**Module 1:Intro to Node JS**

* Introduction:
* What is Node Js

Node Js is a Javascript runtime built on Chrome’s V8 Javascript Engine. Node Js uses an event-driven, Non-Blocking model that makes it lightweight and efficient. Node Js package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

Node Js represents a “javaScript everywhere” paradigm, unifying web application development around a single programming language rather than different languages for server and client side scripts.

Node Js lets developers use JS to write command line tools and for server side to produce dynamic web page content before the page is sent to the user’s web browser.

What it basically does is?

1. A platform which allows us to run JavaScript on a computer/server
2. Read, Delete and Update Files
3. Easily communicate with a DB

* Advantages of Node Js

1. It is very lightweight and fast
2. It is highly customizable and easy to configure
3. It comes with Node Package Manager(NPM) that contains 140,000+ packages that are available for free
4. It allows one to build the application in JS “top-to-bottom”, even down to DB level if NoSQL DB that stores objects in JSON(like MongDB and Firebase) is used. This makes development easier
5. It allows code to be reused across the client-side and server-side of the application.

* Traditional Web Server Model

Havlena.net-> Traditional Web Server(Blocking and Non Blocking I/O)

* Node Js Process Model

As discussed in the previous module, in the traditional web server model, each request is handled by a dedicated thread from the thread pool. If no thread is available in the thread pool at any point of time then the request waits till the next available thread. Dedicated thread executes a particular request and does not return to the thread pool until it completes the execution and returns a response.

Node.js processes user requests differently when compared to a traditional web server model. Node JS runs in a single process and the application code runs in a single thread and therby needs less resources than other platforms. All the user requests to your web application will be handled by a single thread and all the I/O work or long running job is performed asynchronously for a particular request. So, the single thread does not have to wait for the request to complete and is free to handle the next request. When asynchronous I/O work completed then it processes the request further and sends the response.

An event Loop is constantly watching for the events to be raised for an asynchronous job and executing callback function when the job completes. Internally, Node JS uses libev for the event loop which in turn uses internal C++ thread pool to provide asynchronous I/O.

Node.js process model increases the performance and scalability with a few caveats. Node Js is not fit for an application which performs CPU-intensive operations like image processing or other heavy computation work because it takes time to process a request and thereby blocks the single thread.

**Module 3: Node JS Modules**

* Functions

Javascript is a functional programming language, functions are fully typed objects that can be manipulated, extended, and passed around as data.

The default return type of a node js function is undefined.

**Function Example**

Function Scope:

Every time a function is called, a new variable scope is created. Variables declared in the parent scope are available to the function. Variable declared within the new scope are not available where the function exits.

**Example**

**Immediately invoked Function : ()=>{}**

**HIGHER ORDER FUNCTIONS(ADVANCED)**

* **BUFFER**

Need more insights on this.

* **MODULES**

Node Modules can be considered same as JavaScript Libraries. Modules are simple or complex functionality organized in single or multiple file JS files which can be reused throughout the Node Js application

To include a module, use the require() function with the name of the module. For example:

var http=require(‘http’)

We can also create our own module which in turn can also be used in other programs using the require keyword.

Use the exports keyword to make properties and methods available outside the module file.

Let us create a module which gives the current date and time

exports.CurrentDT=function(){

return Date();

};

There are primarily three types of Node Modules:

1. Core Modules
2. Local Modules
3. Third Party Modules

**Core Modules**

The core modules include bare minimum but important functionalities of Node Js. These modules are compiled into its binary distribution and load automatically when Node process starts, however you need to import the node modules manually in order to use it in your application.

**Complete other modules.**

**Module 4: Node Package Manager**

* What is NPM?

NPM is a package manager for the JavaScript programming language. It is the default package manager for the JavaScript runtime environment Node Js. It consists of a command line client, also called npm, and an online databases of free public and paid-for private packages, called the npm registry. The registry is accessed via the client, and the available packages can be browsed and searched via the npm website. The package manager and the registry are managed by npm,Inc.

* Installing Packages Locally (Demo)
* Adding dependency to package.json (Demo)
* Installing package globally (Demo)
* Updating packages (Demo)