Module I: Introduction to Web

# Syllabus

MODULE I: Introduction To Web

Client/Server concepts, Overview of HTTP - HTTP request – response, Generation

of dynamic web pages, Application Servers, Web Security.

1.1 Introduction To Web: Client/Server concepts 1

1.2 Overview of HTTP - HTTP request – response 1

1.3 Generation of dynamic web pages 1

1.4 Application Servers, Web Security. 1

# Introduction to Web

## Web

The term "web" typically refers to the World Wide Web (www), which is a system of interlinked hypertext documents accessed via the internet. With a web browser, you can view web pages that may contain text, images, videos, and other multimedia and navigate between them using hyperlinks. The web was invented in 1989 by Tim Berners-Lee, a British scientist at CERN. It's important to distinguish the web from the internet; the internet is the network of connected computers globally, while the web is a collection of information accessed via the internet.

## Internet

The internet is a vast global network of computers and other electronic devices connected to each other. It allows devices to communicate and share information with each other using standardized communication protocols, the most common being the Internet Protocol (IP). Here are some key aspects of the internet:

1. **Global Network**: The internet connects millions of private, public, academic, business, and government networks from around the world.
2. **Information and Resource Sharing**: It enables the sharing of information and resources, such as web pages, documents, and multimedia content.
3. **Communication**: Email, instant messaging, voice over IP (VoIP), and video conferencing are some of the communication services facilitated by the internet.
4. **Data Transmission**: Data is transmitted over the internet in the form of packets using the Internet Protocol. This transmission can be done through various types of networks including LANs, WANs, and WLANs.
5. **World Wide Web**: One of the most popular services on the internet, the World Wide Web (www) is a collection of interconnected documents and other resources, linked by hyperlinks and URLs.
6. **Protocols**: The operation of the internet relies on a set of protocols, with the TCP/IP (Transmission Control Protocol/Internet Protocol) suite being the most fundamental.
7. **Services**: The internet provides numerous services such as the web, email, file sharing, online gaming, and social networking.
8. **Infrastructure**: It consists of physical components like cables, routers, servers, and wireless networks, as well as software and protocols that manage network traffic.
9. **Accessibility**: It's accessible to billions of users worldwide through various devices, including computers, smartphones, and tablets.

The development and implementation of the internet have had a profound impact on society, transforming the way people communicate, access information, and conduct business. It's an essential tool for globalization, education, and innovation, and continues to evolve with new technologies and applications.

## Websites

A website is a collection of related web pages that are typically identified with a common domain name and published on at least one web server. Examples include wikipedia.org, google.com, and amazon.com. Websites can be accessed via a public internet protocol (IP) network, such as the internet, or a private local area network (LAN), by referencing a uniform resource locator (URL) that identifies the site.

Websites serve a variety of purposes and can be used for various functions. Some of the common types include:

1. **Personal websites**: Created by individuals to share personal information, opinions, or hobbies with a broader audience.
2. **Commercial websites**: Used by businesses for promoting and selling products or services.
3. **Government websites**: Provide information about government policies, programs, and services.
4. **Nonprofit organization websites**: Used to promote a cause, provide information, and facilitate engagement with the public.
5. **Educational websites**: Offer educational material and resources, often associated with educational institutions.
6. **News websites**: Provide reporting on current events, investigative journalism, and other news-related material.
7. **Social media websites**: Enable users to connect and interact with each other through posting text, photos, and videos.
8. **Forums and community websites**: Allow users to interact via posting messages, comments, and participating in discussions.
9. **Blogs**: Regularly updated websites or web pages, typically run by an individual or small group, focused on written content.

