NodeJS Server

Node.JS:

- A server-side platform which is built on Google Chrome's V8-JavaScript Engine which was initially built and developed by Mr. Rayn Dal in the year 2009.
- Using Node.js we could able to built faster and scalable network applications.
- Node.js internally uses event driven and non-blocking I/O mechanism (or) model which makes it light weight and efficient for data intensive real time applications which run on distributive system/devices.
- Node.js is open source and cross-platform runtime environment using which we could develop server side and networking-based applications.
- JavaScript is programming language using which we could able to add instructions within Node.js.
- Node.js comes with rich library of various JavaScript modules which simplifies the development of the web applications using Node.js.
- Node.js is a combination of Run-Time environment and JavaScript library.

Features of Node.JS used for developing web applications:

Asynchronous and Event Driven:

All API's of Node.js are asynchronous and non-blocking which means it is a server which doesn't waits for any API to return the data, the server moves to the next API after calling it, a notification mechanism of events of Node.js helps server to get response from previous API calls.

Faster in Processing:

As the Node.js is built on Google Chrome's V8-JS Engine, it is very faster in executing instructions.

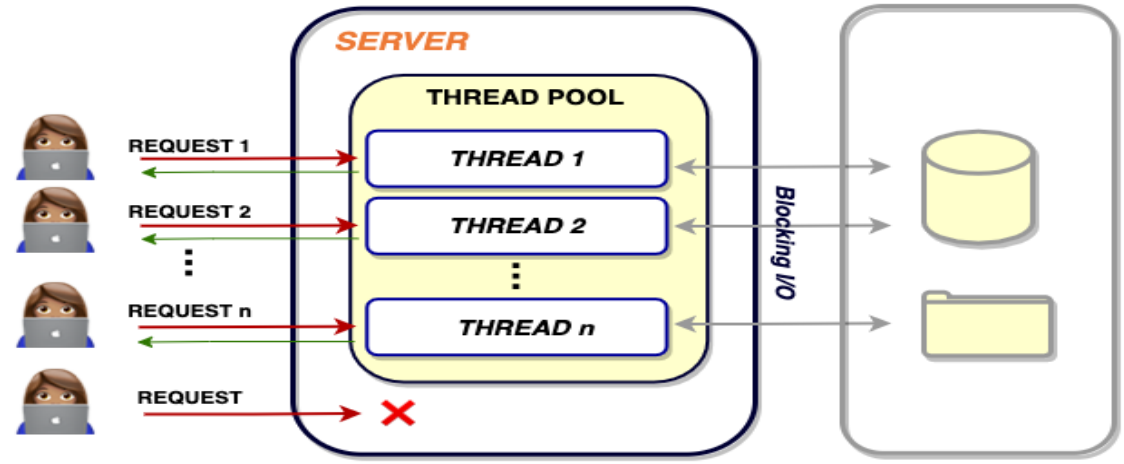
Single Threaded but Highly Scalable:

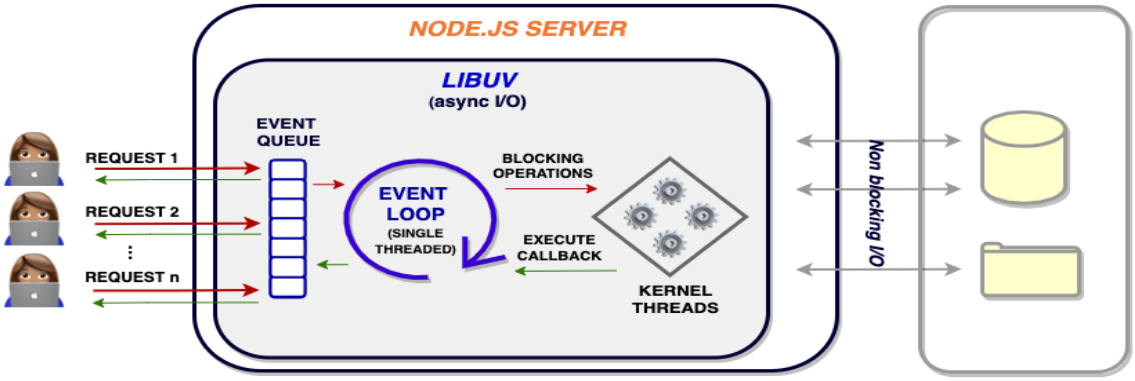
Internally Node.js uses a single threaded model with event looping, the event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable.

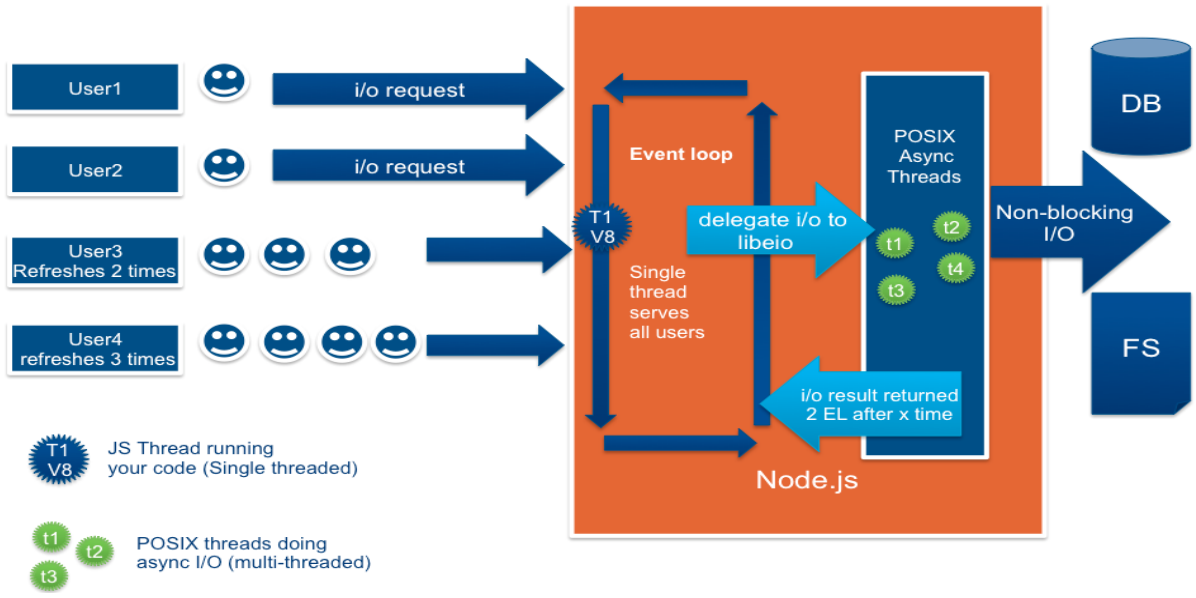
➢ No Buffering:

