Node JS

Node.js is a runtime environment that allows you to run JavaScript outside the browser, typically on the server side.

📌 **In Simple Terms**:
Normally, JavaScript runs in browsers (like Chrome or Firefox) for things like animations, forms, and interactivity.

But Node.js lets you run JavaScript on your computer/server, so you can build backend applications like:

- Servers (e.g., APIs)
- Command-line tools
- Real-time apps (like chat apps)
- File systems or databases interaction

Node.js is a JavaScript Runtime Environment.

- Runs JavaScript code outside the browser.
- Built on Chrome's V8 Engine. (Made with C++)

Who Created Node.js and Why?

- **Created by**: Ryan Dahl
- **Released in**: 2009 (initial work started in 2007)
- **Reason**:
  - Traditional servers like **Apache** handled concurrent requests inefficiently.
  - Node.js was designed for **non-blocking**, event-driven, real-time applications.

Installing Node.js

- Download from https://nodejs.org
- Choose:
  - **LTS (Long Term Support)**: Stable version recommended for most users.
  - **Current**: Latest features but less stable.

Running JavaScript Files with Node

- Use the terminal/command prompt:

```
node <filename> .js
```

- Node provides its own runtime environment with built-in **APIs** like:
  - `fs` (file system)
  - `http` (server creation)

- `path`, etc.

Packages in Node.js

- **Packages** are reusable libraries or tools.
- Installed using **npm (Node Package Manager)**.
- Example:

```
npm install cat-me
```

Packages vs Modules

Feature	Package	Module
Definition	Third-party tools/libraries	Built-in features provided by Node.js
Source	Installed via `npm`	Comes with Node.js
Examples	`express`, `cat-me`	`http`, `fs`, `path`

Server Create Through HTTP Module

- Make a file named `server.js`

```
const http = require('http')
```

- While installing `cat-me` we used `npm install cat-me` but we're not using any npm packages while running `http`

Reason: `http` is a module, not a package.

Server Creation:

```
http.createServer()
```

Server Start:

```
const server = http.createServer()

server.listen(3000,()=>{
  console.log("Server is running on port 3000")
})
```

- The callback will get executed when the server is ready to take requests & handle it.

Request & Response

```
const http = require('http')
const server = http.createServer((req, res)=>{
  res.end("hello World From The Server")
})

server.listen(3000,()=>{
  console.log("Server is running on port 3000")
})
```

- programming the server - if any request comes this will be the consistent response.