"/>
        <xs:element name="designation" type="xs:string"/>
        <xs:element name="salary" type="xs:decimal">
          <xs:attribute name="mode" type="xs:string" use="required"/>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

66. Create an HTML page named as “String_Math.html” and within the script tag define some String variables and use different String functions to demonstrate the use of predefined function. Do the same for the Math function.

```
<script>
  // Define String variables
  let str1 = 'Hello';
  let str2 = 'World';

  // Concatenate strings
  let concatStr = str1.concat(' ', str2);
  console.log('Concatenated String:', concatStr);

  // Get string length
  let strLength = concatStr.length;
  console.log('Length of Concatenated String:', strLength);

  // Convert to uppercase
  let upperCaseStr = concatStr.toUpperCase();
  console.log('Uppercase String:', upperCaseStr);

  // Find index of a substring
  let indexOfWorld = concatStr.indexOf('World');
  console.log('Index of "World":', indexOfWorld);

  // Extract substring
  let subStr = concatStr.substr(6, 5);
  console.log('Substring:', subStr);
</script>
```

```
<h1>Math Functions</h1>
<script>
  // Use Math functions
  let num1 = 25;
  let num2 = 10;

  // Addition
  let sum = num1 + num2;
  console.log('Sum:', sum);

  // Subtraction
  let difference = num1 - num2;
  console.log('Difference:', difference);

  // Multiplication
  let product = num1 * num2;
  console.log('Product:', product);

  // Division
  let quotient = num1 / num2;
  console.log('Quotient:', quotient);

  // Square root
  let sqrtNum = Math.sqrt(num1);
  console.log('Square Root:', sqrtNum);
</script>
```

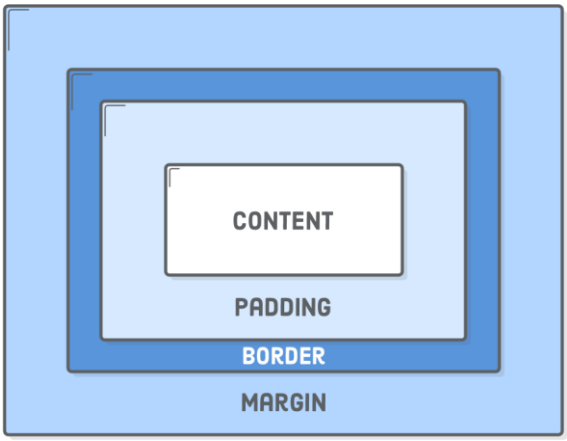
67. Define box model in CSS with the help of block diagram.

The box model in CSS is a fundamental concept that describes how elements on a web page are rendered and how their dimensions are calculated. Here's a brief explanation along with a block diagram to illustrate the box model:

Box Model in CSS:

The box model consists of four main components around an HTML element:

1. **Content:** The actual content of the element, such as text, images, or other media.
2. **Padding:** Space between the content and the element's border. Padding helps create space within the element.
3. **Border:** A border that surrounds the padding and content. Borders can have different styles, colors, and thicknesses.
4. **Margin:** Space between the element's border and neighboring elements. Margins create space around the element.



68. Enlist the advantages of XML schema over Document Type Definition.

Advantages of XML Schema over Document Type Definition (DTD):

1. Strongly Typed: XML Schema supports strong data typing for precise data validation.
2. Namespaces Support: XML Schema handles namespaces more effectively than DTDs.
3. More Expressive: XML Schema offers a wider range of data types and validation rules.
4. Modularity: XML Schema allows for modular schema design and code reusability.
5. Better Documentation: XML Schema supports detailed documentation within the schema.

6. Data Binding: XML Schema works well with data binding tools for code generation.
7. Internationalization: XML Schema provides better support for internationalization.

