Web Basics & Request LifeCycle

Youssef Ghatas

Agenda

- Web Basic Concepts
  - IP Address
  - Domain Name
  - DNS
  - GeoDNS
  - HTTP
  - HTTPS
  - CDN
  - Request
  - Response
  - API
  - Frontend (Client Side)
  - Backend (Server Side)
  - Web Server
  - AJAX
  - Web Development Concepts

Concepts

- IP – Port - Domain name
- DNS – GeoDNS
- HTTP/HTTPS

Web Sites evolution

- Web Server?
- FE: HTML – CSS – JS
- BE: Backend

Get & Deploy a server

- Choices?
- How to deploy

Web Browser Storage

- Cookies
- Session Storage
- Local Storage

IP Address & Domain Name

- IP (Internet Protocol) address

157.240.201.35

  - A unique identifier for devices on a network, used to route data between them.

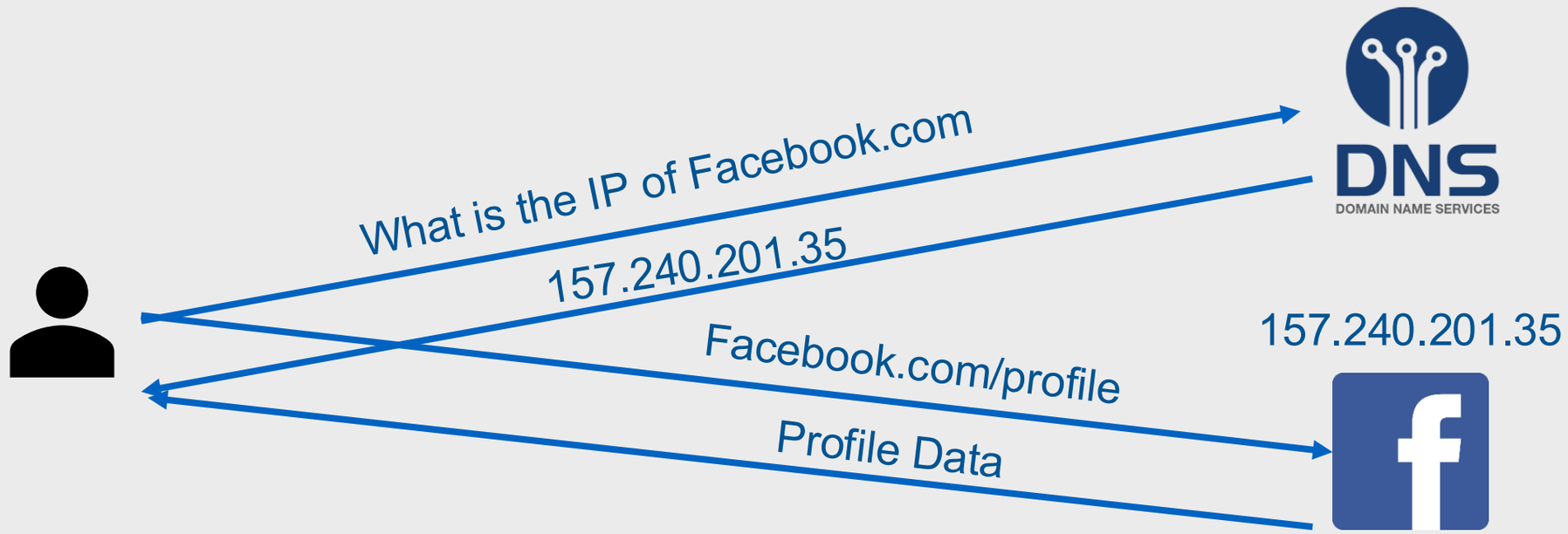
- Domain name

Facebook.com

  - A human-readable address used to access websites or resources on the internet. It provides a user-friendly way to navigate the internet, translating into an IP address for computer network communication.

DNS

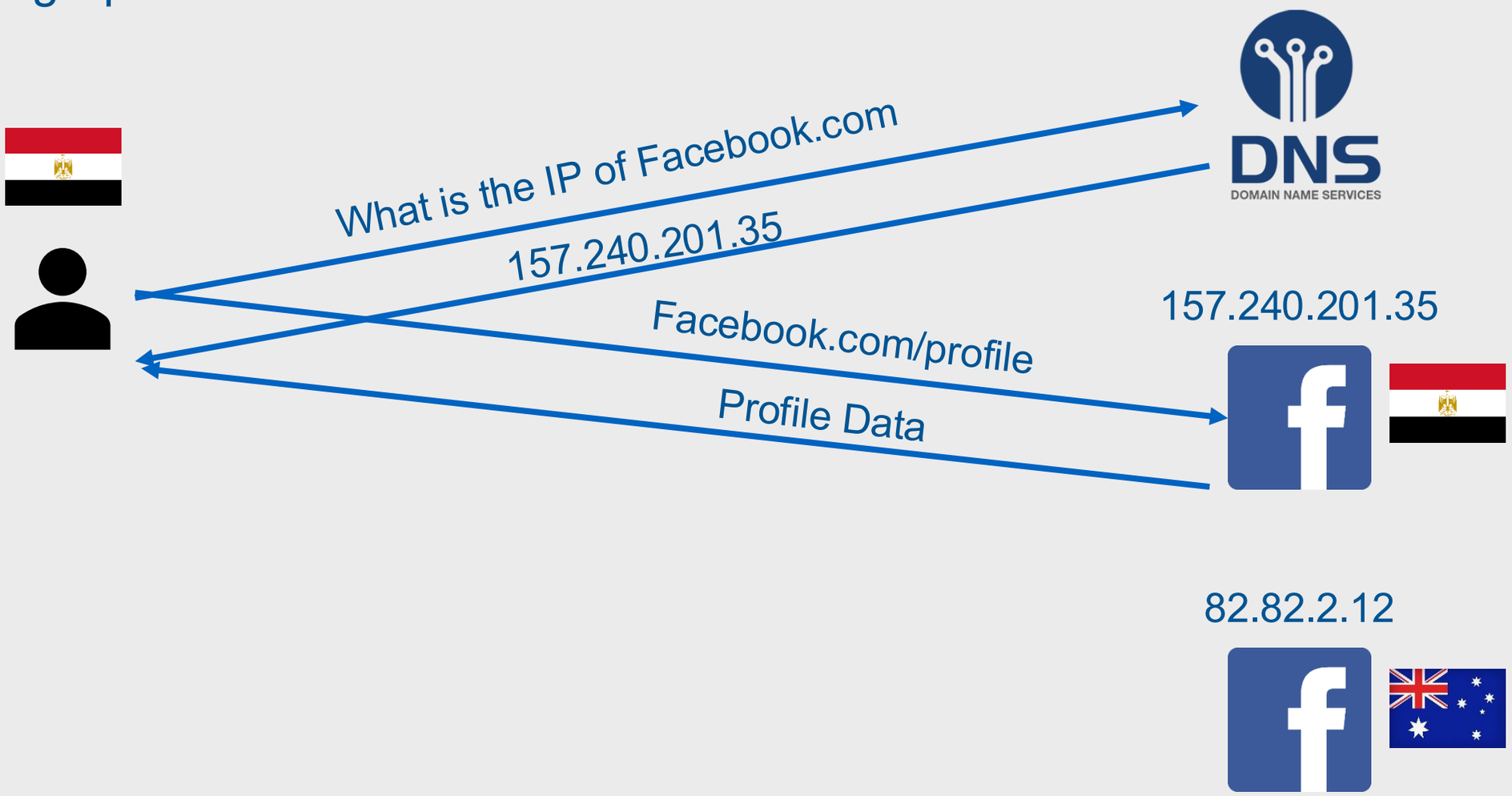
- **DNS (Domain Name System):**
  - naming system for computers, services, or other resources connected to the internet or a private network. It translates human-readable domain names (e.g., facebook.com) into corresponding IP addresses (e.g., 157.240.201.35).