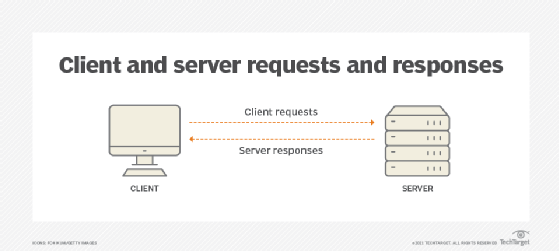
A website is typically hosted on a server and is accessible via a network such as the internet or a private local area network through an internet address known as a URL. Modern websites are usually built using a combination of HTML, CSS, and JavaScript, and they can range from simple, static pages to complex, dynamic web applications.

## Client/Server concepts

The client/server concept is a fundamental model for networked communication, especially prevalent in the context of the internet. This model divides network communication into two distinct roles: the client and the server. Here's an overview of how this concept applies to the internet:

1. **Server:** A server is a computer or computer program that provides a service by responding to requests across a network. In the context of the internet, servers host websites, manage email, handle databases, and provide numerous other services. They are equipped to handle requests from multiple clients simultaneously and operate continuously to ensure availability of their services.
2. **Client**: A client is a computer or software application that requests services or resources from a server. On the internet, this often takes the form of web browsers, email clients, or other applications that users interact with to access remote services or data.
3. **Communication Protocol**: The communication between clients and servers on the internet is governed by specific protocols. For example, the Hypertext Transfer Protocol (HTTP) is used for web communication, Simple Mail Transfer Protocol (SMTP) for sending emails, and File Transfer Protocol (FTP) for transferring files.
4. **Request-Response Cycle**: The interaction typically follows a request-response pattern. The client sends a request to the server (like requesting a web page), and the server processes this request and sends back the appropriate response (such as the HTML content of the web page).
5. **IP Addresses and DNS**: Each server on the internet is identified by a unique IP address. The Domain Name System (DNS) translates human-readable domain names (like [www.example.com](http://www.example.com/)) into IP addresses that computers use to locate each other on the network.
6. **Stateless Communication**: Many client-server interactions on the internet are stateless, meaning each client request is treated as an independent transaction that is unrelated to any previous request. This is a fundamental characteristic of HTTP.
7. **Scalability and Load Balancing**: To handle a large number of client requests, servers may be scaled up and load balancing may be used. This involves distributing the workload across multiple servers to ensure efficient handling of client requests and high availability.
8. **Security**: In client-server communication, security is crucial. Protocols like HTTPS (HTTP Secure) and various authentication mechanisms are used to ensure safe data transmission and access control.

In summary, the client/server model is a cornerstone of how the internet operates, enabling the efficient and structured exchange of information across the global network. It allows different computers and programs to communicate in a standardized way, facilitating a wide range of internet services.



## HTTP Request and Response

HTTP (Hypertext Transfer Protocol) is the foundation of data communication on the World Wide Web. It operates as a request-response protocol between a client and a server. Here's a basic overview of the HTTP request and response process:

### HTTP Request

* Initiation by the Client: An HTTP request is initiated by a client, usually a web browser or a web application, when a user tries to access a webpage or resource on the internet.
* Components of an HTTP Request:
  + Method: Indicates the desired action to be performed (e.g., GET to retrieve data, POST to submit data).
  + URL (Uniform Resource Locator): Specifies the location of the resource on the server (e.g., https://www.example.com/page.html).
  + HTTP Version: Indicates the HTTP version being used (e.g., HTTP/1.1).
  + Headers: Provide additional information about the request (like User-Agent, Accept, Host, etc.).
  + Body: In some methods like POST, the body contains data sent to the server (like form data).

### HTTP Response

* Processing by the Server: The server receives the HTTP request, interprets it, and takes the appropriate action, such as retrieving the requested web page.
* Components of an HTTP Response:
  + Status Line: Contains the HTTP version, a status code (e.g., 200 for success, 404 for not found), and a status message.
  + Headers: Similar to request headers, they provide information about the server and the response (like Content-Type, Content-Length, Server, etc.).
  + Body: Contains the actual data requested, such as an HTML page, image, or file.

