

AI Response:

Node.js is a powerful JavaScript runtime built on Chrome's V8 JavaScript engine

1. **Event-Driven Architecture**

Node.js is event-driven, meaning that it responds to events emitted by various objects.

Key Components:

- **Event Loop**: The core of Node.js. It manages the execution of code and handles events.
- **EventEmitter**: A class that is part of Node.js which allows objects to send and receive events.

2. **Non-Blocking I/O**

Node.js uses non-blocking, asynchronous I/O operations. This means that instead of waiting for an operation to complete, the program can continue to execute other tasks.

Key Features:

- **Callbacks**: Functions passed as arguments that execute when an operation completes.
- **Promises**: Objects representing the eventual completion (or failure) of an asynchronous operation.
- **async/await**: Syntax introduced in ES2017 that allows writing asynchronous code in a synchronous style.

3. **Single-Threaded Model**

Although Node.js uses a single thread to handle requests, it can handle many concurrent connections.

4. **Modules**

Node.js uses a module system based on CommonJS, allowing developers to organize their code into reusable modules.

Key Concepts:

- **Module**: A self-contained block of code that can be exported and imported in other modules.
- **npm (Node Package Manager)**: A package manager for JavaScript that allows developers to install and manage dependencies.

5. **Built-in Libraries**

Node.js comes with a set of built-in libraries, referred to as "core modules," that provide essential functionality.

- **HTTP**: To create web servers.
- **File System (fs)**: To interact with the file system.
- **Path**: To handle and transform file paths.
- **Events**: To handle events in a consistent manner.

6. **Frameworks**

Although Node.js provides a core set of APIs, many frameworks build on top of it to simplify development.

- **Express.js**: A minimalist web framework for building web applications and APIs.
- **NestJS**: A framework for building efficient, scalable Node.js server-side applications.

7. **Middleware**

In the context of frameworks like Express.js, middleware functions are functions that have access to request and response objects.