Ex: Google Public DNS - Cloudflare DNS

GeoDNS

- **GeoDNS (Geographical DNS):**
  - Is an extension to DNS that allows different DNS answers based on client's geographical location



Port

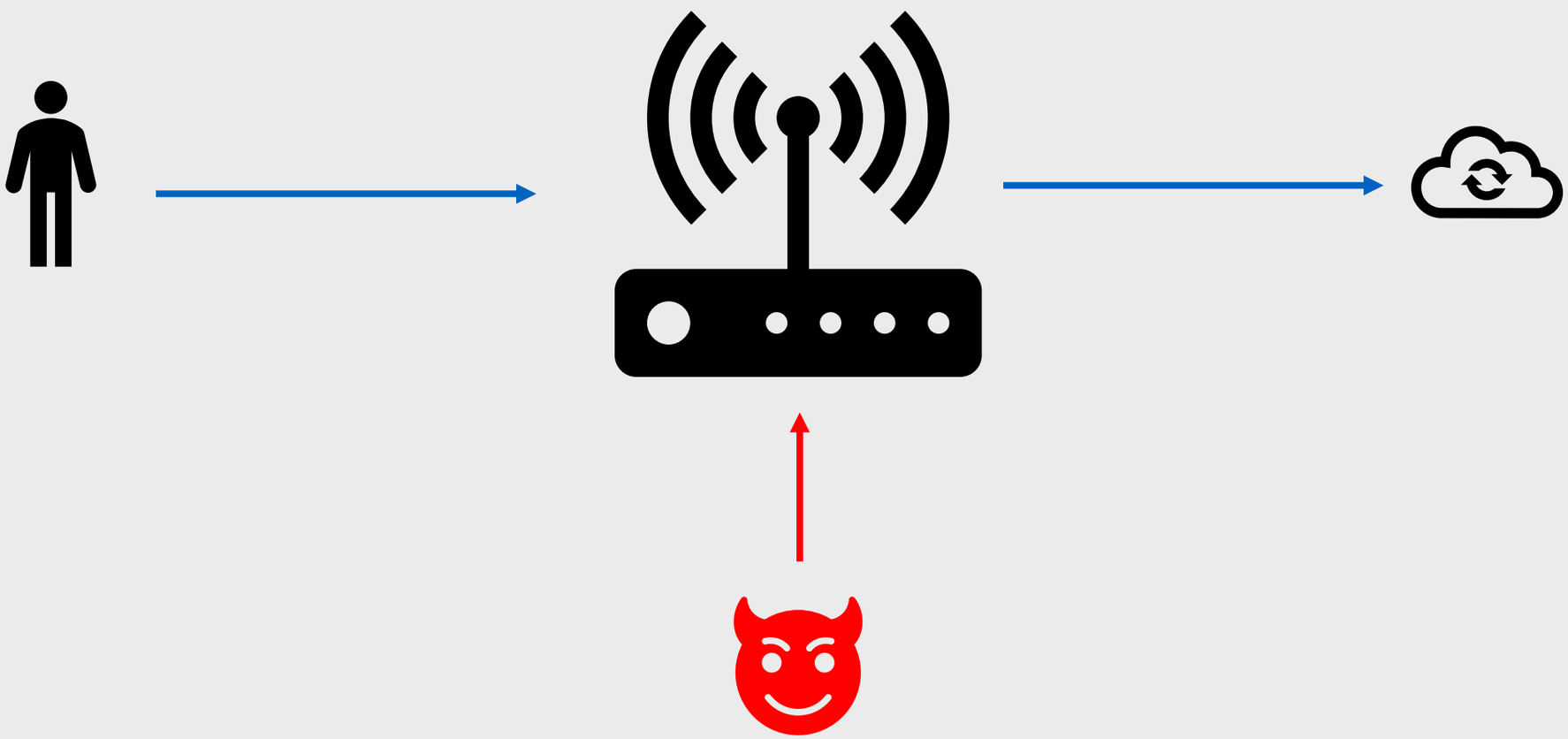
- In addition to the IP, you need a port to connect to the server
- Port: A numbered endpoint on a computer for network communication, allowing multiple services to run simultaneously.
- Ports are identified by numbers ranging from 0 to 65535. Ports enable clients to connect to a particular service on a server using its IP address and port number.

HTTP

- HTTP (**HyperText Transfer Protocol**) is a standard protocol used for transferring and receiving information on the web. It defines how messages are formatted and transmitted, and how web servers and browsers respond to various commands. HTTP operates as a request-response system, where a client (e.g., browser) sends requests for resources, and a server provides responses containing the requested data (e.g., HTML, images) and status information.

- HTTP typically uses TCP/IP (Reliable):
  - Ack
  - Retry
  - Chunks

HTTP Security



HTTPS

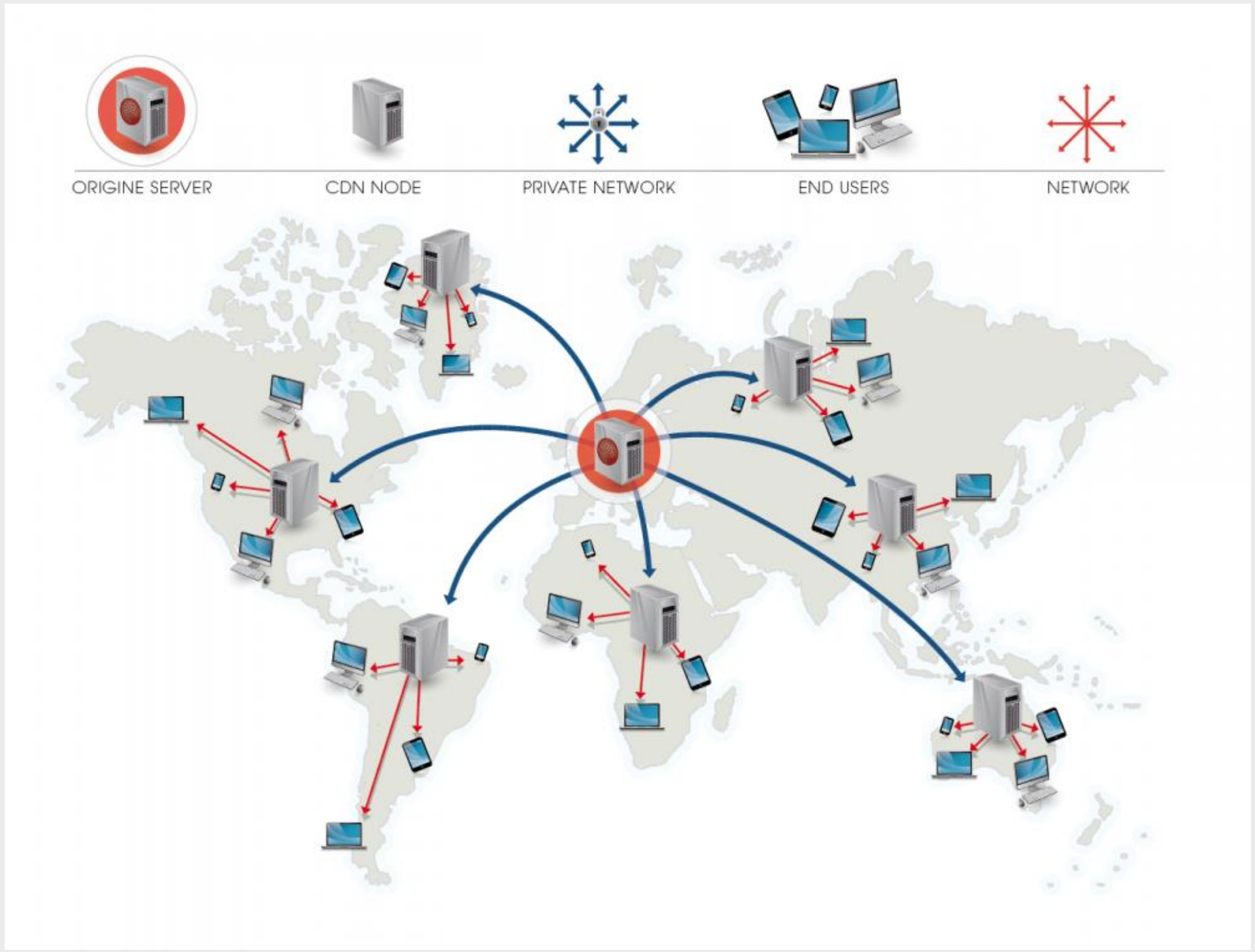
- A secure version of HTTP. It adds encryption so that only the website and the client can read the data sent over the Internet.
- The website encrypt the message with a key that can only be decoded with the user key.
- HTTP uses port 80
- HTTPS uses port 443
- HTTPS needs certificate, how?
  - Free: Certbot (LetsEncrypt)
  - Paid: Why?

