Web Technology

Unit I - Web Introduction

---

1. What is the Web?

- The **World Wide Web (WWW)** is an information space where documents and resources are identified via URLs and accessed over the Internet using **HTTP/HTTPS**.
- It runs on a **client-server model** (browser = client, server = host of data).

2. Protocols Governing the Web

- **HTTP (HyperText Transfer Protocol)**: For communication between browser and server.
- **HTTPS**: Secured version of HTTP (uses SSL/TLS).
- **FTP (File Transfer Protocol)**: For transferring files.
- **SMTP/POP3/IMAP**: For email.

3. Web Development Strategies

- Plan the UI/UX (what users see).
- Backend: Handle data and logic.
- Use frameworks to speed up work.
- Responsive design (mobile-first).
- Optimize performance, security, and SEO.

4. Web Applications

- **Static**: No backend processing (e.g., HTML + CSS only)
- **Dynamic**: Backend logic, databases, real-time updates (e.g., YouTube, Facebook)

5. Web Project and Team Roles

- **Frontend Developer**: HTML, CSS, JS – UI.
- **Backend Developer**: Node.js, Python, etc. – logic + database.
- **Full-stack Developer**: Both frontend + backend.
- **UI/UX Designer**: Design and experience.
- **DevOps Engineer**: Deployment and maintenance.

💼 Industry-Level Insights

✅ **Modern Web Stack (MERN or similar):**

- **Frontend**: React / Next.js
- **Backend**: Node.js / Express
- **Database**: MongoDB / PostgreSQL
- **Deployment**: Vercel / Netlify / Render / AWS

✅ **APIs & Protocols:**

- **REST API** or **GraphQL** for communication.
- **WebSockets** for real-time apps (e.g., chat).
- **CORS, SSL, DNS, CDN** — you deal with all of these.

✅ **Dev Tools:**

- GitHub for version control.
- Postman for API testing.
- Figma for UI design.
- CI/CD pipelines for deployment.

✅ Summary: What You Need to Learn for Both

Concept	College	Industry
What is Web & Protocols	✅	✅
Web Development Lifecycle	✅	✅
Web App Types	✅	✅
Team Roles & Structure	✅	✅
Modern Tools (React, Git, API)	❌	✅
Deployment (CI/CD, Vercel, etc.)	❌	✅

Unit II – HTML, Tables, Forms, CSS

🎓 College-Level: HTML & CSS Basics

1. HTML (HyperText Markup Language)

HTML is the standard markup language used to create web pages.
It defines **structure**, not style.

🔹 **Basic Tags:**

<html>, <head>, <title>, <body>, <h1> to <h6>, <p>, <br>, <hr>, <div>, <span>

🔹 **List Tags:**

- **Ordered list**:

~~~
<ol>
  <li>Item 1</li>
</ol>
~~~

- **Unordered list**:

~~~
<ul>
  <li>Item 1</li>
</ul>
``>
~~~

#### 🔹 Table Tags:

~~~
```html
<table>
  <tr><th>Name</th><th>Age</th></tr>
  <tr><td>John</td><td>30</td></tr>
</table>
~~~

🔹 **Forms:**

Used to collect user input:

```
<form action="/submit" method="POST">
  <input type="text" name="name" />
  <input type="submit" />
</form>
```

🔹 **Image:**

```
<img src="image.jpg" alt="Description" />
```

🔹 **Frames (deprecated but in syllabus):**

```
<frameset cols="25%,75%">
  <frame src="menu.html">
  <frame src="main.html">
</frameset>
```

✅ **Note:** <frameset> is outdated and replaced by **iframes** or modern SPAs.

---

2. CSS (Cascading Style Sheets)

CSS is used for **styling** HTML content.

🔹 **Basic Syntax:**

```
selector {
  property: value;
}
```

🔹 **Example:**

```
<style>
  body {
    background-color: lightblue;
    font-family: Arial;
  }
  h1 {
    color: navy;
    text-align: center;
```

