Topic 1 Introduction to Internet and World Wide Web

CST4013 | Website Designing

Learning outcomes

- Describe the concept of internet and website.
- Discuss the categories of website development.

Internet and World Wide Web

Internet	World Wide Web (WWW)
The internet is a global network of interconnected computers and devices that communicate using standardized protocols.	The WWW is a global system of interconnected documents and resources, accessed via the internet.
It allows for the transfer of data and information between connected devices regardless of their location.	It allows users to navigate between web pages by clicking on hyperlinks, which are typically denoted by blue, underlined text
The internet enables various services such as email, web browsing, file sharing, online gaming, and more.	These web pages may contain various types of content, including text, images, videos, and interactive elements.

Website and Webpage

Webpage

- A webpage is a single document or file containing content that is displayed in a web browser.

Website

- A website is a collection of related web pages and other digital assets (such as images, videos, and interactive elements) that are hosted on a web server and accessible via the Internet.

Types of Websites

Static Websites

- A static website consists of web pages with fixed content.
- The content of these pages is delivered exactly as stored, meaning it doesn't change unless manually updated by a developer.

Dynamic Websites

- A dynamic website is more complex and provides interactive features, with content that can change based on user interactions or other factors.
- This type of site can pull data from databases and adjust its content accordingly.

Types of Websites

Static Websites Characteristics

- Fixed content
- Simple structure
- No server-side processing
- Faster loading
- Limited Interactivity

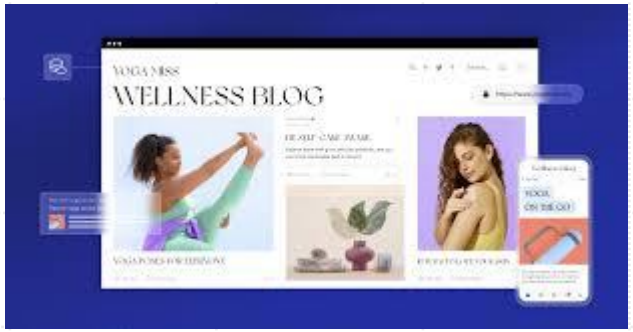
Dynamic Websites Characteristics

- Interactive content
- Server-side scripting
- Database integration
- Complexity

Types of Websites

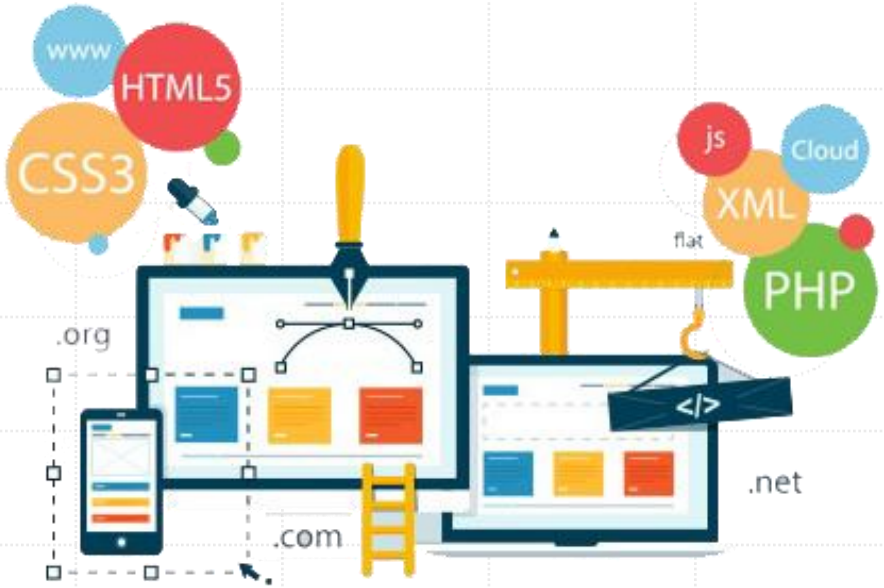
Static Websites uses

- Personal blogs
- Portfolio websites
- Business websites



Dynamic Websites uses

- E-commerce websites
- Social media platform
- Banking websites



Web Designing

- Web designing is the process of creating the visual appearance and layout of websites.
- It involves a combination of graphic design, user interface (UI) design, and user experience (UX) design to produce aesthetically pleasing and functional websites

Web Development

- Web development refers to the process of building and maintaining websites.
- It involves a combination of tasks and skills used to create the various aspects of a website or web application, from its layout and design to its functionality and interactivity.

