Questions:-

======================NODE JS Topics 3RI Techlologies=======================

1. Foundation

2. Introduction to Node JS Framework

3. Installing NodeJs

4. Using NodeJs to execute scripts.

5. Node Package Manager

6. package.json configuration

7. Global Vs Local Package Installation

8. Automating task with Gulp.

9. HTTP Protocall

10. Building HTTP Server

11. Rendering a response.

12. Using RePresentational State Transfer.

13. Nodemon

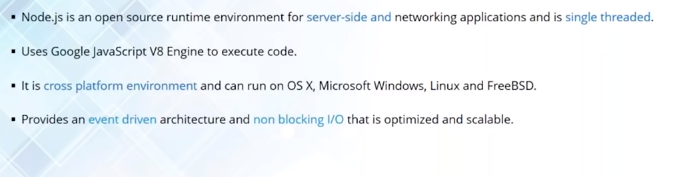
=================== EDUREKA =================

1. What is Node Js

Ans -

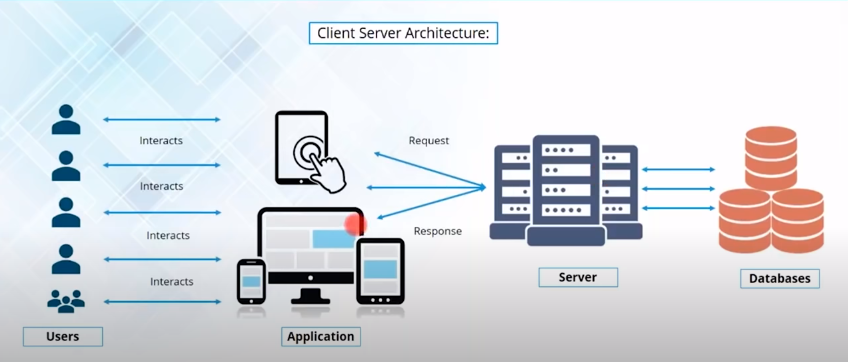
Node js is a javascript App build on top of chrome v8 engine.It uses event driven, Non blocking I/O

model which work async and on single thread architecture. It is cross platform.



**2. Why Node Js**

**Ans -** Node Js Follows Client Server Architecture.



Cleint/Users ------------→ Websites/Apps -----------→ Servers ---------------→ Databases (REQUEST)

Cleint/Users <----------- Websites/Apps <---------- Servers <-------------- Databases (RESPONSE)

OLA/Uber Example -

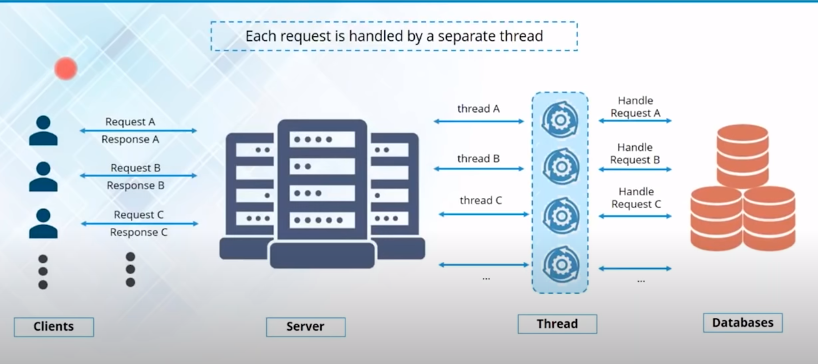
Uber customer/passenger make request to book cab using uber app. These request are sent to uber server and then further sent to uber database to check nearest available cab.

These cabs info are sent back to customer in form of response.

**3. What is single Thread and Multi Thread Modal**

**Ans : -**

Each request is handles by a sepereate thread.



Disadvantages -

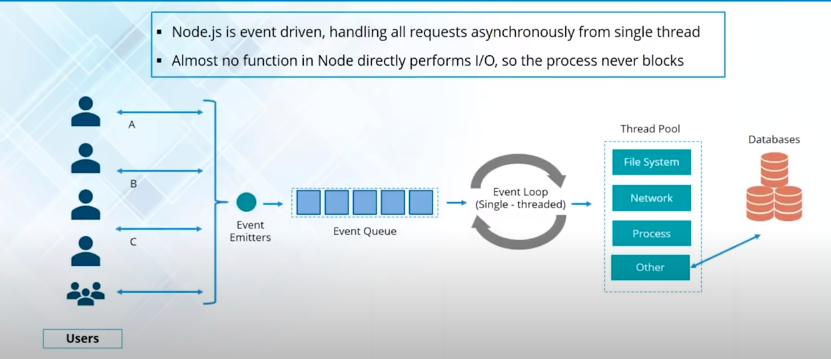
1. Number of request = (equal to) Number of threads.

2. if millions of request are coming per seconds we need millions of threads assign to each request which is complex and costly.

**Single Thread Modal**

Node js is event driven(events like – click, dbl click, mouse event etc). Handle all request asychronous using single thread.

Almost no functions directly performs Input/Output operations, So process never blocks.



In single thread, when a user perform an activity(click, submit etc), an event is generated. Every New request is a threated as an event. Event emitter allocates those events in event queue which is single thread. These event are executed/processed using event-loop machanism which is called single thread machanism. Single thread takes an event in the queue and sends in the thread pool.