CDN

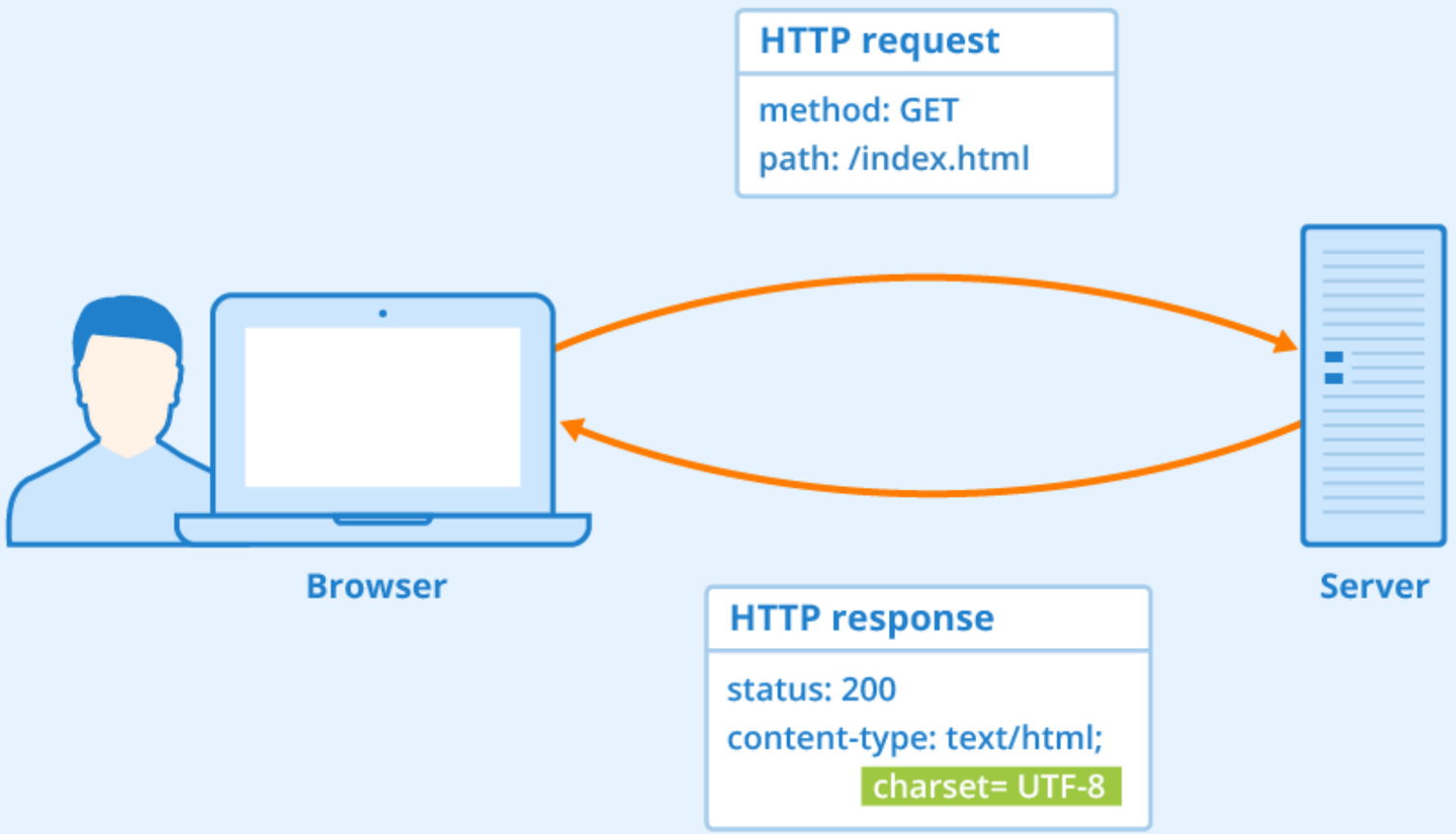
- CDN (**Content Delivery Network**) is a network of distributed servers strategically placed to deliver web content (e.g., images, videos, scripts) to users efficiently by minimizing latency and improving website performance.

Amazon CloudFront – Cloudflare - Microsoft Azure CDN - Google Cloud CDN

CDN



HTTP(S) Request & Response



Author: Seobility - License: CC BY-SA 4.0

Concepts

- IP – Port - Domain name
- DNS – GeoDNS
- HTTP/HTTPS

Web Sites evolution

- Web Server?
- FE: HTML – CSS – JS
- BE: Backend

Get & Deploy a server

- Choices?
- How to deploy

Web Browser Storage

- Cookies
- Session Storage
- Local Storage

Imaginary Evolution of Websites

How can we convert a normal PC to Server

Imaginary Evolution of Websites

We need a tool to convert a normal PC to a server!

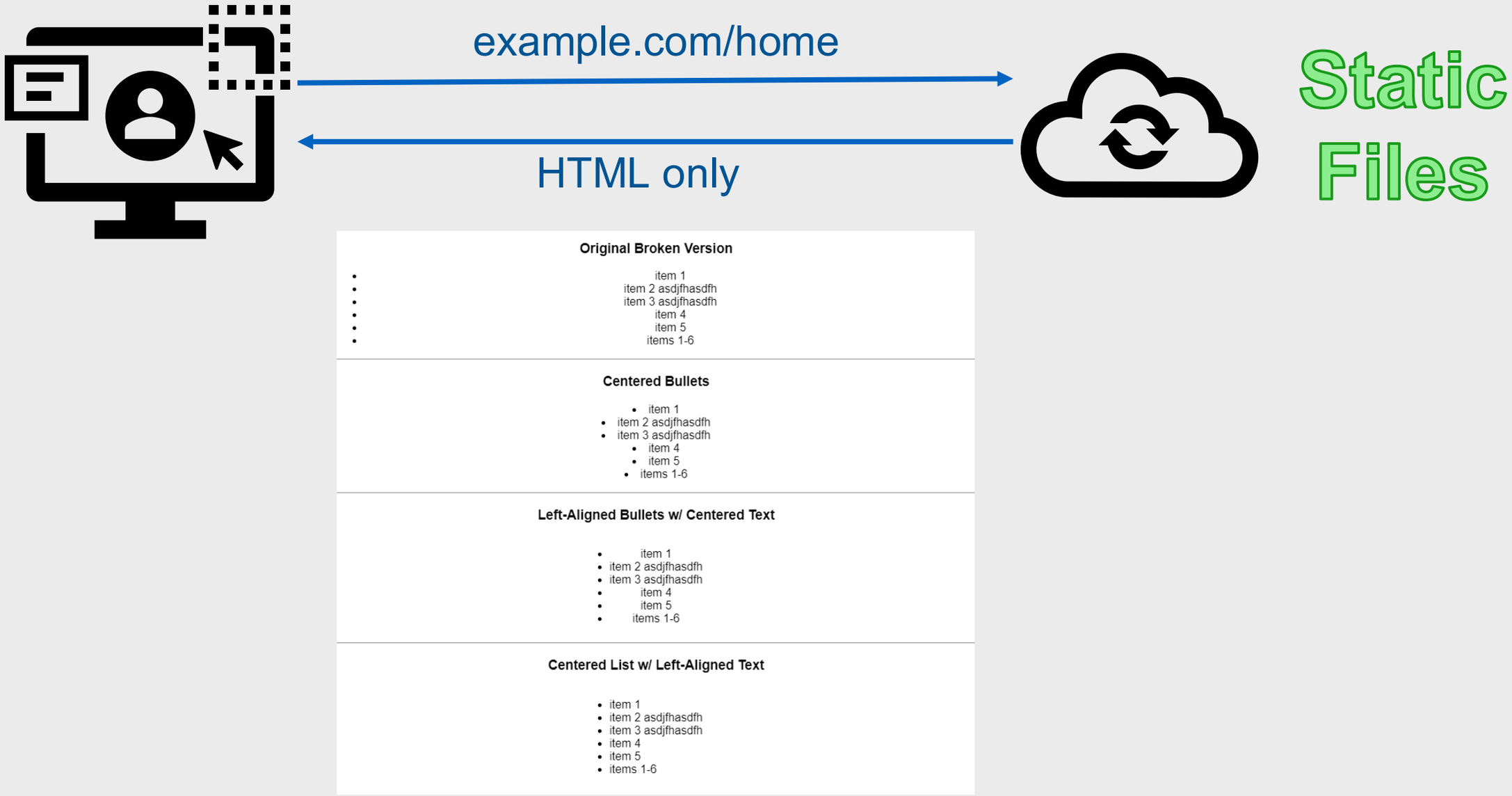
- Web Server (Apache, Nginx, IIS)
  - **Listens for Requests**
  - **Serves Files and Content**
  - **Reroute requests to the correct backend code base.**
  - **Sends Responses**
  - **Logs and Monitoring**
  - **Security: Limit unauthorized.**
  - **HTTP & HTTPS**
  - **Manages Connections** (some tweaks can be very critical)