69. What is the use of CSS classes in web designing?

- **Consistency:** Classes help maintain consistent styles across multiple elements.
- **Reusability:** Styles defined in classes can be reused across different elements.
- **Modularity:** CSS becomes more organized and manageable with classes.
- **Selective Application:** Classes allow for targeted styling based on element context.
- **Variations and Overrides:** Classes enable creating variations or overrides of styles.
- **Responsive Design:** Classes play a role in implementing responsive styles based on device type or screen size.
- **Efficiency:** Using classes reduces code duplication and improves design efficiency.

70. What is the use of bootstrap in web programming? How can we add bootstrap in our web page?

Bootstrap is a front-end framework used in web programming for:

- **Responsive Design:** Easily create responsive layouts.
- **Pre-styled Components:** Access pre-designed UI components.
- **Customizability:** Customize styles to fit design needs.
- **Cross-browser Compatibility:** Ensures compatibility across browsers.
- **JavaScript Plugins:** Includes interactive JavaScript components.

To add Bootstrap to your web page:

1. Download Bootstrap files or link to Bootstrap CDN.
2. Include Bootstrap CSS and JavaScript files in your HTML.
3. Start using Bootstrap classes and components in your code.

,,,,,,,,,,,,,,,,,,,,,,,

Develop and interpret a XHTML file that includes Javascript script for the following problems:
a) Input: A number n obtained using prompt box
Output: The first n Fibonacci numbers
b) Input: A number n obtained using prompt
Output: A table of numbers from 1 to n and their squares using alert

```
<html>
    <body>
        <script type="text/javascript">
            //initialize variables
            var fib1=0,fib2=1,fib=0;
            var n=prompt("enter a number");
            if(n!=null && n>0)
                {
                    document.write("<h1>First " + n + " fibonacci numbers are: </h1><br>");
                    //if input is one number
                    if(n==1)
                        document.write("<h1>" + fib1 + "</h1><br>");
                    //if input is two numbers
                    else
                        document.write("<h1>" + fib1 + "</h1><br><h1>" + fib2 + "</h1><br>");
                        //if input is more than two numbers, find the next Fibonacci number
                        for(i=3;i<=n;i++)
                            {
                                fib=fib1+fib2;
                                document.write("<h1>" + fib + "</h1><br>");
                                fib1=fib2;
                                fib2=fib;
                            }
                }
            else
                alert("No proper input");
        </script>
    </body>
 </html>
```

,,,

XML file which will display the Book information

AIM: Write an XML file which will display the Book information.

It includes the following:

1) Title of the book
2) Author Name
3) ISBN number
4) Publisher name
5) Edition
6) Price

Write a Document Type Definition (DTD) to validate the above XML file.

Display the XML file as follows.

The contents should be displayed in a table. The header of the table should be in color GREY. And the Author names column should be displayed in one color and should be capitalized and in bold. Use your own colors for remaining columns.

Use XML schemas XSL and CSS for the above purpose.

DESCRIPTION:

DTD vs XML Schema

The DTD provides a basic grammar for defining an XML Document in terms of the metadata that comprise the shape of the document. An XML Schema provides this, plus a detailed way to define what the data can and cannot contain. It provides far more control for the developer over what is legal, and it provides an Object Oriented approach, with all the benefits this entails.

Many systems interfaces are already defined as a DTD. They are mature definitions, rich and complex. The effort in re-writing the definition may not be worthwhile.

DTD is also established, and examples of common objects defined in a DTD abound on the Internet -- freely available for re-use. A developer may be able to use these to define a DTD more quickly than they would be able to accomplish a complete re-development of the core elements as a new schema.

Finally, you must also consider the fact that the XML Schema is an XML document. It has an XML Namespace to refer to, and an XML DTD to define it. This is all overhead. When a parser examines the document, it may have to link this all in, interpret the DTD for the Schema, load the namespace, and validate the schema, etc., all *before* it can parse the actual XML document in question. If you're using XML as a protocol between two systems that are in heavy use, and need a quick response, then this overhead may seriously degrade performance.

