# IN1621

## INTRODUCTION TO WEB BASED SYSTEMS, ARCHITECTURES AND PROTOCOLS

# What is the Internet ?

- Internet is the technical infrastructure which allows billions of computers to be connected  with each other.
- Internet is a large network of computers which communicate with each other.
- The Internet is the backbone of the Web.
- Internet  makes the rapid exchange information between computers across the world possible.

# What is World Wide Web (WWW) ?

The World Wide Web, usually called the Web is a collection of different websites we can access through the Internet.

The building blocks of the Web are web pages which are formatted in HTML and connected by links called "hypertext" or hyperlinks and accessed by HTTP.

Web provides a communication platform for users to retrieve and exchange information over the internet.

World Wide Web we follow a web of hypertext links to visit a web page and from that web page to move to other web pages.

Main authority governing the Web is the W3 Consortium. [www.w3.org](http://www.w3.org)

# What is a Web page ?

A document which can be displayed in a web browser such as Firefox, Google Chrome, Opera, Microsoft Internet Explorer or Edge, or Apple's Safari.

Such documents are written in the HTML language.

A web page can embed a variety of different types of resources such as

*style information* — controlling a page's look-and-feel

*scripts* — which add interactivity to the page

*media* — images, sounds, and videos.

# What is Website ?

- A website is a collection of linked web pages (plus their associated resources) that share a unique domain name.
- Each web page of a given website provides explicit links—most of the time in the form of clickable portion of text—that allow the user to move from one page of the website to another.
- To access a website, type its domain name in your browser address bar, and the browser will display the website's main web page, or homepage.

# Web Browsers

- A web browser retrieves information from the web and displays it on our computer.
- The information is transferred using the Hypertext Transfer Protocol, which defines how text, images and video are transmitted on the web.
- Web standards creates consistency between browsers, so that any user can enjoy the internet, regardless of the browser they choose.
- When the web browser fetches data from an internet connected server and it then uses a piece of software called a rendering engine to translate that data into text and images.

# Web Servers and Bowsers

# Introduction

- Web servers and browsers are the core components of the web.
- Web Browser - The client software used to access web content.
- Web Server - The server software that stores, processes, and delivers web pages to clients.
- Browsers request web pages from servers using HTTP/HTTPS, and servers respond with the requested resources.

# Web Browsers

A web browser is a client application that retrieves, interprets, and displays content from web servers.

**Common Browsers**

Chrome, Firefox, Edge, Safari, Opera.

# Web Browsers

**Key Components of a Browser**

User Interface (UI) – Address bar, back/forward buttons, bookmarks.

Browser Engine – Coordinates between UI and rendering engine.

Rendering Engine – Parses HTML, CSS, and JavaScript to display the page (e.g., Blink, Gecko).

Networking – Handles HTTP/HTTPS requests.

JavaScript Engine – Executes JavaScript code (e.g., V8 in Chrome).

Data Storage – Caches, cookies, local storage, IndexedDB.

# Activity

As pairs discuss what is JIT in Web Browsers.

# Browser Workflow

- User types URL → browser sends HTTP request.
- Server responds with HTML, CSS, JS.
- Browser parses HTML, builds DOM tree.
- CSS applied → render tree constructed.
- JS executed → final layout rendered.

# Web Servers

A web server is software that hosts websites and delivers web content to clients via the web.

"Hosting" means that all the web pages and their supporting files are available on that computer.

The web server will send any web page from the website it is hosting to any user's browser, per user request.

**Popular Web Servers**

Apache HTTP Server

Nginx

Microsoft IIS

LiteSpeed

# Key Functions

- Handle Requests  and receives HTTP/HTTPS requests from browsers.
- Returns static content (HTML, images) or dynamic content via

server-side scripting (PHP, Java, Node.js).

- Logging & Security
  - Tracks requests, manages access, implements SSL/TLS.

# Web Server Workflow

- Browser sends HTTP request.
- Server checks the requested resource.
- If static → serves the file directly.
- If dynamic → runs server-side code (PHP, Python, Java) to generate content.
- Server sends HTTP response → browser renders content.