Web Development Categories

- **Front-End Development (Client-Side)**
  - This focuses on what the user interacts with directly in a web browser.
  - It involves creating the visual aspects of the website such as layout, design, and navigation.
  - Front-end developers use HTML (Hypertext Markup Language) to structure the content, CSS (Cascading Style Sheets) for styling and layout, and JavaScript to add interactivity and dynamic elements.
  - The goal is to provide a seamless and engaging user experience.

Web Development Categories

- **Back-End Development (Server-Side)**
  - This involves the behind-the-scenes operations of a website, including managing databases, user authentication, and server-side logic.
  - Back-end developers typically work with server-side languages such as PHP, Python, Ruby, Java, and databases like MySQL, PostgreSQL, or MongoDB.
  - The back end is responsible for processing requests made by the front end, retrieving or storing data, and sending the appropriate response back to the client.

Web Development Categories

- **Full-Stack Development**
  - A full-stack developer works on both the front-end and back-end of a website or web application.
  - They are proficient in both client-side and server-side technologies, allowing them to handle the entire development process.

Technologies of Web Development

- **HTML (Hypertext Markup Language)**
- HTML is the backbone of every web page.
- This is the standard language used to create and structure content on the web.
- It defines the structure of a webpage using tags and elements, like headings, paragraphs, images, and links.

Technologies of Web Development

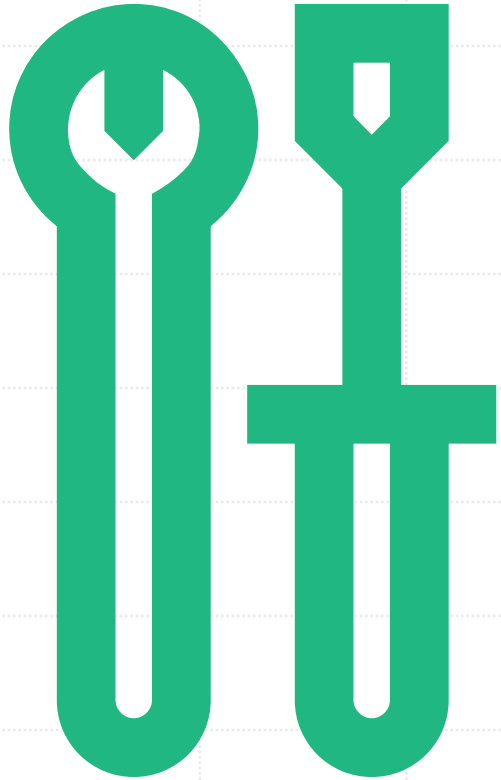
- **CSS (Cascading Style Sheets)**
- CSS controls the look and feel of the website.
- It styles the HTML elements, including setting colors, fonts, spacing, and positioning, and ensures a consistent layout across different screen sizes and devices.

Technologies of Web Development

- **JavaScript**
- JavaScript adds interactivity to websites.
- It can manipulate HTML and CSS dynamically, responding to user actions such as clicks or form submissions, and allowing for features like animations, real-time updates, and interactive elements.

Setting Up the Development Environment

1. Choosing a Text Editor or Integrated Development Environment (IDE)
2. Installing a Web Browser
3. Setting Up a Local Web Server
4. Version Control with Git
5. Installing Front-End and Back-End Tools
6. Package Managers and Frameworks
7. Testing and Debugging Tools



Choosing a Text Editor or Integrated Development Environment (IDE)

- A text editor or IDE is where you will write the code for your website.
- **Visual Studio Code (VS Code)**
  - A powerful, lightweight text editor with extensive plugin support for HTML, CSS, JavaScript, and other languages.
  - It offers features like code completion, debugging tools, and version control integration.
- **Sublime Text**
  - Known for its speed and simplicity,
  - It has syntax highlighting and supports various programming languages.
- **Atom**
  - A customizable, open-source text editor that offers features like a built-in package manager and a wide range of plugins.

Installing a Web Browser

- A modern web browser is essential for testing and debugging.
- Developers should use multiple browsers for cross-browser compatibility testing includes:
  - **Google Chrome**: Known for its developer tools (DevTools), which allow you to inspect elements, debug JavaScript, and test performance.
  - **Mozilla Firefox**: Offers similar developer tools to Chrome, with additional focus on privacy and security.
  - **Safari** and **Microsoft Edge** are also widely used for testing.

Setting Up a Local Web Server

XAMPP

- A free and open-source cross-platform web server solution package that includes Apache, MySQL, PHP, and Perl.
- It's a great option for developing PHP-based websites.

MAMP

- Similar to XAMPP but designed for macOS.
- It allows you to run a local server environment using Apache, MySQL, and PHP.

