


Topic 7

Implementing and Managing a Web Server

CST4013 | Website Designing





Learning Outcomes

1

Describe the concept of web server and its implementation.

2

Explain the web server processes and architecture.



Web Server

- A web server is a software application or program that runs on a computer (either physical or virtual) and is responsible for serving web content to clients over the internet.
- It processes requests from web clients, such as web browsers, and delivers web pages, files, and other resources in response.

A physical web server refers to a physical computer system dedicated to hosting and serving websites or web applications over the internet.

A virtual web server, also known as a virtual private server (VPS), is a virtualized server environment created by partitioning a physical server into multiple virtual servers.

Web Server Processes

Client Request

- When a user enters a URL in a web browser, the browser generates an HTTP request for that URL.
- This request is sent over the internet to the web server hosting the website.

DNS Resolution

- Before the request reaches the web server, it goes through the Domain Name System (DNS) to resolve the domain name (e.g., www.example.com) into an IP address.
- This IP address points to the server where the website is hosted.

TCP/IP Connection

- The client (web browser) establishes a TCP/IP connection with the web server using the resolved IP address.
- The connection is typically made to port 80 for HTTP or port 443 for HTTPS.

Web Server Processes

Request Handling

- The web server listens for incoming requests on the specified port.
- When it receives a request, it parses the request to determine what is being requested (e.g., a specific HTML page, an image, a script).

Processing the Request

- The web server processes the request by mapping the requested URL to the corresponding file or resource on the server.
- If the request is for a static file (like an HTML page, CSS file, or image), the server retrieves the file from its storage.

Generating the Response

- Once the requested resource is retrieved or generated, the web server constructs an HTTP response.
- This response includes status information (such as 200 OK, 404 Not Found), headers (metadata about the response), and the actual content.



Web Server Processes



Sending the Response

- The web server sends the HTTP response back to the client over the established TCP/IP connection.
- The client (web browser) receives the response and processes it accordingly.



