Introduction to Node.js

Why Do We Need a Backend?

In web development, **frontend** and **backend** are two key parts of an application.

- The **frontend** is what users see and interact with—like buttons, forms, and text on a webpage. It runs in the browser.
- The **backend** is like the behind-the-scenes part. It handles things like:
    - Storing and retrieving data from databases
    - Authenticating users (login/signup)
    - Processing business logic
    - Securing sensitive operations
    - Handling API requests from the frontend

Without a backend, a website can’t store user data, communicate with a database, or perform secure tasks. The frontend would just be a static page with limited interactivity.

Example

Imagine a to-do list app:

- The **frontend** displays the tasks and lets the user add or remove them.
- The **backend** saves these tasks to a database, so they’re still there when the user comes back later.

Introduction to Node.js

Node.js is a runtime environment that lets you run JavaScript on the **server**, not just in the browser.

- Normally, JavaScript runs only in the browser (client-side).
- Node.js allows you to run JavaScript on your computer or server (server-side).

With Node.js, you can build the backend of your application using JavaScript—the same language you use for the frontend. This makes development faster and easier, especially for beginners.

Key Features of Node.js

- Built on Chrome's V8 JavaScript engine
- Handles many requests at once using non-blocking (asynchronous) code
- Has a large ecosystem of libraries (called npm)
- Great for building APIs, real-time apps (like chat apps), and full-stack JavaScript projects

---

Server-side vs. Client-side JavaScript

Feature	Client-side (Browser)	Server-side (Node.js)
Runs on	User's browser	Web server
Use case	Displaying content, UI interaction	Storing data, handling logic
Access to system resources	Limited (sandboxed)	Full access (file system, network)
Performance	Depends on user's device	Depends on server
Examples	Form validation, animations	Database queries, user authentication

Why the Difference Matters

- Running JavaScript in the **client** is good for creating interactive webpages, but it’s limited and not secure for sensitive operations.
- Running JavaScript in the **server** (with Node.js) allows you to perform secure tasks and manage application logic centrally.

Summary

- A **backend** is essential for dynamic websites that need to store data, handle users, or connect to databases.
- **Node.js** lets you write backend code using JavaScript, making full-stack development more accessible.
- **Client-side** JavaScript is for user interaction; **server-side** JavaScript (via Node.js) handles the logic and data behind the scenes.