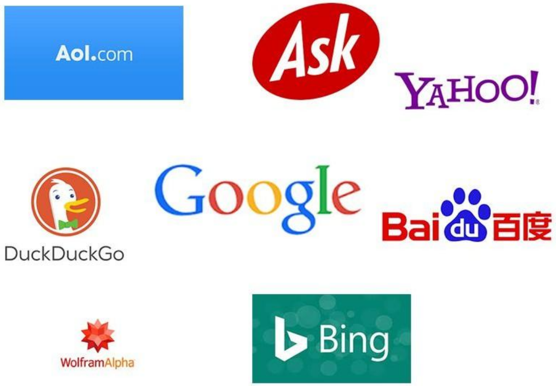
# Client-Server Communication

- Protocol used: HTTP / HTTPS
- Request → Response cycle

| Step | Description |
|---|---|
| Request | Browser sends URL, headers, optional data (POST/GET). |
| Processing | Server retrieves or generates content. |
| Response | Server sends status code, headers, and content. |
| Rendering | Browser parses and displays content. |

# Search Engines

- A web service that helps you find other web pages, such as Google, Bing, Yahoo, or DuckDuckGo.
- Search engines are normally accessed through a web browser (e.g. you can perform search engine searches directly in the address bar of Firefox, Chrome, etc.) or through a web page (e.g. bing.com or duckduckgo.com)



# Web Architectures

# Client - Server Model

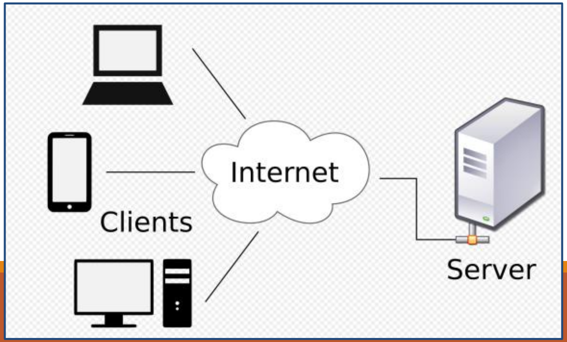
- The Client-server model is a distributed application structure that partitions task or workload between the providers of a resource or service, called servers, and service requesters called clients.
- The client-server model describes how a server provides resources and services to one or more clients.
- Examples of servers include web servers, mail servers, and file servers.
- Each of these servers provide resources to client devices, such as desktop computers, laptops, tablets, and smartphones.
- Web browser is also a client.

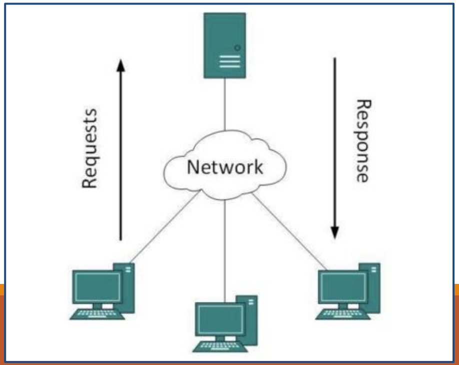
# Client - Server Model

Most servers have a one-to-many relationship with clients, meaning a single server can provide resources to multiple clients at one time.

When a client requests a connection to a server, the server can either accept or reject the connection.

If the request is accepted, the server establishes and maintains a connection with the client over a specific protocol and deliver the data packets requested back to the client.





# 3-tier Architecture

- 3-tier architecture is a software design pattern that separates an application into three logical layers, each with a different responsibility.
- It is commonly used in modern web-based systems and enterprise applications.
- 3-tier architecture splits an application into UI, logic, and database layers, making systems more secure, scalable, and easier to maintain

# The 3 tiers

| Tier | Other Names | Responsibility | Technology Examples |
|---|---|---|---|
| **Presentation Tier** | Front-End / UI | What the user sees and interacts with | HTML, CSS, JavaScript, React, Angular |
| **Logic Tier** | Application Tier / Middle Tier / Business Tier | Processes logic, rules and operations | PHP, Java, C#, Python, Node.js |
| **Data Tier** | Database Tier / Storage | Handles storage, retrieval and data management | MySQL, PostgreSQL, MongoDB, Oracle |

# How the 3 tiers communicate?

- The user interacts with the UI (Presentation Tier)
- UI sends a request to the Logic Tier
- Logic Tier processes the request and communicates with the Data Tier
- Data Tier fetches or stores data
- Logic Tier returns processed results to the UI
- UI displays information to the user

# Microservices Architecture

- Microservices Architecture is an architectural style where an application is built as a collection of small, independently deployable services, each responsible for a specific business function.
- Each microservice runs in its own process.
- Microservices communicate over a network (usually HTTP/REST, gRPC, or messaging queues).
- Emphasizes autonomy, scalability, and resilience.
- Reference: [https://microservices.io/](https://microservices.io/)

# Web Protocols

# What are Web Protocols?

A set of rules that define how data is transmitted over the web.

The language computers use to communicate.