### Process Flow

1. User Action: A user clicks a link or enters a URL in a web browser.
2. Browser Sends Request: The browser sends an HTTP request to the server.
3. Server Processes Request: The server processes the request and generates an HTTP response.
4. Server Sends Response: The response is sent back to the client.
5. Browser Displays Result: The browser interprets the response and displays it to the user, which might be a webpage, an error message, or another type of content.

### Important Notes

* Statelessness: HTTP is a stateless protocol, meaning each request-response pair is independent; servers do not retain session information.
* Secure HTTP (HTTPS): HTTPS is the secure version of HTTP, which encrypts the request and response to prevent eavesdropping and tampering.

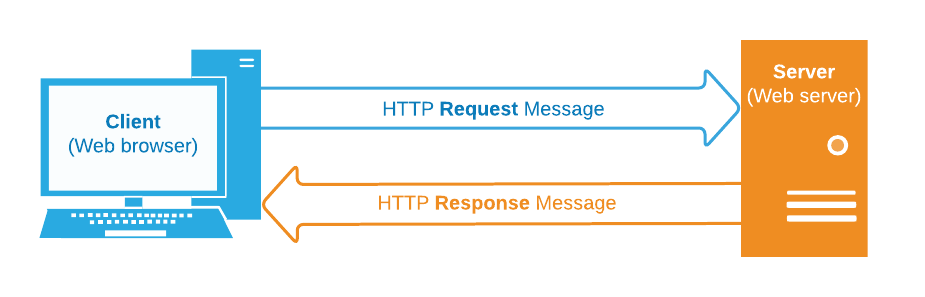
This request-response pattern is fundamental to the functionality of the web, allowing web browsers and servers to communicate efficiently.

### HTTP Vs. HTTPS

HTTP (Hypertext Transfer Protocol) and HTTPS (HTTP Secure) are both protocols used for transmitting data over the web, but they differ significantly in terms of security:

1. HTTP (Hypertext Transfer Protocol):
   * Unsecured: HTTP is an unsecured protocol. Data transferred via HTTP is not encrypted, which means it's vulnerable to eavesdropping and man-in-the-middle attacks.
   * Port 80: By default, HTTP uses port 80 for communication.
   * Usage: Initially, HTTP was more commonly used, especially for websites that did not require encryption. However, its usage has declined in favor of HTTPS.
2. HTTPS (HTTP Secure):
   * Encrypted Transmission: HTTPS is a secure version of HTTP. It uses SSL (Secure Sockets Layer) or TLS (Transport Layer Security) to encrypt data transmissions. This encryption ensures that data transferred between the user and the website is secure and cannot be easily intercepted or tampered with.
   * Port 443: HTTPS typically uses port 443 for communication.
   * Security Indicators: Browsers often indicate HTTPS connections with a padlock icon or a green address bar, signaling to users that their connection is secure.
   * Usage: HTTPS is essential for any website that handles sensitive data, like financial transactions, login information, and personal data. Due to increasing concerns about privacy and security, many websites have migrated to HTTPS, even those that do not handle sensitive information.
3. Key Differences:
   * Security: The main difference is the level of security - HTTPS provides a secure channel over an insecure network, while HTTP does not.
   * Performance: Historically, HTTPS was considered slower than HTTP due to the overhead of encryption. However, with modern computing power and optimized protocols, this difference is now negligible.
   * SEO and Trust: Search engines like Google favor HTTPS websites, and users are more likely to trust and complete transactions on HTTPS sites.
4. Transition to HTTPS:
   * In recent years, there has been a significant push to adopt HTTPS to ensure a more secure and private internet. Tools like Let's Encrypt offer free SSL/TLS certificates to promote the widespread adoption of HTTPS.

In summary, the primary difference between HTTP and HTTPS lies in security. HTTPS provides a secure, encrypted connection, making it the preferred choice for web communications, particularly where privacy and data protection are of paramount concern.