Node.js applications never buffer any data, these applications output data in chunks.

➢ Architecture of Node.JS:







➢ Node Package Manager (NPM):

- NPM is an online code repository publishing of open source Node.js code, it is also a command line utility for interacting with NPM repository for package installing and version management.
- It is through NPM we could work with installation (or) uninstallation (or) upgradation node modules. Through NPM we could even start or stop node server.

➢ Node.js Module:

A module in node.js is a simpler or complex functionality organized in single or multiple JavaScript files which can be reserved throughout the node.js application. Each module in node.js has its own context so that it doesn't interfere with other modules or doesn't interrupt the global scope.

It is through **NPM** we install (or) uninstall version manager of any module can be done.

Following are the basic steps which we need to create a **HTTP Node Server,**

Step-1: Download Node HTTP Module through NPM,

```
npm install http    //Downloads a HTTP Module.
```

Step-2: Create a JavaScript file and include **HTTP Module** through require method.

```
var http = require("http");      //including a Node Module.
```

Step-3: Use **"createServer"** method under HTTP, which takes a callback method as a parameter with default request and response objects.

```
http.createServer ( (req,res) => {

    …..    //Code to handle request and response

});
```

Step-4: Using response object we can specify the type of content being returned from the server.

```
res.writeHead(200,{'content-type': 'text/HTML'});
```

Using **writeHead** method we can specify the response status code along with the response text type.

Step-5: using **res.write()** method we can write the content to be responded from the server.

res.write("<Content>");

Step-6: res.end() is an predefined method through which we can send request using user response.

res.end();

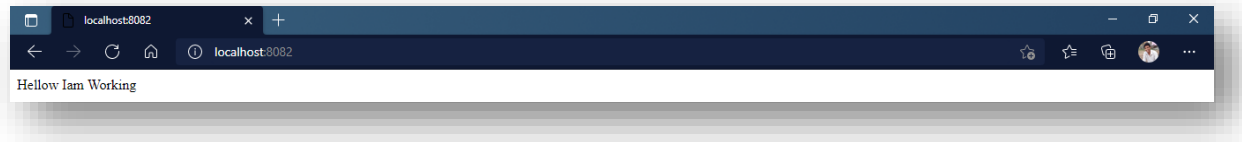
Step-7: Make the server to listen at a particular port number

server.listen(8080);

```
var http = require( "http" );


//creating a server
var server = http.createServer( ( req, res ) => {
     res.writeHead( 200, { 'content-type': 'text/html' } );
     res.write( "Hellow Iam Working" );
     res.end();
} )

//server Port Number
server.listen( 8082, () => {
     console.log( "Server is listing at 8082" );
} );
```



➢ Performing Node Operations in Node.js through fs module:

"fs" is a predefined node module through which we could able to perform read, write and append operations or any file. Following are the steps to be followed while making use of FS module,

Step-1: Download fs Module through NPM

npm i fs

Step-2: Include and create a reference for FS Module,

var fs = require("fs");

Step-3: Use the following predefined methods under FS Module through which we could able to do any file operations in files.

Fs.readFile("<filename>", (err,data) => {

//err objects holds the error if any while reading file

//Data holds the actual data of file.

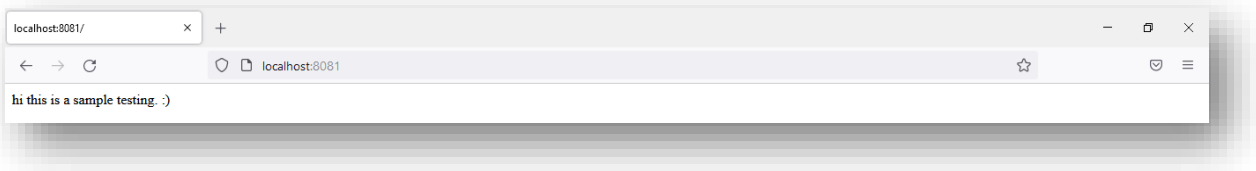
});

Ex:

```
var http = require( "http" );
var fs = require( "fs" );

var server = http.createServer( ( req, res ) => {
     var data;
     fs.readFile( 'sample.txt', ( err, data ) => {
          if ( err ) {
               data = 'error while reading the file';
          } else {
               res.writeHead( 200, { 'content-type': 'text/html' } );
               res.write( data );
               res.end();
          }
     } );
} );

server.listen( 8081, () => {
     console.log( "started" );
} );
```



➢