Live Server (VS Code Extension)

- For simple static websites (HTML, CSS, JavaScript), the Live Server extension in VS Code creates a local server and automatically reloads the page when changes are made.

Version Control with Git

- Git is a version control system that allows developers to track changes in their code and collaborate with other developers.
- Git:
  - Install Git from git-scm.com.
  - It's a command-line tool but also use it via Git GUI applications like GitHub Desktop or SourceTree.
- GitHub:
  - A popular online platform to store and share code repositories.
  - Can use GitHub to host your projects, track issues, and collaborate with others.

Installing Front-End and Back-End Tools

- **Front-End Tools**
- Node.js: A JavaScript runtime environment used for building server-side applications. Node.js comes with npm (Node Package Manager) that helps you manage libraries and dependencies.
- CSS Preprocessors (e.g., Sass, LESS): These tools extend CSS by adding features like variables, nested rules, and mixins. Install them using npm.
- JavaScript Frameworks (e.g., React, Angular, Vue.js): These libraries and frameworks help organize and speed up front-end development by offering reusable components and advanced features

Installing Front-End and Back-End Tools

- **Back-End Tools**
- PHP, Python (Django, Flask), or Ruby on Rails
  - These are some of the most popular back-end programming languages and frameworks.
  - Setting up these environments will require downloading the language's runtime and a web server (e.g., Apache or Nginx).
- Database Management Systems
  - MySQL, PostgreSQL, or MongoDB are commonly used databases.
  - Install a local version or use a service like MongoDB Atlas for cloud-based storage.

Package Managers and Frameworks

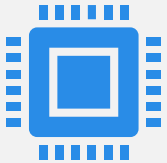
- npm (Node Package Manager) helps you manage JavaScript libraries and dependencies.
- Yarn: A popular alternative to npm, which speeds up the package management process.
- Frameworks and Libraries:
  - For JavaScript, React, Vue.js, and Angular provide powerful ways to build dynamic, single-page applications.
  - CSS frameworks like Bootstrap or Tailwind CSS provide pre-built styles and components that help create responsive designs quickly.

Testing and Debugging Tools

- Browser Developer Tools: Use the built-in dev tools in browsers like Chrome and Firefox to inspect the HTML/CSS structure, debug JavaScript, and monitor network activity.
- Linting Tools: These tools help identify issues in your code (e.g., ESLint for JavaScript, Stylelint for CSS) and enforce coding standards.
- Automated Testing Tools: Frameworks like Jest (for JavaScript) or Selenium (for end-to-end testing) help automate testing processes to ensure functionality across all parts of your website.

Client-Server Model

The client-server model is a fundamental concept in computer networks and web development, where two entities, the client and the server, communicate over a network to exchange data or services.