# Why Web Protocols are Important

- INTEROPERABILITY

- SECURITY

- RELIABILITY

- EFFICIENCY

# Common Web Protocols

- HTTP (Port 80)

- HTTPS (Port 443)

- FTP (Ports 20/21)

- SMTP (Port 25/587)

- IMAP & POP3 (Ports 143/110)

- TCP/IP

- WebSocket

- DNS (Port 53)

# HTTP & HTTPS

HTTP: Fetches web pages (not encrypted)

HTTPS: Encrypted version using TLS/SSL

# HTTP

- HyperText Transfer Protocol (HTTP) is an application-level protocol on top of TCP/IP protocol.
- Designed for communication between web browsers and web servers.
- Allows the fetching of resources, such as HTML documents.
- HTTP follows a classical client-server model, with a client opening a connection to make a request, then waiting until it receives a response.
- HTTP is a 4 step process per transaction.

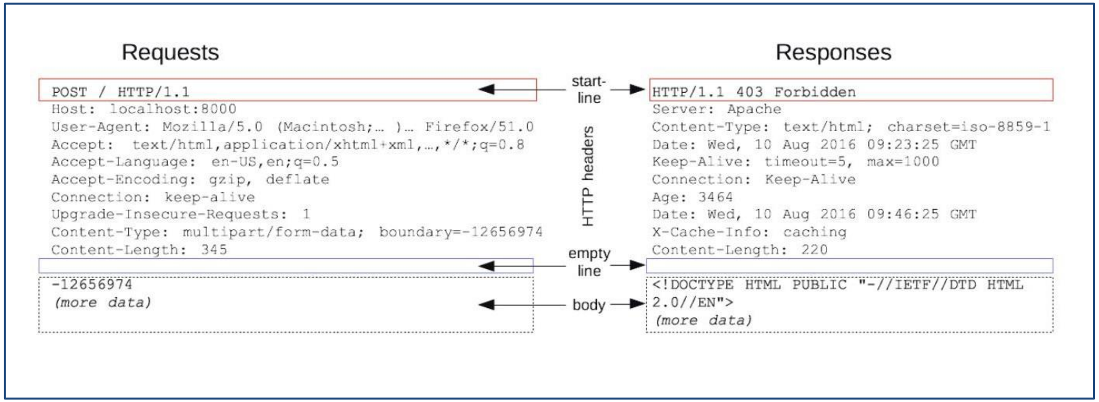
RFC - https://tools.ietf.org/html/rfc7231

# HTTP Flow

1. Client opens a TCP connection.The TCP connection is used to send a request, or several, and receive an answer. The client may open a new connection, reuse an existing connection, or open several TCP connections to the servers.
2. Send an HTTP message
3. Read the response sent by the server
4. Close or reuse the connection for further requests

# HTTP Message

- HTTP messages are how data is exchanged between a server and a client.
- There are two types of messages
  - *requests* sent by the client to trigger an action on the server
  - *responses*, the answer from the server.



Requests

```
POST / HTTP/1.1
Host: localhost:8000
User-Agent: Mozilla/5.0 (Macintosh;… )… Firefox/51.0
Accept: text/html,application/xhtml+xml,…,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=-12656974
Content-Length: 345

-12656974
(more data)
```

Responses

```
HTTP/1.1 403 Forbidden
Server: Apache
Content-Type: text/html; charset=iso-8859-1
Date: Wed, 10 Aug 2016 09:23:25 GMT
Keep-Alive: timeout=5, max=1000
Connection: Keep-Alive
Age: 3464
Date: Wed, 10 Aug 2016 09:46:25 GMT
X-Cache-Info: caching
Content-Length: 220

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML
2.0//EN">
(more data)
```

# HTTP Message

HTTP requests, and responses, share similar structure and are composed of:

1. A *start-line* describing the requests to be implemented, or its status of whether successful or a failure. This start-line is always a single line.
2. An optional set of *HTTP headers* specifying the request, or describing the body included in the message.
3. A blank line indicating all meta-information for the request has been sent.
4. An optional *body* containing data associated with the request (like content of an HTML form), or the document associated with a response. The presence of the body and its size is specified by the start-line and HTTP headers.

# HTTP Methods

HTTP defines a set of request methods to indicate the desired action to be performed for a given resource

- GET - GET is used to request data from a specified resource
- POST - POST is used to send data to a server to create/update a resource
- PUT - PUT is used to send data to a server to create/update a resource
- HEAD - HEAD is almost identical to GET, but without the response body
- DELETE - The DELETE method deletes the specified resource