Imaginary Evolution of Websites

Nginx Example

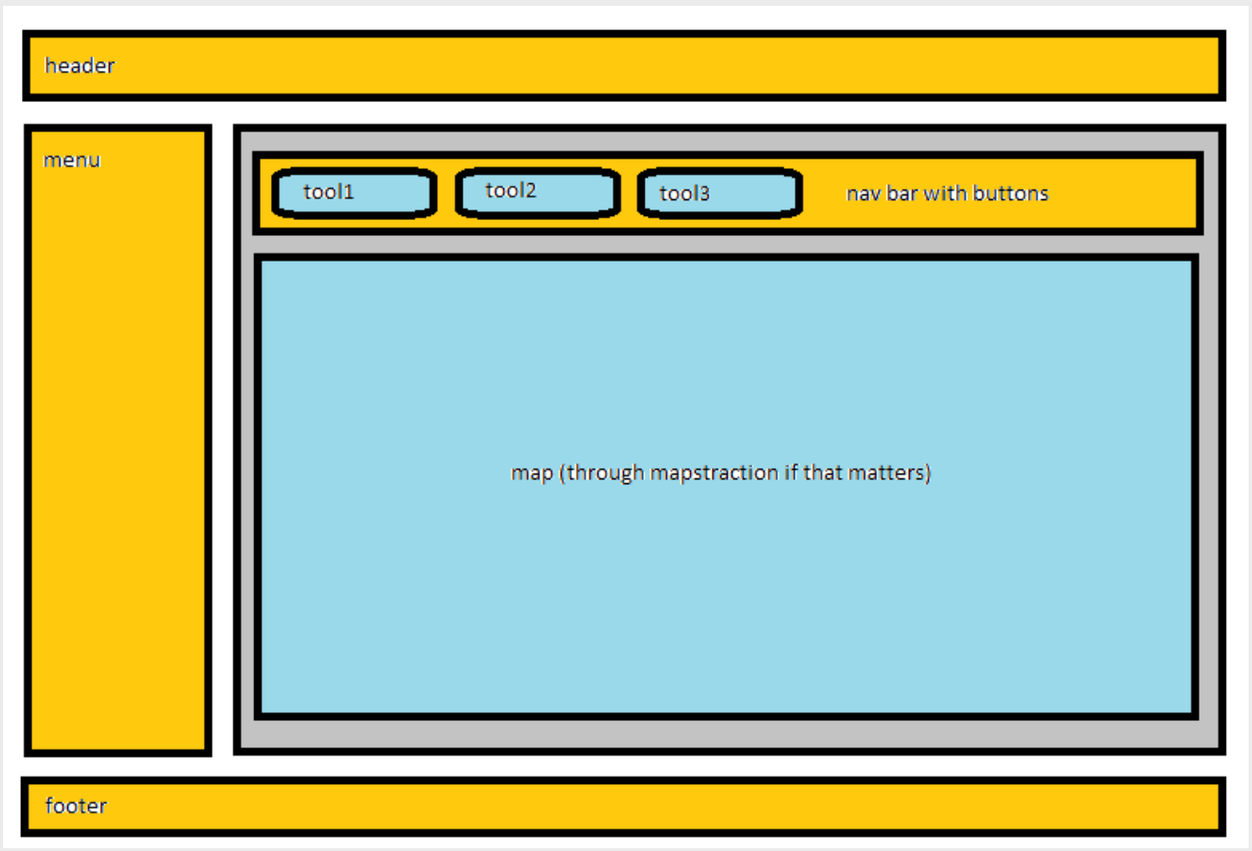
```
server {

        listen 80;
        server_name example.com; # Replace with your domain or IP address location
        / {
                root /path/to/your/html/file/directory; # Replace with the actual path
                index index.html;  # index file is index.html
        }

}
```

Imaginary Evolution of Websites



Imaginary Evolution of Websites

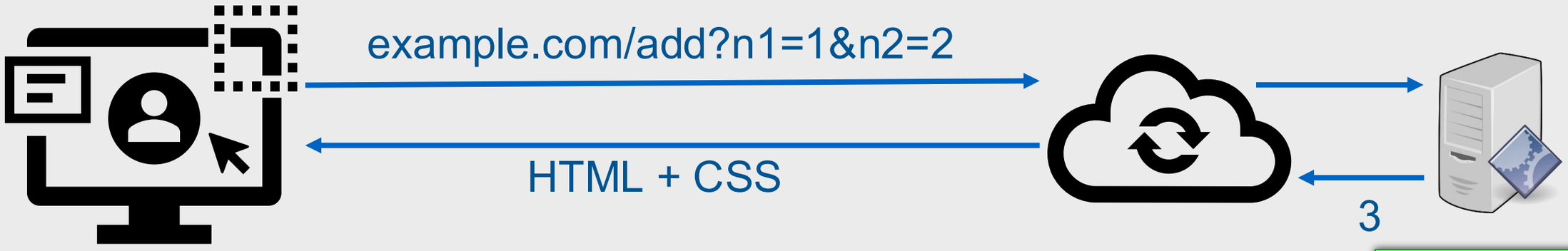
Static Files

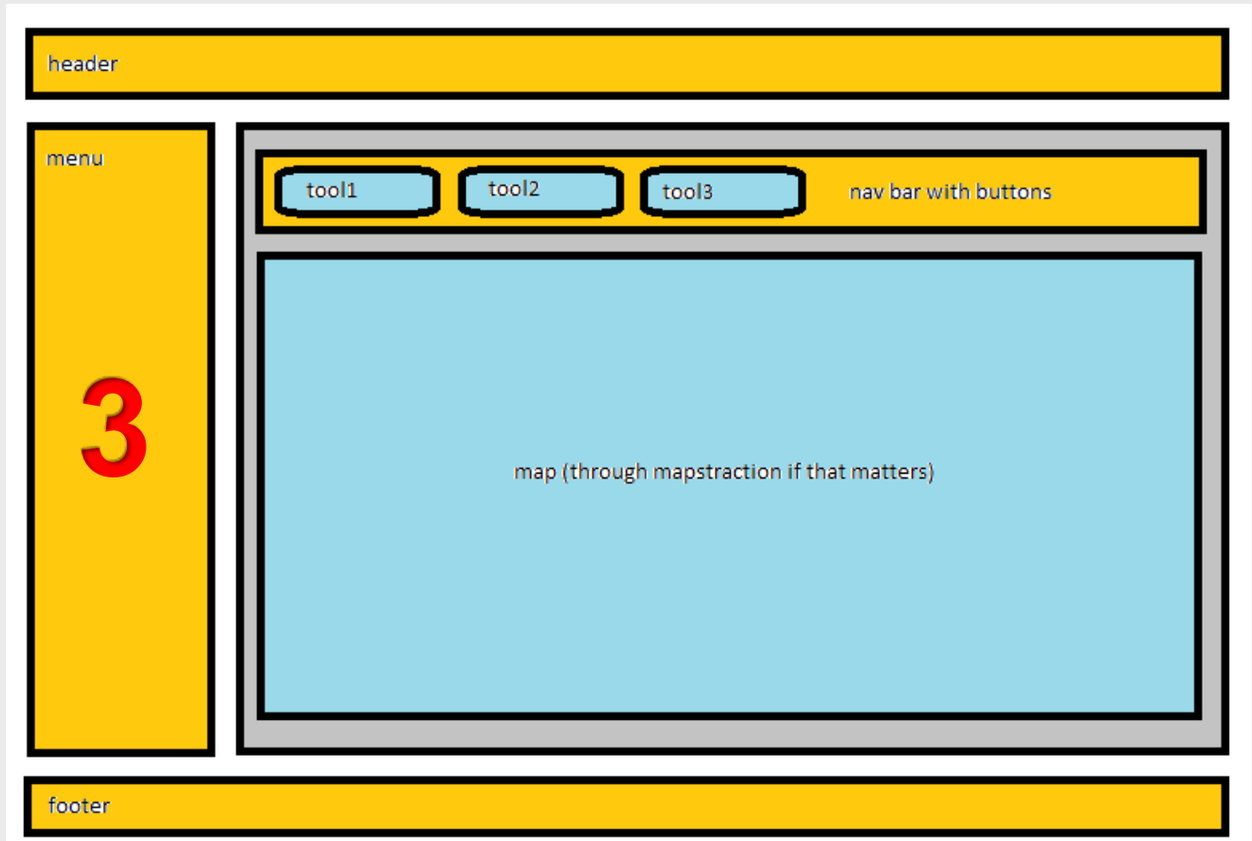


FE only

What if the server does an operation based on input (assume the it adds two numbers)

Imaginary Evolution of Websites





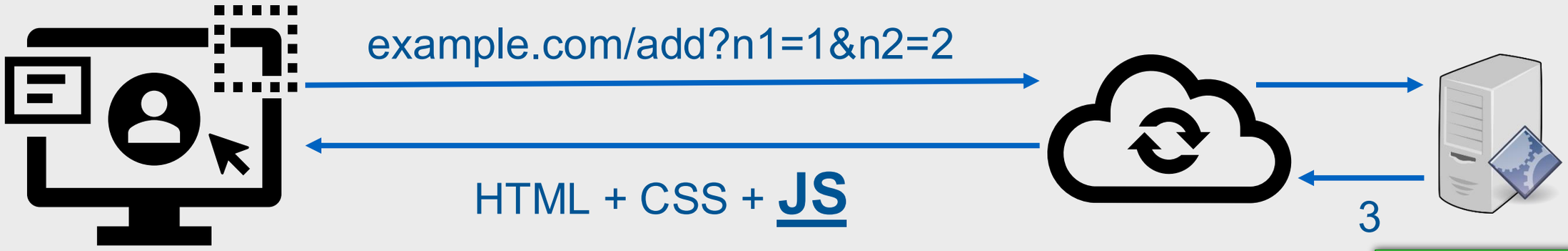
PHP
NodeJS
...