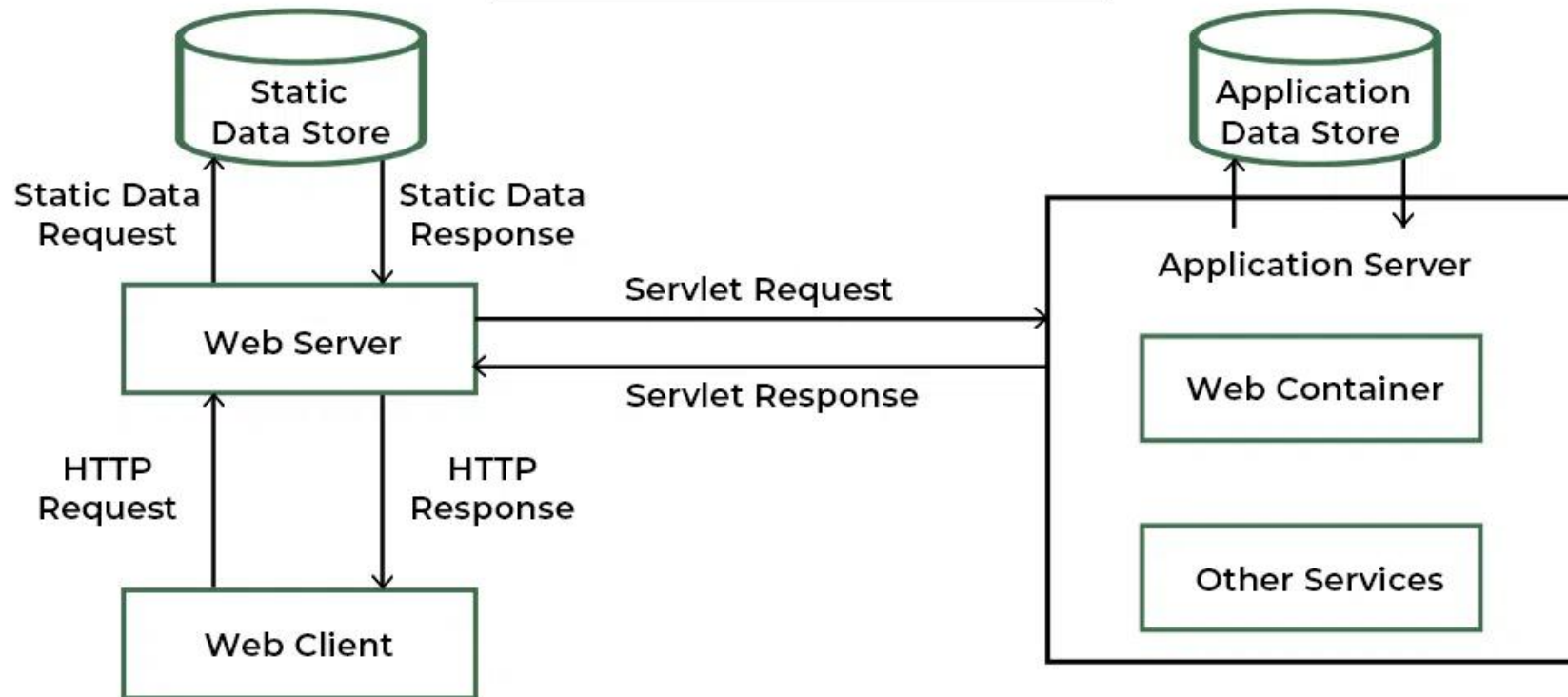
Rendering the Contentt

- The web browser interprets the HTML, CSS, JavaScript, and other resources received in the response and renders the web page for the user to view and interact with.

Web Server Processes



Working of Web Server





Web Server Components

Web Server Software

- Examples include Apache, Nginx, Microsoft IIS, and LiteSpeed.
- This software handles incoming requests, processes them, and serves the appropriate responses.

Operating System

- The web server runs on an operating system (e.g., Linux, Windows) that manages hardware resources and provides the necessary environment for the web server software.



Web Server Components

Application Server (for dynamic content)

- When dynamic content is requested, the web server may rely on an application server (e.g., Apache Tomcat, Node.js) or language-specific interpreter (e.g., PHP, Python) to process the request and generate the response.

Database Server

- For data-driven web applications, the web server may interact with a database server (e.g., MySQL, PostgreSQL, MongoDB) to retrieve or store data.



Web Server Architecture

- Web server architecture refers to the structure and organization of components that enable a web server to handle client requests, process data, and serve content efficiently.
- There are different architectural models for web servers, each suited to various scales, performance requirements, and types of applications.



Web Server Architecture

Web Server Architecture follows the following approaches:

- 1** Single-Tier Architecture (Monolithic)
- 2** Multi-Tier Architecture
- 3** Microservices Architecture



Web Server Architecture

Web Server Architecture follows the following approaches:

4

Serverless Architecture

5

Clustered Architecture



Single-Tier Architecture

- A single server handles all aspects of the web application, including the web server, application logic, and database.

Advantages	Disadvantages
<ul style="list-style-type: none">• Simplicity in setup and management.• Cost-effective for small-scale applications.	<ul style="list-style-type: none">• Limited scalability.• Single point of failure.

Multi-Tier Architecture

- Divides the application into three layers:
 - Presentation Layer: The user interface, usually the web server.
 - Application Layer: The business logic, often an application server.
 - Data Layer: The database server.

Advantages	Disadvantages
<ul style="list-style-type: none">• Separation of concerns improves maintainability.• Scalability by adding or upgrading individual layers.	<ul style="list-style-type: none">• Increased complexity compared to single-server setups.• Potential latency between layers.

Microservices Architecture

- The application is divided into small, loosely coupled services, each responsible for a specific functionality and can be developed, deployed, and scaled independently.

Advantages	Disadvantages
<ul style="list-style-type: none">• High scalability and flexibility.• Independent development and deployment of services.	<ul style="list-style-type: none">• Complexity in managing and orchestrating multiple services.• Potential for increased latency and network communication overhead.

Serverless Architecture

- Utilizes cloud services to run application code in response to events.
- The cloud provider manages the infrastructure, scaling, and server management.

