## ​ What is Node JS and course Introduction?

## ​ Basic theory of node JS

## ​ Installation and setup

## ​ First script program with node JS

## ​ Fundaments of node JS

## ​ Core module in Node JS

## ​ Make basic server in node JS

* open-source server environment. uses JavaScript on the server.
* runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
* Node.js can generate dynamic page content
* Node.js can create, open, read, write, delete, and close files on the server
* Node.js can collect form data
* Node.js can add, delete, modify data in your database.
* Node.js is written in C, C++, and JavaScript.
* Node.js uses the “Single Threaded
* Node.js is not a programming language. Rather, it’s a runtime environment that’s used to run JavaScript outside the browser.
* Neither is Node.js a framework
* A common misconception among developers is that Node.js is a backend framework and is only used for building servers. This isn’t true: Node.js can be used both on the frontend and the backend.
* Difference between JavaScript and NodeJS- 1. Javascript is a programming language that is used for writing scripts on the website. NodeJS is a Javascript runtime environment.
* Features
* Packages
* Speed
* Multi-platform

1. ​ What is the package.json file and use in Node JS

records important metadata about a project (dependencies, repository, main, version).usually in the project root

1. ​ Small challenge for you
2. ​ What is Nodemon in node JS

* nodemon is a tool that helps develop node. js based applications by automatically restarting the node application when file changes in the directory are detected.
* It helps to reduce the time of typing the default syntax node <file name> for execution again and again.
* difference between node and Nodemon?
* When you are using node you have to restart on your own to see the changes you made But nodemon watches the particular path for any changes. If you make any changes in your file, nodemon will restart it for you.

## ​ Make Simple API in node JS

## ​ Input from command line

## ​ Show file List with file system

## ​ Curd with file system

## ​ Asynchronous Node JS

NodeJS is an asynchronous event-driven JavaScript runtime environment designed to build scalable network applications. Asynchronous here refers to all those functions in JavaScript that are processed in the background without blocking any other request.

## ​ Handle Asynchronous Data in Node JS

## ​ How Node JS Works

NodeJS is an asynchronous event-driven JavaScript runtime environment designed to build scalable network applications. Asynchronous here refers to all those functions in JavaScript that are processed in the background without blocking any other request.Node.js basically works on two concepts

* Asynchronous
* Non-blocking, I/O

## ​ Express JS Introduction

provides broad features for building web and mobile applications. It is used to build a single page, multipage, and hybrid web application. Express is the most popular Node web framework,

**Difference between Node.js and Express.js:**

Node.js is a platform for building the i/o applications which are server-side event-driven and made using JavaScript.

Express.js is a framework based on Node.js for which is used for building web-application using approaches and principles of Node.js. event-driven.

## ​ Routing with express JS

## ​ Render html in node JS

## ​ Make HTML pages in node JS

## ​ Remove file extension

## ​ Template Engine eJS

## ​ Dynamic page in node JS

## ​ Middleware node JS

Middleware functions can perform the following tasks:

* Execute any code.
* Make changes to the request and the response objects.
* End the request-response cycle.
* Call the next middleware function in the stack.

**Types of express middleware**

* Application-level middleware app.use (Auth/Logging/Router Level /Error Handing middleware
* Router level middleware router.use
* Built-in middleware express.static,express.json,express.urlencoded
* Error handling middleware app.use(err,req,res,next)
* Third-party middleware bodyparser,cookieparser

Advantages of using middleware:

* Middleware can process request objects multiple times before the server works for that request.
* Middleware can be used to add logging and authentication functionality.
* Middleware improves client-side rendering performance.
* Middleware is used for setting some specific HTTP headers.
* Middleware helps for Optimization and better performance.

## ​ Route level middleware

## ​ Install Mongodb for node

## ​ Basics of MongoDB

## ​ CURD operation with MongoDB

## ​ Connect Mongodb with node

## ​ Read data from MongoDB to node JS

## ​ Insert data from MongoDB to node JS

## ​ Update data from MongoDB to node JS

## ​ Read data from MongoDB to node JS

## ​ GET API with MongoDB and node JS

## ​ POST API method with node JS and MongoDB

## PUT API in node JS and mongo DB

## DELETE API in node JS and mongo DB

## Node JS mongoose

## CRUD operations with node JS and mongoose

## POST API with node JS and mongoose

## GET, DELETE AND PUT APIs with node JS and mongoose

## Search API with node JS and mongoose

## CRUD operations with node JS and mongoose

## OS module node JS

## CRUD operations in node JS

## Events and event emitter in node JS

The EventEmitter is a module that facilitates communication/interaction between objects in Node. EventEmitter is at the core of Node asynchronous event-driven architecture

## REPL in node JS

Read-Eval-Print-Loop (REPL) is an easy-to-use command-line tool, used for processing Node. js expressions.