HTML
CSS

FE + BE

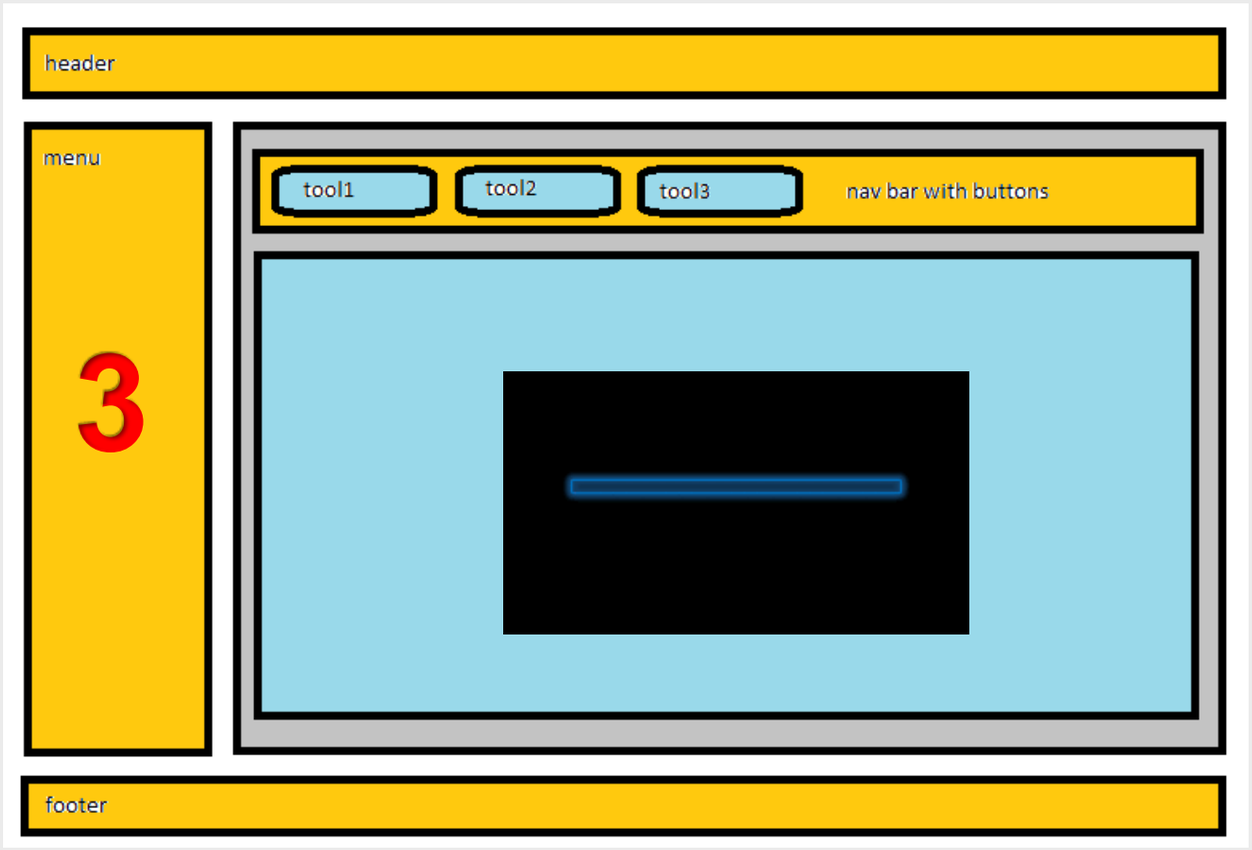
What if we want the page to be dynamic (animation or reload part without refresh)?

Imaginary Evolution of Websites



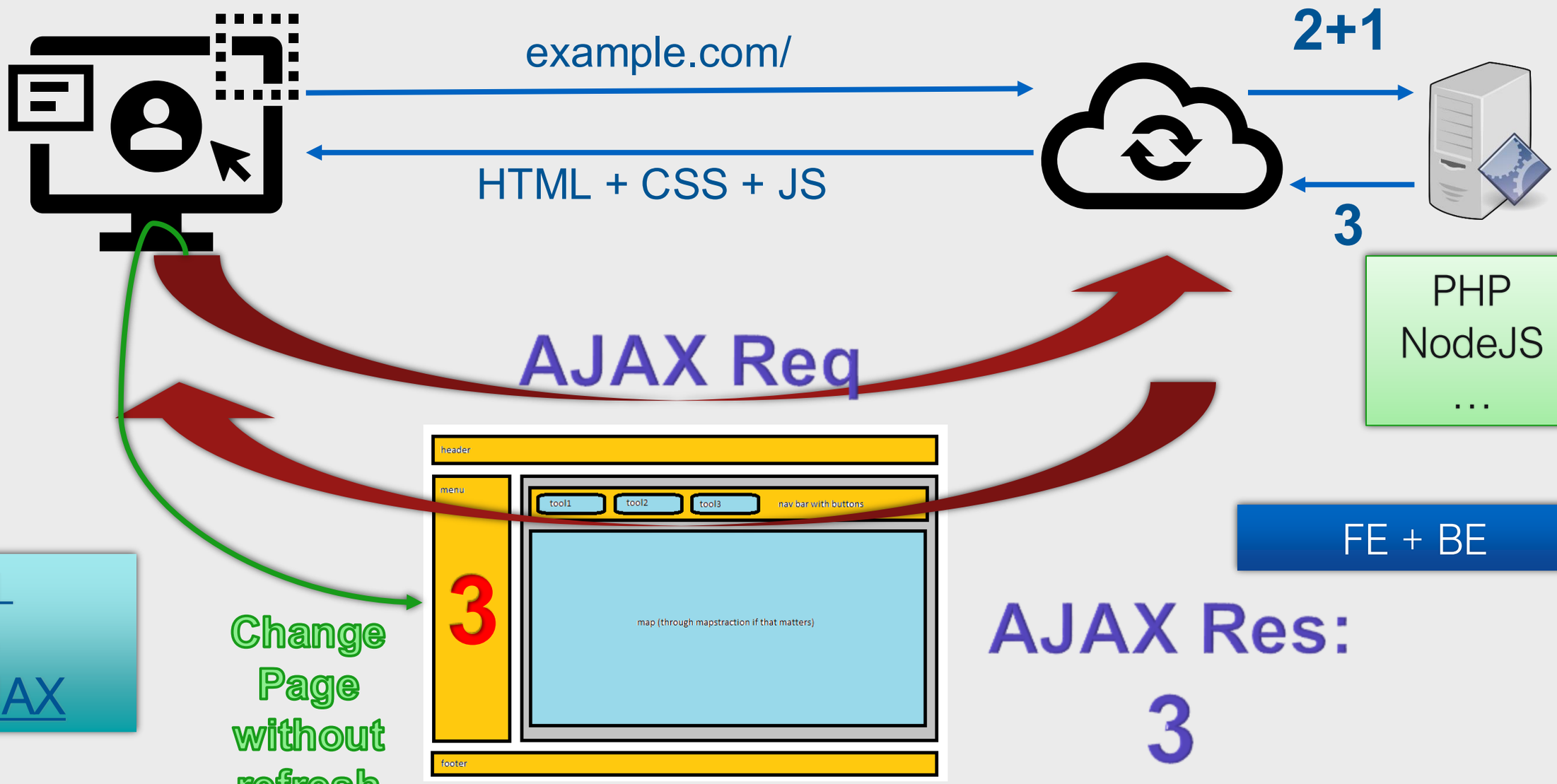
PHP
NodeJS
...