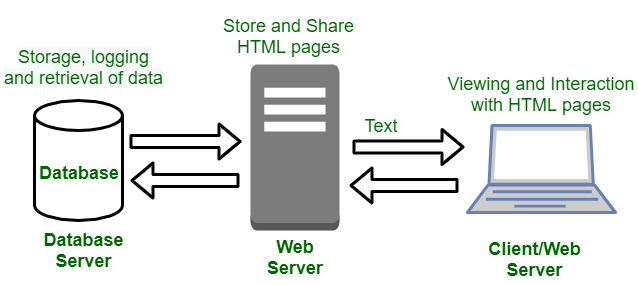


## Generation of Dynamic Web Pages

Dynamic web pages are created in real-time, often based on user input or other factors, and can change each time they are accessed. This contrasts with static web pages, which are the same every time they are loaded. Here's a brief overview of how dynamic web pages are generated:

1. **Client-Server Interaction**: The process typically starts with a user making a request to a web server for a specific page. This request might include user input, like a search query.
2. **Server-Side Processing**:
   * Web Server: The server receives the request and determines what needs to be done to generate the appropriate content.
   * Server-Side Scripts: The server runs scripts written in languages such as PHP, Python, Ruby, or Java. These scripts are used to create the content of the page. They can access databases, files, or other resources to gather the necessary data.
   * Database Interaction: For many dynamic pages, a database query is required. The server-side script sends queries to a database to retrieve or update data.
3. **Content Generation**:
   * Assembling the Page: Based on the script's logic and the database's response, the server assembles the HTML of the web page. This HTML will include the dynamic content that was just generated.
   * Sending the Response: The server then sends this HTML back to the client as the response to the original request.
4. **Client-Side Rendering**:
   * The client's web browser receives the HTML and renders it, displaying the dynamic web page to the user.
   * Client-side scripts, like JavaScript, can also play a role in making a page dynamic. After the page is loaded, JavaScript can make additional server requests (using AJAX, for instance), update the page's content in real-time, and create interactive components.
5. **Caching and Updating**:
   * Dynamic pages can be cached for efficiency, but they often need mechanisms to ensure that the user sees the most up-to-date information.

This dynamic generation allows for personalized, interactive, and frequently updated web pages, which are essential for modern web experiences like online shopping, social media, and content management systems.

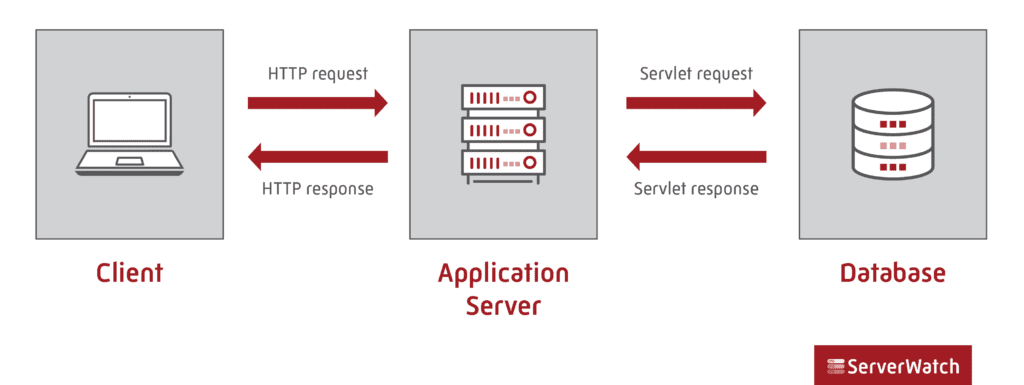


## Web Application Servers

Web application servers are specialized servers designed to host and serve web applications. They provide the necessary infrastructure and services to run web-based applications, handling complex backend tasks such as session management, database interaction, and execution of application logic. Here’s a brief overview:

1. Role and Functionality:
   * Web application servers act as a middle layer between the front-end user interface (web browser) and the backend resources like databases and file systems.
   * They process incoming requests from clients (typically through a web browser), execute the required application logic, and then send the appropriate responses (often as HTML pages).
2. Key Features:
   * Session Management: They manage user sessions, maintaining state information over the stateless HTTP protocol.
   * Security: They implement security measures like authentication, authorization, and secure communication.
   * Database Connectivity: Provide mechanisms to connect to databases, retrieve data, and update data, facilitating dynamic content generation.
   * Scalability and Load Balancing: Capable of handling multiple concurrent requests and can be scaled to manage increased loads.
3. Server-Side Scripting:
   * They support server-side scripting languages such as PHP, Python, Java, Ruby, and .NET for executing application logic.
   * Scripts or compiled code run on the server to process the logic of the web application.
4. Frameworks and Tools:
   * Often associated with specific development frameworks and tools that simplify web application development (like Ruby on Rails, Django, ASP.NET, etc.).
5. Deployment and Hosting:
   * They can be hosted on-premises or on cloud platforms.
   * Support for containers and virtualization allows for flexible deployment and scalability.
6. Types and Examples:
   * Examples include Apache Tomcat, Microsoft's Internet Information Services (IIS), and NGINX for PHP or Java applications.
   * Some are specialized for particular programming languages or environments.
7. Integration with Other Components:
   * Often integrated with web servers (like Apache or NGINX) that handle the delivery of static content, while the application server focuses on executing application logic.

In summary, web application servers are crucial for the development and deployment of dynamic, interactive, and scalable web applications, offering a range of functionalities necessary for modern web services.



## Web Security

Web security refers to the protective measures and protocols implemented to safeguard information and services over the web from unauthorized access, use, disclosure, disruption, modification, or destruction. Given the vast amount of sensitive data and transactions conducted online, web security is critical for maintaining the integrity, confidentiality, and availability of web resources. Here's a brief overview:

1. Threats to Web Security:
   * Malware: Malicious software designed to harm or exploit any programmable device, service, or network.
   * Phishing Attacks: Deceptive attempts to obtain sensitive information by masquerading as a trustworthy entity.
   * Cross-Site Scripting (XSS): Injecting malicious scripts into otherwise benign and trusted websites.
   * SQL Injection: Exploiting vulnerabilities in a web application's database layer by injecting malicious SQL queries.
   * Denial of Service (DoS)/Distributed Denial of Service (DDoS) Attacks: Overwhelming a server or network with traffic to make it unavailable to its intended users.
2. Protective Measures:
   * HTTPS (SSL/TLS): Encrypting data in transit between the client and the server to prevent data interception and tampering.
   * Firewalls: Filtering incoming and outgoing traffic based on a set of security rules to block unauthorized access.
   * Anti-Malware Software: Protecting against malware through detection, prevention, and removal tools.
   * Regular Updates and Patching: Keeping all software and systems up to date to protect against known vulnerabilities.
3. Best Practices:
   * Strong Authentication Mechanisms: Using multi-factor authentication, strong password policies, and secure password recovery methods.
   * Input Validation: Ensuring that all input is validated, sanitized, and escape to prevent XSS, SQL injection, and other injection attacks.
   * Access Control: Implementing proper access controls to ensure that users can only access the resources that they are authorized to use.
   * Regular Security Audits and Monitoring: Regularly auditing web applications for vulnerabilities and continuously monitoring for suspicious activities.
4. Data Protection and Privacy:
   * Encryption of Sensitive Data: Both in transit (using HTTPS) and at rest.
   * Privacy Policies and Compliance: Adhering to legal requirements and standards regarding user data privacy (like GDPR, HIPAA).
5. Awareness and Training:
   * Educating users and administrators about the latest threats and best practices in web security.

In summary, web security encompasses a broad range of practices, tools, and strategies designed to protect websites, web applications, and online services from threats and vulnerabilities. It's a critical aspect of managing web resources, especially in an era where cyber threats are continuously evolving and increasing in sophistication.