- **Write a Document Type Definition (DTD) to validate the XML file.**

PROGRAM:

XML document (bookstore.xml)

```
<bookstore>
    <book>
        <title>web programming</title>
        <author>chrisbates</author>
        <ISBN>123-456-789</ISBN>
        <publisher>wiley</publisher>
        <edition>3</edition>
        <price>350</price>
    </book>
    <book>
        <title>internet worldwideweb</title>
        <author>ditel&amp;ditel</author>
        <ISBN>123-456-781</ISBN>
        <publisher>person</publisher>
        <edition>3</edition>
        <price>450</price>
```

</book>

</bookstore>

XML document Validation using DTD

DTD document (bookstore.dtd)

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT bookstore (book+)>
<!ELEMENT book (title,author,ISBN,publisher,edition,price)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT edition (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

Bookstore.xml

<!DOCTYPE bookstore SYSTEM "C:\Documents and Settings\Administrator\My Documents\bookstore.dtd">

```
<bookstore>
        <book>
                <title>web programming</title>
                <author>chrisbates</author>
                <ISBN>123-456-789</ISBN>
                <publisher>wiley</publisher>
                <edition>3</edition>
                <price>350</price>
        </book>
        <book>
                <title>internet worldwideweb</title>
                <author>ditel&amp;ditel</author>
                <ISBN>123-456-781</ISBN>
                <publisher>person</publisher>
                <edition>3</edition>
```

```
        <price>450</price>
    </book>
</bookstore>
```

XML document Validation using DTD

XML Schema (bookstore.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="bookstore">
        <xs:complexType>
        <xs:sequence>
<xs:element name="book" maxOccurs="unbounded">
<xs:complexType>
    <xs:sequence>
<xs:element name="title"       type="xs:string"></xs:element>
<xs:element name="author"  type="xs:string"></xs:element>
<xs:element name="ISBN"    type="xs:string"></xs:element>
<xs:element name="publisher"         type="xs:string"></xs:element>
<xs:element name="edition"  type="xs:int"></xs:element>
<xs:element name="price"     type="xs:decimal"></xs:element>
    </xs:sequence>
</xs:complexType>


</xs:element>

        </xs:sequence>
        </xs:complexType>

    </xs:element>
</xs:schema>
```

Bookstore.xml

<bookstore xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="C:\Documents and Settings\Administrator\My Documents\bookstore.xsd">