HTML
CSS
JS



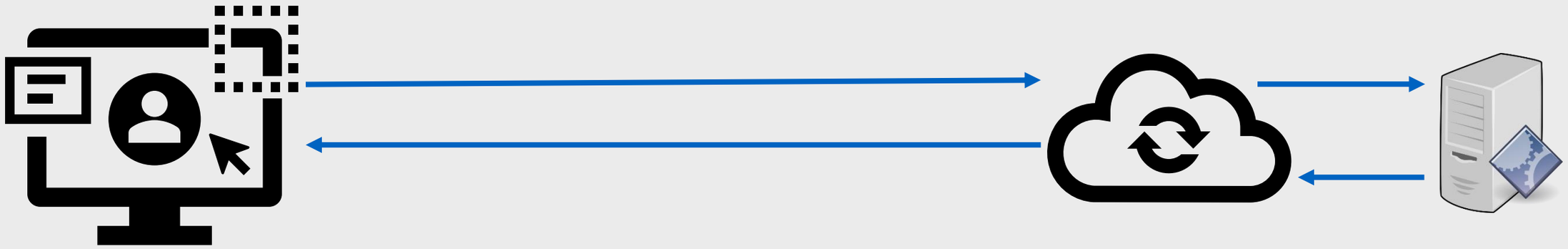
FE + BE

Imaginary Evolution of Websites



Page is loaded now

Let’s make our system more complicated!



- We need to store user data (on the server or on another)?
- We need to store user files (on the server or on another)?
- We need to deal with machine learning model?
- We need to send mails?

Journey: Running your server

The code is done, how do we deploy (Basic Method)

- **Choosing a Hosting Provider**
- **Server Types**
  - Shared hosting: it is like sharing your apartment (CPU, Memory, etc)
  - Virtual private servers (VPS): Still sharing the physical machine, but your resources are assigned to you.
  - Dedicated servers: Your server
  - Cloud hosting: **Pay for what you need** (multiple services) + **Dynamic**

Sort by Price!

- **Domain Setup**
  - Rent domain name

    DNS Propagation

How can users reach it?

We are ready and have a machine, how can we put our work?

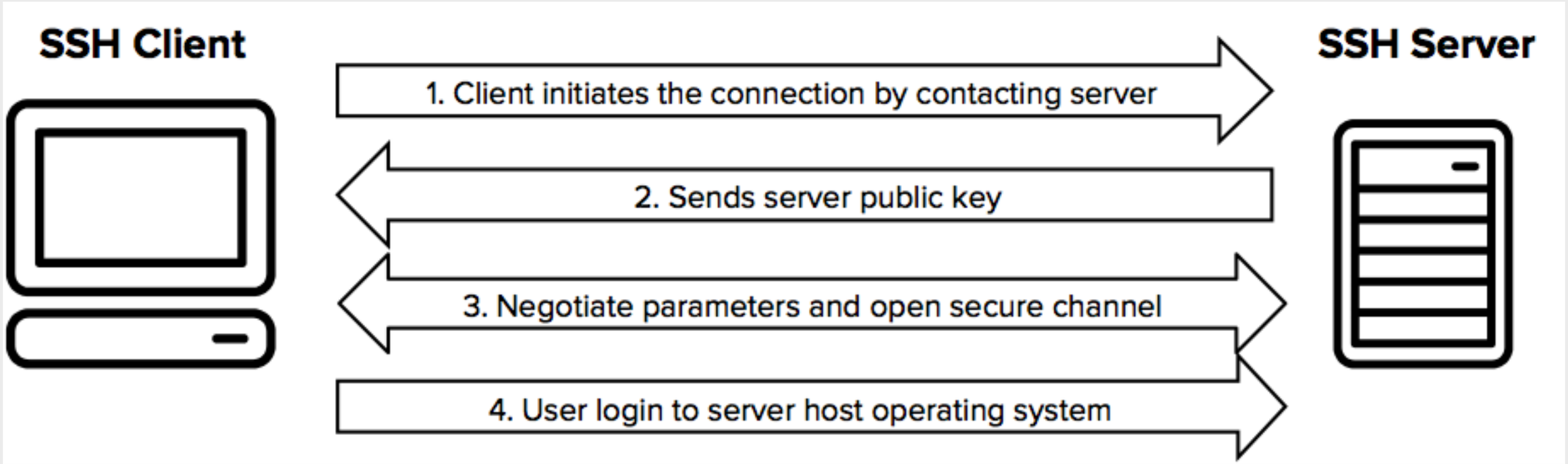
The code is done, how do we deploy (Basic Method)

- **Access the server**
  - SSH
- **Install Pre-requisites**
  - For example: PHP, MySQL, Apache…
- **Upload Project Files**
  - FTP, SFTP
- **Configuration**
  - Set environmental variables
    - Ex: DB Password.
  - Set Apache configuration
- **Generate Certificate & Configure them**
  - Certbot (LetsEncrypt)
- **CI / CD**

Getting into Server

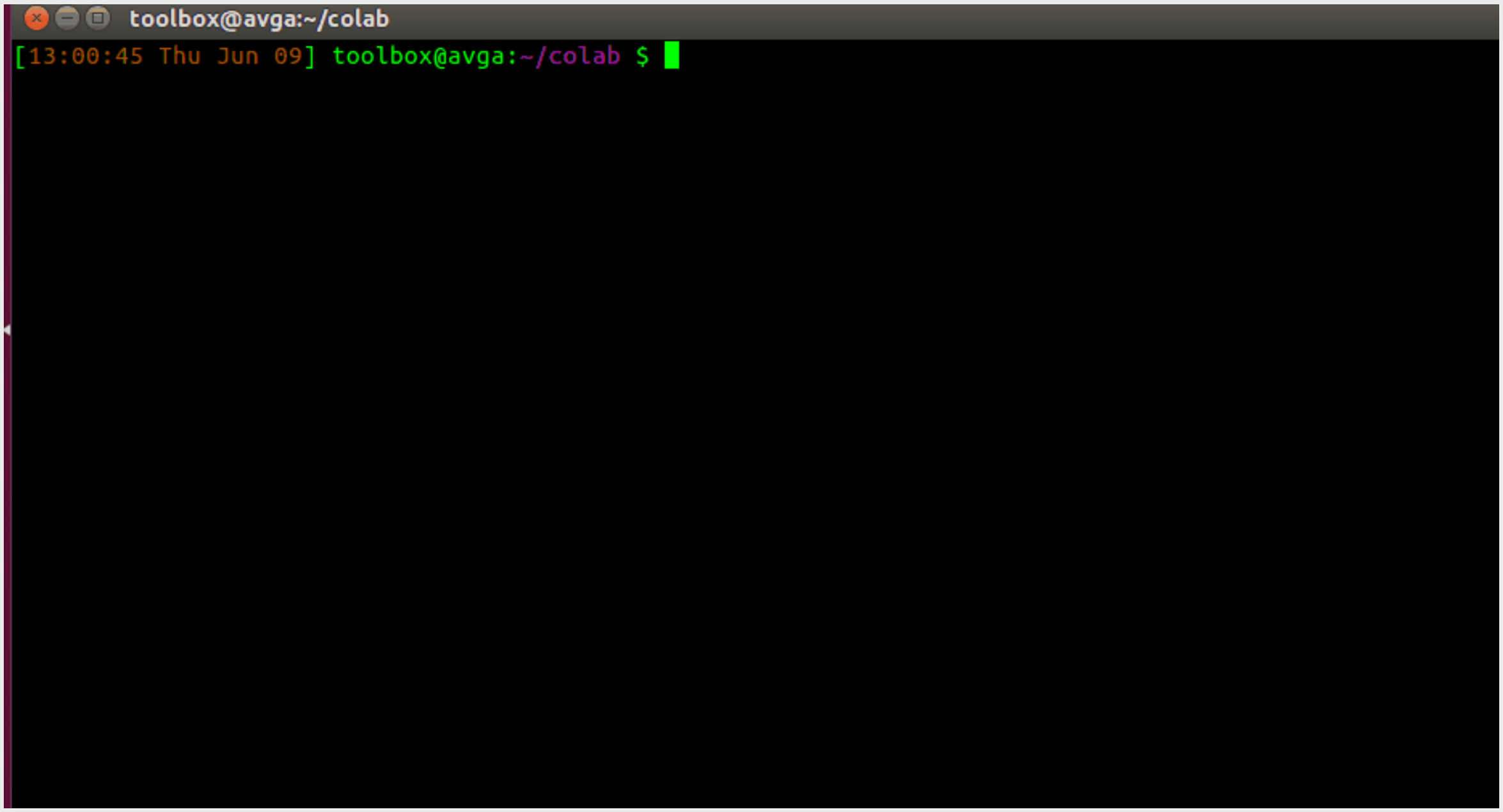
- Install Dependencies: SSH
- Move files: FTP, SFTP

