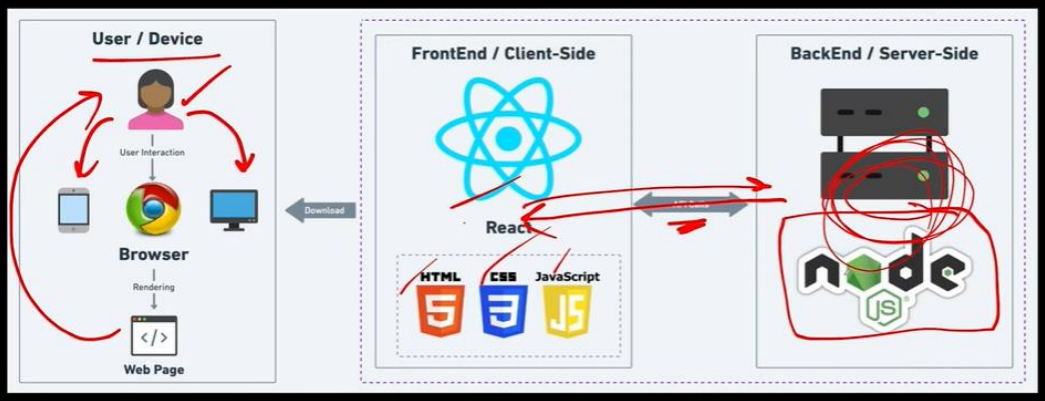
Lesson_1

8. Server architecture with NodeJs

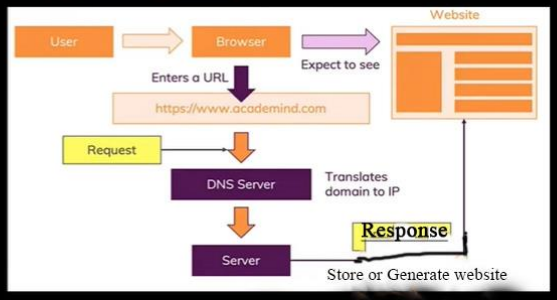


Nodejs server will:

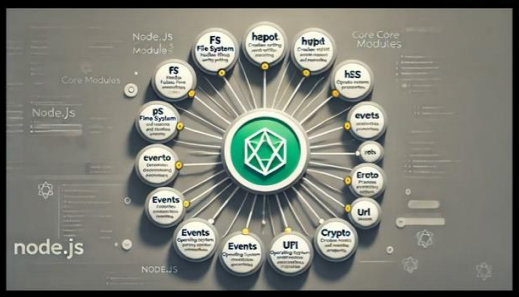
1. Create server and listen to incoming requests
2. Business logic: validation, connect to db, actual processing of data
3. Return response HTML, JSON, CSS, JS

3.2 How Web Works?

1. Client Request Initiation: The client (browser) initiates a network call by entering a URL.
2. DNS Resolution: The browser contacts a DNS server to get the IP address of the domain.
3. TCP Connection: The browser establishes a TCP connection with the server's IP address.
4. HTTP Request: The browser sends an HTTP request to the server.
5. Server Processing: The server processes the request and prepares a response.
6. HTTP Response: The server sends an HTTP response back to the client.
7. Network Transmission: The response travels back to the client over the network.
8. Client Receives Response: The browser receives and interprets the response.
9. Rendering: The browser renders the content of the response and displays it to the user.



3.4 Node Core Modules



1. Built-in: Core modules are included with Node.js installation.
2. No Installation Needed: Directly available for use without npm install.
3. Performance: Highly optimized for performance.

3.4 Node Core Modules

1. fs (File System): Handles file operations like reading and writing files.
2. http: Creates HTTP servers and makes HTTP requests.
3. https: Launch a SSL Server.
4. path: Provides utilities for handling and transforming file
5. paths.os: Provides operating system-related utility methods and properties.
6. events: Handles events and event-driven programming.
7. crypto: Provides cryptographic functionalities like hashing and encryption.
8. url: Parses and formats URL strings.

3.5 Require Keyword

1. Purpose: Imports modules in Node.js.
2. Caching: Modules are cached after the first require call.
3. .js is added automatically and is not needed to at the end of module name.
4. Path Resolution: Node.js searches for modules in core, node_modules, and file paths.

Syntax:

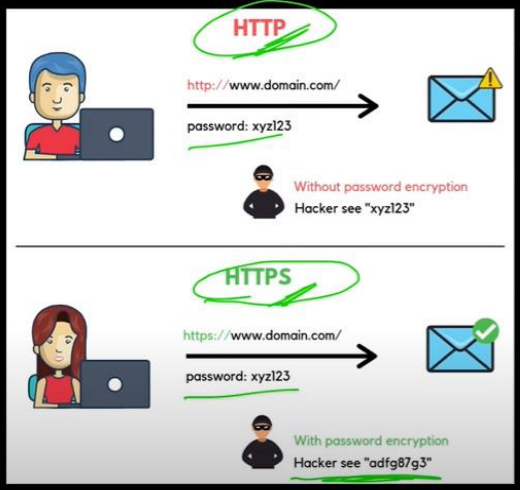
```
const moduleName = require('module');

// Load the built-in http module
const http = require('http');

// Load the third party express module
const express = require('express')

// Load the custom myModule module
const myModule = require('./myModule')
```

3.3 What are Protocols?



Http (HyperText Transfer Protocol):

- Facilitates communication between a web browser and a server to transfer web pages.
- Sends data in plain text (no encryption).
- Used for basic website browsing without security.

HTTPS (HyperText Transfer Protocol Secure):

- Secure version of HTTP, encrypts data for secure communication.
- Uses SSL/TLS to encrypt data.
- Used in online banking, e-commerce.

TCP (Transmission Control Protocol):

- Ensures reliable, ordered, and error-checked data delivery over the internet.
- Establishes a connection before data is Transfered

3.6 Creating first Node Server

```
1  // Simple NodeJS server
2  const http = require('http');
3
4  const server = http.createServer((req, res) => {
5    console.log(req);
6  });
7
8  const PORT = 3000;
9  server.listen(PORT, () => {
10   console.log(`Server running at http://localhost:${ PORT}
11 });
```