```
    <book>
        <title>web programming</title>
        <author>chrisbates</author>
        <ISBN>123-456-789</ISBN>
        <publisher>wiley</publisher>
        <edition>3</edition>
        <price>350</price>
    </book>
    <book>
        <title>internet worldwideweb</title>
        <author>ditel&amp;amp;ditel</author>
        <ISBN>123-456-781</ISBN>
        <publisher>person</publisher>
        <edition>3</edition>
        <price>450</price>
    </book>
</bookstore>
```

- **Display the XML file as follows.**

PROGRAM:

XML:

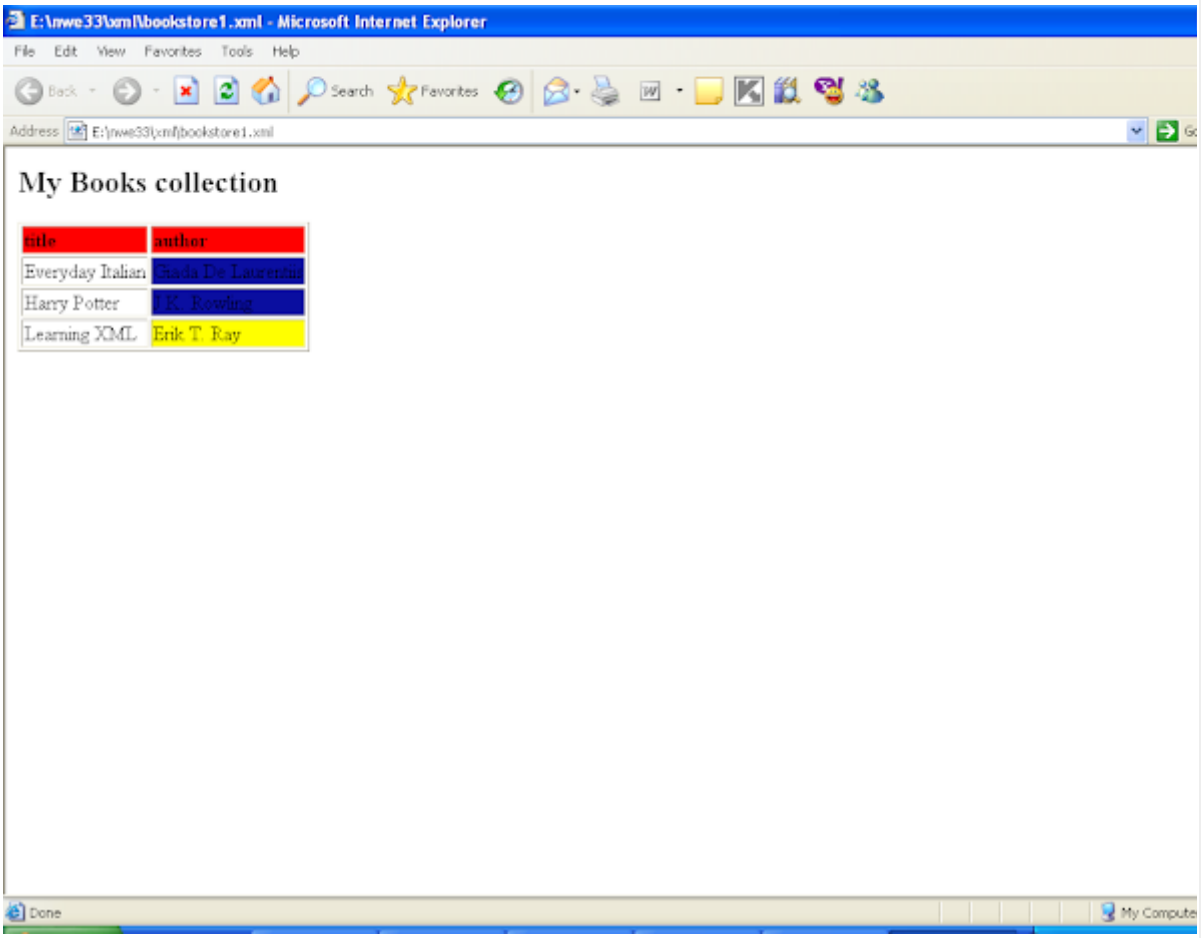
```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="bookstore.xsl"?>

<bookstore>
<book>
  <title>Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>
<book>
  <title>Harry Potter</title>
  <author>J K. Rowling</author>
```

```
  <year>2005</year>
  <price>29.99</price>
</book>
<book>
  <title>Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
</book>
</bookstore>
```

XSL:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
<h2> My Books collection</h2>
<table border="1">
<tr bgcolor="red">
<th align="left">title</th>
<th align="left">author</th>
</tr>
<xsl:for-each select="bookstore/book">
<tr>
<td><xsl:value-of select="title"/></td>
<xsl:choose>
<xsl:when test="price &gt; 30">
<td bgcolor="yellow"><xsl:value-of select="author"/></td>
</xsl:when>
<xsl:when test="price &gt; 10">
<td bgcolor="magneta"><xsl:value-of select="author"/></td>
</xsl:when>
<xsl:otherwise>
<td><xsl:value-of select="author"/></td>
</xsl:otherwise>
</xsl:choose>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

OUTPUT:



RESULT: Thus the XML stylesheets are successfully used to display the content in a table format.

SOURCE:PVPSIT FOR JNTUK