```
 }
</style>
```

🔹 **Ways to Use CSS:**

- Inline: <p style="color:red;">Text</p>
- Internal: Inside <style> tag in HTML head
- External: Link .css file using <link rel="stylesheet" href="style.css">

💼 Industry-Level HTML & CSS

✅ **What Professionals Use**

Feature	**Description**
HTML5	Latest standard (semantic tags: <section>, <article>, <main>, etc.)
CSS3	Modern features: animations, flexbox, grid, transitions
Flexbox & Grid	Used for responsive layouts
Media Queries	Make designs responsive
SCSS/SASS	Preprocessors for better CSS structure
Tailwind CSS / Bootstrap	Utility-first CSS libraries to speed up styling
Accessibility (ARIA tags)	For screen readers and inclusive design
SEO-friendly structure	Using correct heading, alt tags, etc.
Responsive Design	Works on all screen sizes
Lighthouse + DevTools	Performance + accessibility audits

✅ **Example with Tailwind CSS:**

```
<h1 class="text-4xl font-bold text-center text-blue-600">
 Hello, World!
</h1>
```

✅ **Modern HTML Forms:**

- Real-time validation with JavaScript

- API submission (AJAX/fetch)
- File uploads, date pickers, and more
- Often styled with Tailwind or custom components

✅ **Summary Table: HTML + CSS**

Topic	**College**	**Industry**
HTML Basic Tags	✅	✅
Tables, Lists, Forms	✅	✅
Frames	✅	❌ (deprecated)
CSS Basics	✅	✅
Semantic HTML (HTML5)	❌	✅
CSS Flexbox/Grid	❌	✅
Responsive Design	❌	✅
Tailwind / Bootstrap	❌	✅
Form Validation (JS/AJAX)	❌	✅

Unit III – XML (Extensible Markup Language)

📚 **College-Level Understanding**

🔸 **What is XML?**

- XML stands for **eXtensible Markup Language**.
- It is used to **store and transport data**, not display it (unlike HTML).
- It is **both human-readable and machine-readable**.
- XML is platform-independent and language-neutral.

✅ **Example:**

```
<student>
  <name>John Doe</name>
  <roll>101</roll>
  <branch>Computer Science</branch>
</student>
```

🔹 XML Syntax Rules (Important in College):

- Tags are case-sensitive.
- Every tag must be closed.
- Tags must be properly nested.
- There must be exactly one root element.

❌ **Invalid:**

```
<name>John</Name> <!-- tag mismatch -->
```

🔹 DTD (Document Type Definition)

Used to define the **structure** of XML documents.

👉 **Internal DTD Example:**

```
<!DOCTYPE student [
  <!ELEMENT student (name, roll, branch)>
  <!ELEMENT name (#PCDATA)>
  <!ELEMENT roll (#PCDATA)>
  <!ELEMENT branch (#PCDATA)>
]>
<student>
  <name>John</name>
  <roll>101</roll>
  <branch>CSE</branch>
</student>
```

👉 **External DTD:**

- You can reference a .dtd file using:

```
<!DOCTYPE student SYSTEM "student.dtd">
```

🔹 **XML Schema (XSD)**

- Schema is **more powerful than DTD**.
- Supports **data types**, namespaces, default values, etc.

👉 **XML Schema Example:**

```
<xs:element name="roll" type="xs:int"/>
```

You use a .xsd file or inline schema to validate XML.

🔹 **Presenting XML: Using CSS or XSL**

XML with CSS:

```
<?xml-stylesheet type="text/css" href="style.css"?>
```

XML with XSL (eXtensible Stylesheet Language):