SSH



https://www.ssh.com/academy/ssh

SSH

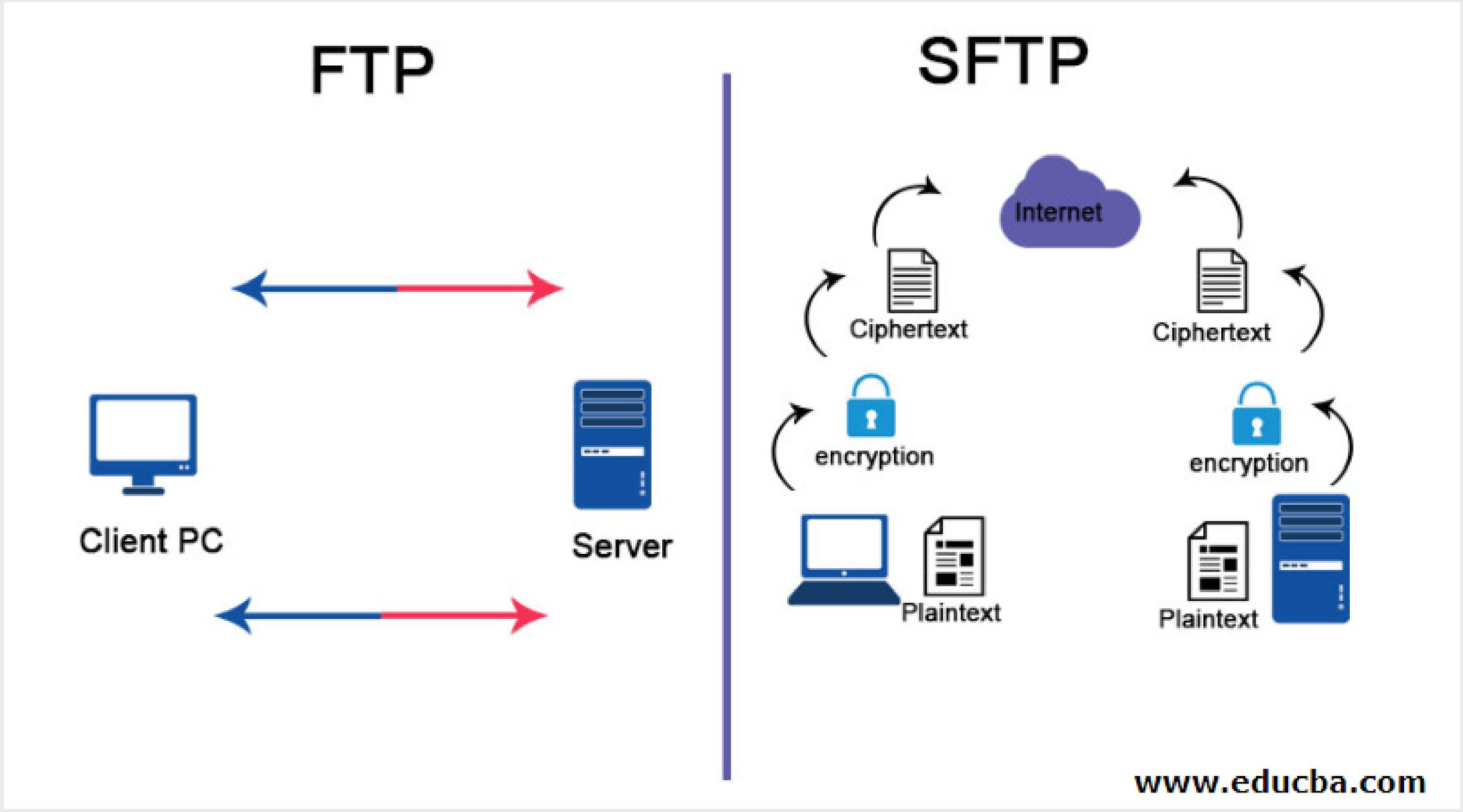


https://colab.pages.oit.duke.edu/-/external-docs/-/jobs/31499/artifacts/_book/guides/ssh.html

Transferring File: FTP vs SFTP (SSHFTP)

Feature	FTP (File Transfer Protocol)	SFTP (Secure File Transfer Protocol)
Security	Data is transferred in plain text, making it vulnerable to eavesdropping and attacks.	Data is encrypted, providing a secure transfer over the network.
Authentication	Basic authentication based on username and password, which can be intercepted.	Secure authentication through SSH (Secure Shell) keys or credentials.
Port	Uses port 21 for communication.	Uses port 22 for communication.
Encryption	No built-in encryption.	Data is encrypted using SSH protocols.
Key Authentication	Does not support key-based authentication.	Supports key-based authentication for enhanced security.

Transferring File: FTP vs SFTP (SSHFTP)



Concepts

- IP – Port - Domain name
- DNS – GeoDNS
- HTTP/HTTPS

Web Sites evolution

- Web Server?
- FE: HTML – CSS – JS
- BE: Backend

Get & Deploy a server

- Choices?
- How to deploy

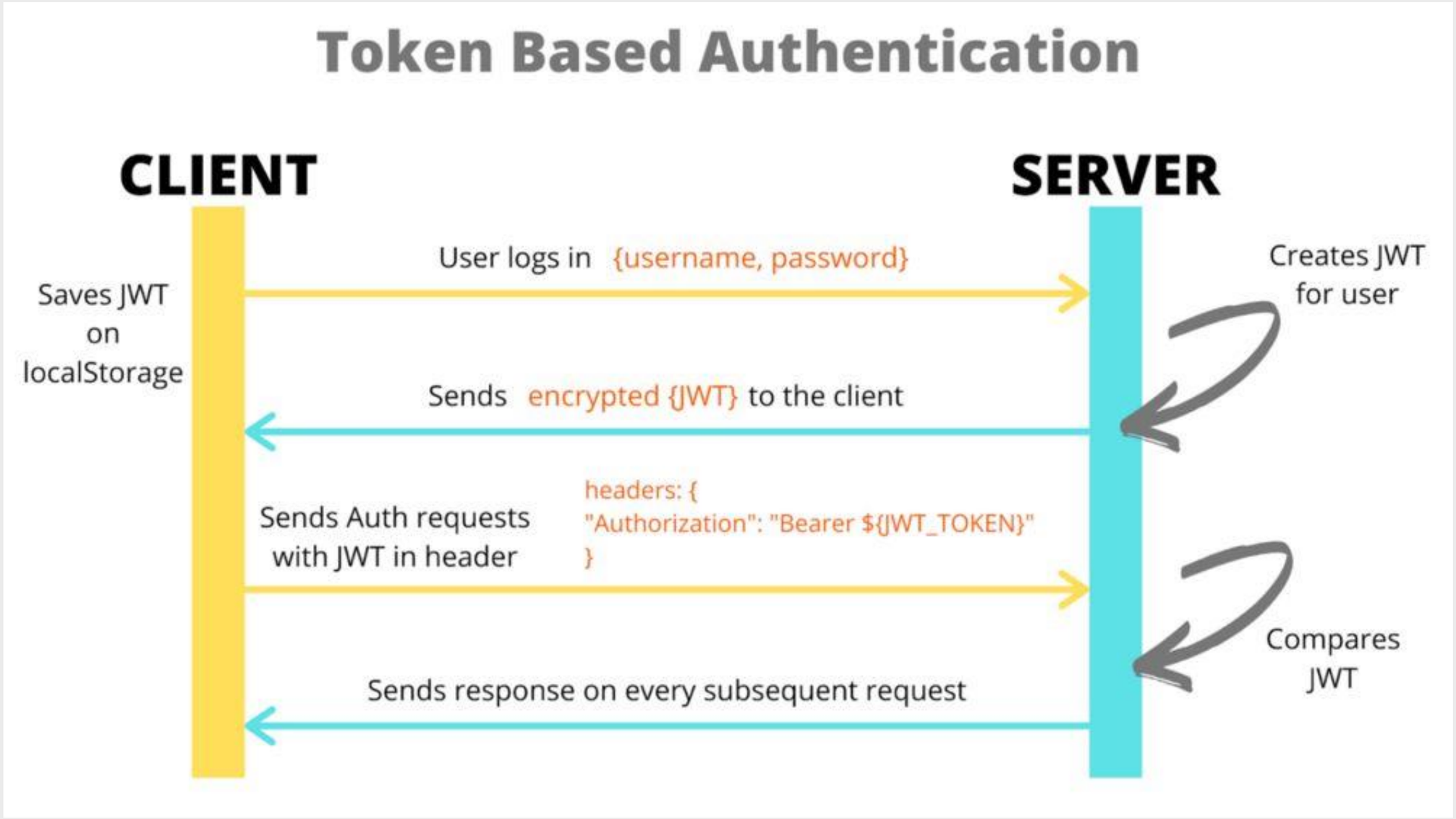
Web Browser Storage

- Cookies
- Session Storage
- Local Storage

Web Development Concepts: Authentication

- **JWT (JSON Web Tokens):**

  are commonly used for authentication and authorization in web applications.