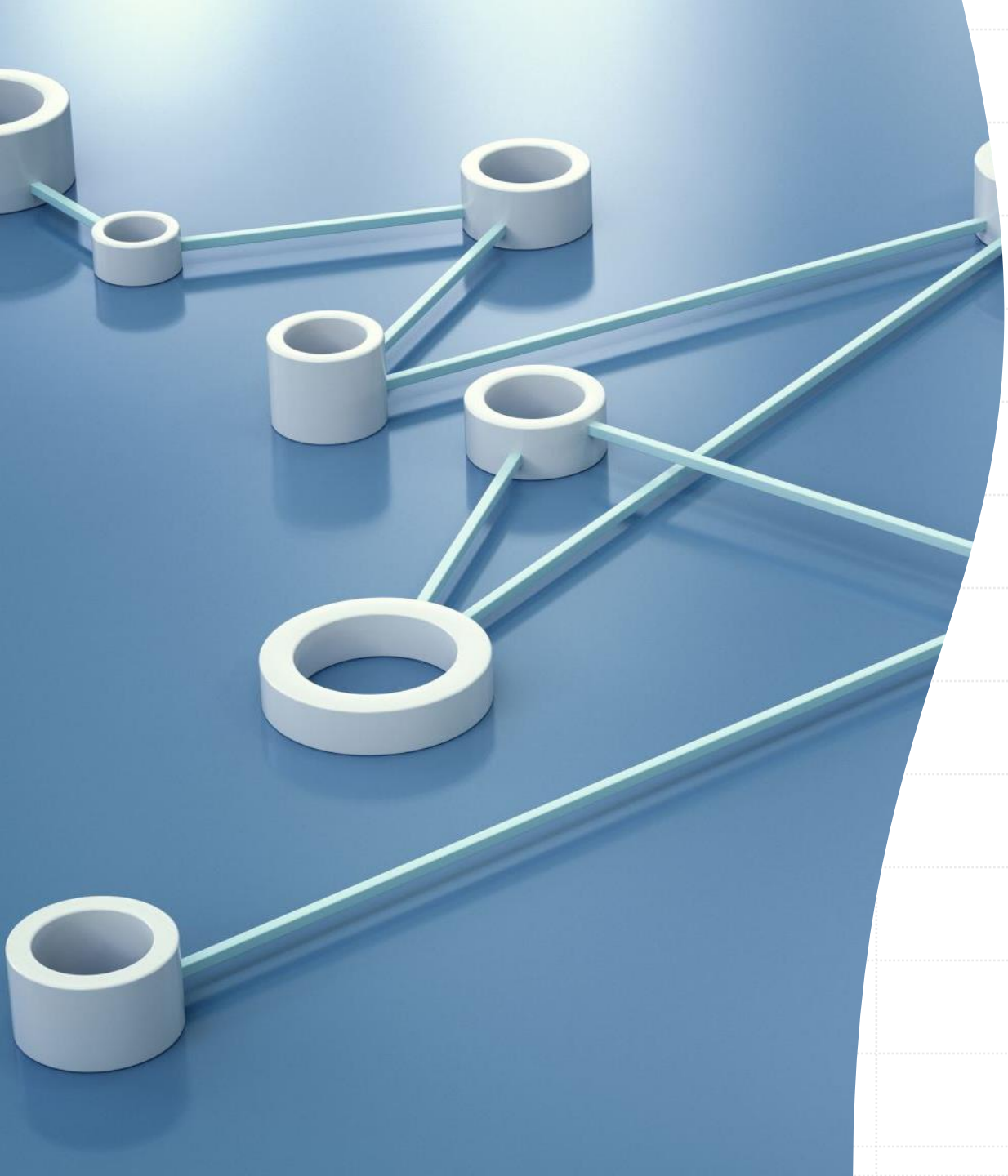


The client is a device or program that sends requests to the server, which is a system or application that responds to those requests.

The server typically provides some kind of service, such as delivering data, processing information, or handling operations.

Key Components of Client-server Model

- Client:
  - The client is any device or application that makes requests for services or resources from a server.
  - Common clients include web browsers (Chrome, Firefox, Safari), mobile apps, and email clients.
- Server:
  - The server is a machine or program that waits for requests from clients and then processes those requests to provide the required services.
  - A server can host various services, such as websites (web servers), email services (email servers), file storage (file servers), and more.



Key Components of Client-server Model

- Communication Protocol:
  - The communication between the client and the server is usually governed by specific protocols.
  - For example, HTTP/HTTPS for web browsing, FTP for file transfers, or SMTP for email communication.
  - These protocols define how the data is structured, transmitted, and received.

Examples of the Client-Server Model

- Web Browsing
  - When you open a website, your browser (client) sends a request to a web server.
  - The server processes the request and returns the webpage, which is displayed in the browser.
- Email
  - Your email client (Outlook, Gmail, etc.) sends requests to the email server, which retrieves or sends email messages.
- Database Access
  - A database client (e.g., an application) requests data from a database server, which processes the query and returns the data.

The Role of Web Browsers

- Request Handling:
  - Browsers act as clients, sending HTTP/HTTPS requests to web servers when users access a URL or click on a link.
  - They request resources such as HTML documents, CSS stylesheets, JavaScript files, images, and multimedia.
- Rendering Content:
  - Browsers receive responses from the server and render the content for display.
  - This involves interpreting HTML for structure, CSS for design, and JavaScript for interactivity.

The Role of Web Browsers

- User Interaction:
  - Browsers allow users to interact with websites through forms, buttons, and other elements.
  - These interactions often result in new requests to the server for data or actions.
- Developer Tools:
  - Modern browsers provide built-in tools (e.g., Chrome DevTools, Firefox Developer Tools) to debug and analyze web pages, aiding in development and troubleshooting.

The Role of Web Servers

- Resource Hosting
  - Servers store and manage the files required to display a website, such as HTML, CSS, and JavaScript files.
- Request Processing
  - Servers receive HTTP/HTTPS requests, process them, and send appropriate responses back to the client.

The Role of Web Servers

- Middleware Integration
  - In dynamic websites, servers act as intermediaries, handling requests, querying databases, and returning results.
- Security and Performance
  - Servers enforce security protocols like HTTPS, ensuring encrypted communication.
  - They optimize performance using caching, load balancing, and compression techniques.

Thank you