```
<?xml-stylesheet type="text/xsl" href="style.xsl"?>
```

XSL lets you transform XML into other formats like HTML.

💼 **Industry-Level XML Usage**

✅ **Where XML Is Used in Real Projects:**

Domain	**XML Usage**
Web Services (SOAP)	Data exchanged between servers
Android Apps	Layout files (.xml) for UI
Configuration Files	.pom.xml in Maven, .plist in iOS
Data Interchange	RSS feeds, SVG images, etc.
Documents	Office files (Word, Excel are XML-based)

✅ **Industry Standards:**

- **Use JSON** instead of XML for most modern REST APIs (lighter, faster).
- XML is still used in:
    - Banking
    - Legacy systems
    - Telecom
    - Enterprise software

🔥 Modern Comparison: XML vs JSON

Feature	XML	JSON
Syntax	Verbose	Clean, lightweight
Used in	Legacy, SOAP, config	REST APIs, modern apps
Readability	Complex	Easy
Schema support	Yes (DTD, XSD)	Yes (JSON Schema)

✅ Summary: XML (College vs Industry)

Concept	College	Industry
XML Syntax	✅	✅
DTD	✅	✅
XML Schema (XSD)	✅	✅
XML Presentation (CSS)	✅	❌ (rare)
XML with XSLT	✅	✅ (limited)
XML in Real Systems	❌	✅
JSON Replacing XML	❌	✅

Ready to go ahead with **Unit IV – JavaScript, Events, AJAX, VBScript, CGI**?

Unit IV – JavaScript, Events, AJAX, VBScript, CGI

📘 Part 1: JavaScript (College-Level + Modern Use)

🔹 What is JavaScript?

- A **scripting language** that runs in the browser.
- Used to **manipulate HTML/CSS**, handle events, validate forms, and create interactivity.

✅ Basic Syntax:

```
let name = "Ghanshyam";

alert("Hello " + name);
```

🔹 JavaScript in HTML:

```
<script>

  document.write("Welcome to Web Tech!");

</script>
```

🔹 JavaScript Document Object Model (DOM):

Allows you to interact with and manipulate HTML.

```
<button onclick="changeText()">Click Me</button>

<p id="demo">Original</p>


<script>

function changeText() {

  document.getElementById("demo").innerText = "Changed!";

}

</script>
```

🔹 JavaScript Forms and Validation:

```
function validateForm() {
  let x = document.forms["myForm"]["name"].value;
  if (x === "") {
    alert("Name must be filled out");
    return false;
  }
}
```

🔹 JavaScript Events:

Event	Triggered When...
onclick	Element is clicked
onload	Page finishes loading
onmouseover	Mouse hovers
onkeyup	Key is released

```
<button onclick="alert('Clicked!')">Click Me</button>
```

💼 JavaScript in Industry

- Used for frontend frameworks like **React**, **Vue**, **Angular**.
- Backend with **Node.js**.
- Event handling is done via addEventListener() in modern apps.

```
document.querySelector("button").addEventListener("click", () => {
  alert("Clicked!");
});
```

⚡ Part 2: AJAX (Asynchronous JavaScript and XML)

🔹 What is AJAX?

- It allows you to send/receive data **without reloading the web page**.
- Uses: autocomplete, live search, chat, etc.

✅ **Example (Vanilla JS):**

```
let xhr = new XMLHttpRequest();
xhr.open("GET", "data.txt", true);
xhr.onload = function () {
  if (xhr.status == 200) {
    document.getElementById("demo").innerHTML = xhr.responseText;
  }
};
xhr.send();
```

💡 Modern alternative: fetch() and Axios in industry.

```
fetch('/api/user')
  .then(res => res.json())
  .then(data => console.log(data));
```

🧠 Part 3: VBScript (Only College-Relevant)

- Microsoft-only scripting language, **mostly obsolete**.
- Used inside HTML pages but works **only in Internet Explorer**.