https://www.freecodecamp.org/news/how-to-sign-and-validate-json-web-tokens/

Web Development Concepts

- **JWT (JSON Web Tokens):**

1 2

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MjIyfQ.XbPfbIHMI6arZ3Y922BhjWgQzWXcXNrz0ogtVhfEd2o 3

1 Header

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

2 Payload

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022
}
```

3 Signature

```
HMACSHA256(
BASE64URL(header)
.
BASE64URL(payload),
secret)
```


https://www.freecodecamp.org/news/how-to-sign-and-validate-json-web-tokens/

Web Development Concepts

- **Cookies:**

  commonly used for storing small amounts of data that need to persist **across multiple browser sessions or even when the browser is closed and reopened**. They are typically used for tasks like user authentication, and tracking user behavior.

- **Session Storage**:

  Session storage is useful for storing data that needs to be available **during the same browser session**. Information stored in session storage is accessible as long as the same tab or window is open. It's commonly used for temporary data needed for a specific task or page. For example, page history, or data between pages.

- **Local Storage**:

  Suitable for storing **larger amounts of data** that need to **persist** beyond the current session. It's commonly used for caching data, storing user preferences, and saving application state.

Web (Browser) Storage Options

CRITERIA	COOKIES	LOCAL STORAGE	SESSION STORAGE
MAXIMUM DATA SIZE	4 KB	5 MB	5 MB
BLOCKABLE BY USERS	YES	YES	YES
AUTO-EXPIRY OPTION	YES	NO	YES
SUPPORTED DATA TYPES	STRING ONLY	STRING ONLY	STRING ONLY
BROWSER SUPPORT	VERY HIGH	VERY HIGH	VERY HIGH
ACCESSIBLE SERVER SIDE	YES	NO	NO
DATA TRANSFERRED ON EVERY HTTP REQUEST	YES	NO	NO
EDITABLE BY USERS	YES	YES	YES
SUPPORTED ON SSL	YES	N/A	N/A
CAN BE ACCESSED ON	SERVER & CLIENT SIDE	CLIENT SIDE ONLY	CLIENT SIDE ONLY
CLEARING/DELETING	PHP, JS, AUTOMATIC	JS ONLY	JS & AUTOMATIC
LIFETIME	AS SPECIFIED	TILL DELETED	TILL TAB IS CLOSED
SECURE DATA STORAGE	NO	NO	NO

https://stackoverflow.com/a/70846045

Web Development Concepts

- **JWT (JSON Web Tokens):**
- **Storing Options:**
  1. **HTTP Cookies**
  2. **Session Storage** : **why not widely used?**
  3. **Local Storage**

Security of local storage: XSS?

Cookie: HTTPOnly?

https://www.freecodecamp.org/news/how-to-sign-and-validate-json-web-tokens/

Question: Choose storage mechanism

Scenario: Storing a token for authenticating a user after login.

a)Cookies
b)Session Storage
c)Local Storage

Question: Choose storage mechanism

Scenario: Saving a user's selected items in an online shopping cart for a seamless shopping experience across pages.

a)Cookies
b)Session Storage
c)Local Storage

Question: Choose storage mechanism

Scenario: Holding form data while the user fills out a multi-step form and ensuring the data is not lost during navigation or accidental refresh.

a)Cookies
b)Session Storage
c)Local Storage

Answers

Question: Choose storage mechanism

Scenario: Storing a token for authenticating a user after login.

a)Cookies
b)Session Storage
c)Local Storage

Question: Choose storage mechanism

Scenario: Saving a user's selected items in an online shopping cart for a seamless shopping experience across pages.

a)Cookies
b)Session Storage
c)Local Storage

Question: Choose storage mechanism

Scenario: Holding form data while the user fills out a multi-step form and ensuring the data is not lost during navigation or accidental refresh.

a)Cookies
b)Session Storage
c)Local Storage

How do we sign in with Facebook, Google, ..

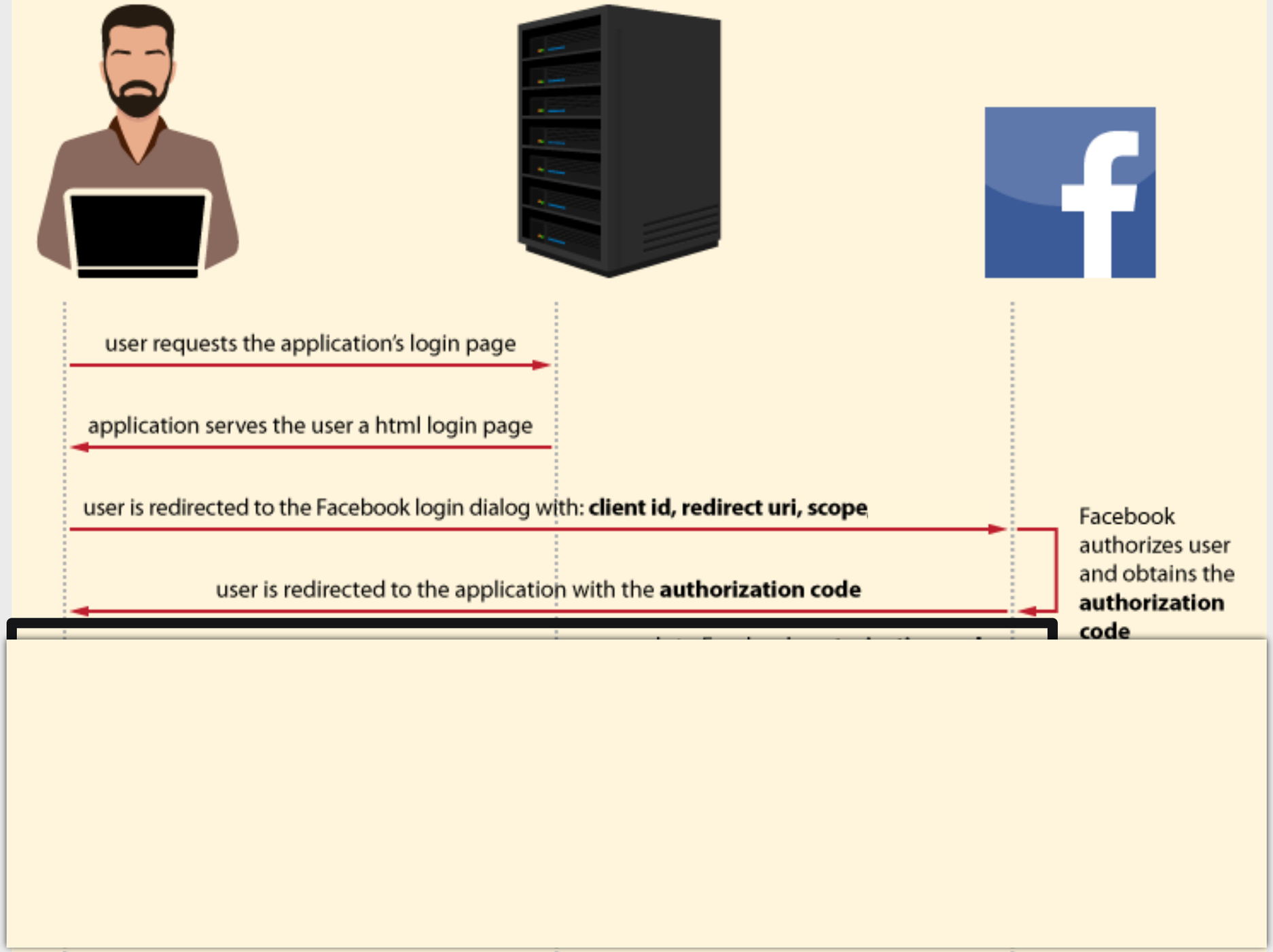
oAuth

oAuth

The developer creates a facebook App and receives:

1. Client id: can be used directly by user.
2. Client Secret: is only used by the server.

oAuth


user requests the application's login page
application serves the user a html login page
user is redirected to the Facebook login dialog with: client id, redirect uri, scope
Facebook authorizes user and obtains the authorization code
user is redirected to the application with the authorization code