Advantages	Disadvantages
<ul style="list-style-type: none">• Reduced operational overhead.• Automatic scaling.• Cost-effective as billing is based on actual usage.	<ul style="list-style-type: none">• Cold start latency.• Vendor lock-in risks.• Debugging and monitoring can be challenging.






Clustered Architecture




- Multiple servers (nodes) work together as a single system to provide high availability and load balancing.
- Often used in combination with a load balancer.

Advantages	Disadvantages
<ul style="list-style-type: none">• High availability and redundancy.• Scalability by adding more nodes.	<ul style="list-style-type: none">• Increased complexity in setup and management.• Requires sophisticated load balancing and failover mechanisms.

Type of Web Servers

Type of Servers	Description
Apache HTTP Server 	<ul style="list-style-type: none">• Apache is known for its stability and flexibility, making it a popular choice for hosting a wide range of websites.• It's open-source and highly customizable, supporting various modules for extending its functionality.
Nginx 	<ul style="list-style-type: none">• Nginx is another popular open-source web server known for its high performance and scalability.• It's often used as a reverse proxy server, load balancer, and HTTP cache in addition to serving web content.
Microsoft Internet Information Services (IIS) 	<ul style="list-style-type: none">• IIS is a web server created by Microsoft for hosting websites and web applications on Windows servers.• It's tightly integrated with other Microsoft technologies like ASP.NET

Type of Web Servers

Type of Servers	Description
<p>LiteSpeed Web Server</p> 	<ul style="list-style-type: none">• LiteSpeed is a commercial web server known for its high performance and low resource usage.• It's compatible with Apache configurations, making it easy to switch from Apache to LiteSpeed without significant modifications.
<p>OpenLiteSpeed</p> 	<ul style="list-style-type: none">• OpenLiteSpeed is the open-source version of LiteSpeed Web Server.• It offers similar features and performance benefits as LiteSpeed but is freely available for use.
<p>Node.js</p> 	<ul style="list-style-type: none">• Node.js is a runtime environment that allows JavaScript to be executed server-side.• Developers can create web servers using Node.js using frameworks like Express.js

Webmaster

- A webmaster is a person responsible for maintaining, managing, and overseeing websites.
- This role can encompass a wide range of tasks, from technical aspects to content management, security, and user experience.





Webmaster Responsibilities

Website Maintenance and Updates

- Regularly updating website content to keep it current and relevant.
- Ensuring that all links, images, and other media function correctly.
- Managing CMS (Content Management System) updates and ensuring compatibility with themes and plugins.

Technical Management

- Overseeing the hosting environment, including server setup and configuration.
- Implementing and maintaining SSL certificates for secure data transmission.
- Monitoring website performance and implementing optimizations for speed and reliability



Webmaster Responsibilities

Security

- Protecting the website from cyber threats by implementing security measures such as firewalls, anti-malware tools, and regular backups.
- Keeping software, plugins, and themes up to date to mitigate vulnerabilities.
- Managing user access and permissions to ensure secure content management.

SEO (Search Engine Optimization)

- Optimizing website content for search engines to improve visibility and ranking.
- Conducting keyword research and implementing SEO best practices.
- Analyzing traffic and performance data to make informed SEO decisions.



Webmaster Responsibilities

User Experience (UX) and Design

- Ensuring the website is user-friendly and provides a positive experience for visitors.
- Implementing responsive design practices to ensure the site works well on various devices and screen sizes.
- A/B testing and usability testing to identify and fix UX issues.

Content Management

- Coordinating with content creators to ensure timely publication of new content.
- Managing the organization and structure of content on the site for easy navigation and accessibility.
- Ensuring all content aligns with the brand's voice and standards.



Webmaster Responsibilities

Analytics and Reporting

- Using tools like Google Analytics to track website traffic and user behavior.
- Generating reports on key metrics such as page views, bounce rates, and conversion rates.
- Using data insights to drive website improvements and strategy.

Compliance and Legal

- Ensuring the website complies with legal requirements such as GDPR, ADA, and other regulations.
- Managing privacy policies and terms of service.



Thank you