```
<script type="text/vbscript">
  MsgBox "Hello VBScript"
</script>
```

❌ Industry does **not** use VBScript anymore.

🌐 Part 4: CGI (Common Gateway Interface)

🔹 **What is CGI?**

- A standard protocol for web servers to **execute programs** (usually in Python, Perl, etc.) to generate dynamic content.

✅ **Example:**

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
print "<html><body>Hello CGI</body></html>";
```

💡 **Real-life:**

- CGI was used before PHP, Node.js, etc.
- **Now replaced** by faster server-side frameworks.

🔍 **Summary Table (College + Industry)**

Topic	College Level ✅	Industry Use ✅	Notes
JavaScript	✅	✅	Core web tech
Events	✅	✅	addEventListener preferred
Form Handling	✅	✅	With validation libs
AJAX	✅	✅	Now replaced with fetch, Axios
VBScript	✅	❌	Legacy, avoid in real-world
CGI	✅	❌ (legacy only)	Replaced by modern backend tools

Unit V – Server-Side Programming (ASP, ASP.NET, JSP, Tomcat, Debugging)

🧠 **First, What is Server-Side Programming?**

- Unlike JavaScript (which runs in browser), **server-side code runs on a web server**.
- It handles:
  - **Database queries**
  - **Business logic**
  - **User authentication**

- **Dynamic content generation**

📘 College Topics Breakdown

🔹 ASP (Active Server Pages) — *(Obsolete but in syllabus)*

- Microsoft’s old server-side scripting.
- Syntax similar to VBScript.
- Executes on IIS (Internet Information Services).

```
<%
  Response.Write("Hello from ASP")
%>
```

❌ **Not used in the industry anymore** (replaced by ASP.NET, Node.js, etc.)

🔹 ASP.NET — *(Still used in enterprise systems)*

- A **.NET-based framework** for web apps (developed by Microsoft).
- Supports C# and VB.NET.
- Industry still uses **ASP.NET Core** for high-performance APIs.

```
public IActionResult Hello() {
  return Content("Hello from ASP.NET");
}
```

💼 **Industry Usage:**

- Used in large-scale enterprise apps.
- Combined with SQL Server, IIS, Azure, etc.

🔹 JSP (Java Server Pages)

- Java-based web technology.
- Used with **Tomcat server** to embed Java code in HTML.

- Runs on the server, renders dynamic content.

```
<%@ page language="java" %>
<html>
<body>
<%= "Hello from JSP!" %>
</body>
</html>
```

🔧 **Tomcat Server**

- Open-source **Java servlet container**.
- Deploys JSP + Servlets.

```
# JSP + Servlet deployment on Tomcat
# Place `.jsp` files inside: /webapps/myApp/
```

🛠️ **JSP Application Design Includes:**

Component	Purpose
jspInit()	Initialize resources
jspDestroy()	Cleanup before shutting down
jspService()	Handles each client request
JSP Tags	<%= %>, <% %>, <%! %>
Servlet Mapping	Configured in web.xml

🧪 **Debugging in JSP / ASP.NET**

Technique	Used For
console.log() or System.out.println()	Print debug values
Breakpoints	Used in IDEs like Eclipse / VS

Technique	Used For
Logging	Log files for errors and tracing

```
<%
System.out.println("Debug info here...");
%>
```

💼 Industry Alternatives Today (Modern Stack)

Legacy	Modern Replacement
ASP	Node.js, Django, Flask
ASP.NET	ASP.NET Core, Blazor
JSP + Tomcat	Spring Boot (Java), Express.js
CGI	REST APIs, GraphQL

✅ Summary (College + Industry)

Topic	College Level ✅	Industry Use ✅	Notes
ASP	✅	❌	Legacy only
ASP.NET	✅	✅	In Microsoft enterprise
JSP	✅	⚠️ (declining)	Legacy Java apps
Tomcat	✅	⚠️ (declining)	Used with JSP
Modern Alt.	❌	✅	Node.js, Flask, Spring Boot

🔚 Done! This completes the full Web Technology syllabus — college + industry level.