There are differnet operation can be handled in Thread Pool Like

File Operations

I/O Operations

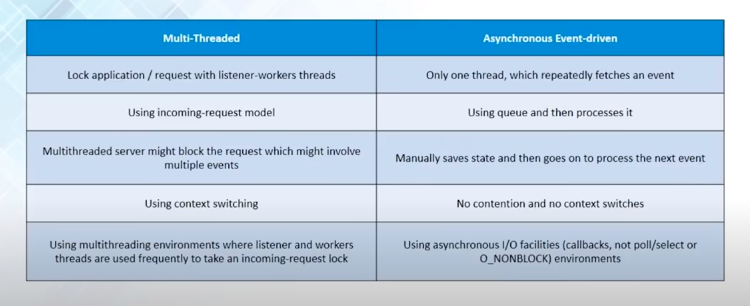
Network Opeartion

CPU Intensive Operations

The Thread in thread pool also called worker thread takes those operations asychronously(one thread handle all oprations so no blocking happens).

The processing of events in the event queue by event loop is also called event driven Modal or single thread modal.

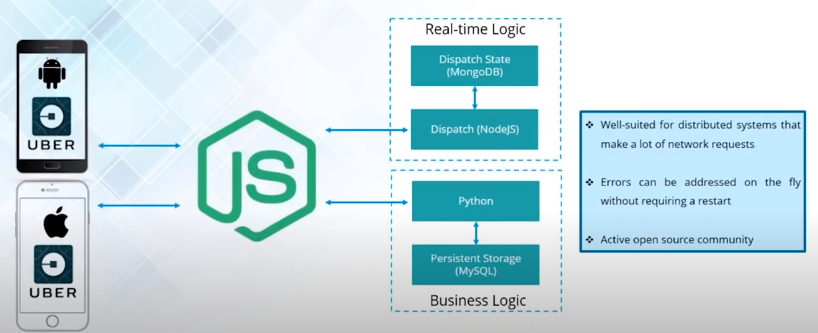
**Multi Thread Vs Event Driven**

****

incoming request model – a new thread is allocated for a new request everytime.

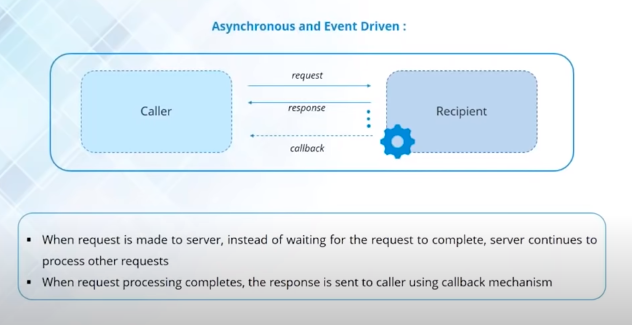
Multi thread works in synchronous.

UBER EXAMPLE -



Here there are two DB. Mongo DB for car details and MySql for driver details. Since Node is async so two request can be send simenteniously and get the response.

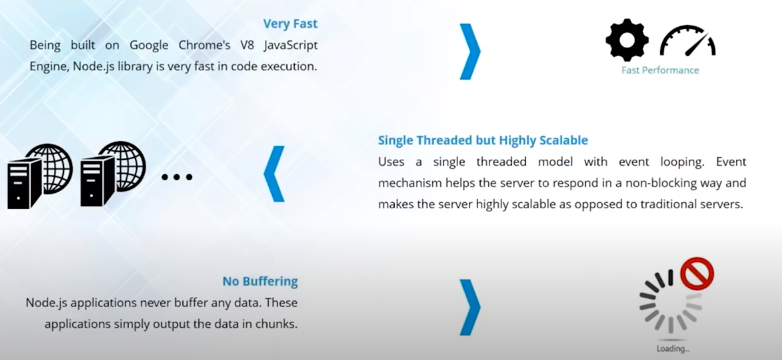
Error handing is easy and quick in node js.



NOTE : - All API in node js are single thread. So one API Never waits another API to respond.

**There is a notification machenism used in node js which is a callback function.**

**NODE JS FEATURES :-**

****

Node Js Installation -

https://nodejs.org/en/download/

**Simple node js exaple using sync(blocking)**

var fs = require('fs');

var data = fs.readFileSync(‘blog.txt’);

console.log(data.toString());

console.log(‘End Here’);

blog.txt

hello nodejs.

Output : hello nodejs

end here

**Async (Non Blocking) :-**

var fs = require('fs');

fs.readFile('blog.txt', (err, data) => {

    if(err) {

        console.log(err)

    } else {

        setTimeout(function() { // this request will go to event queue.

            console.log('show after 2 sec.');

        }, 2000);

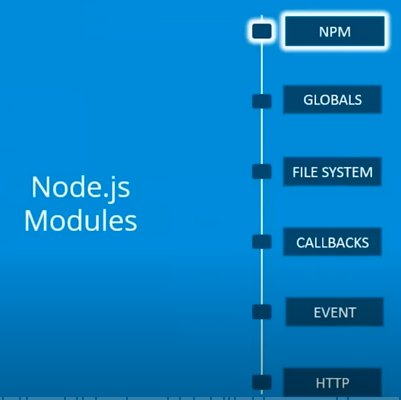
    }

});

console.log(‘start here’);

output : - start here

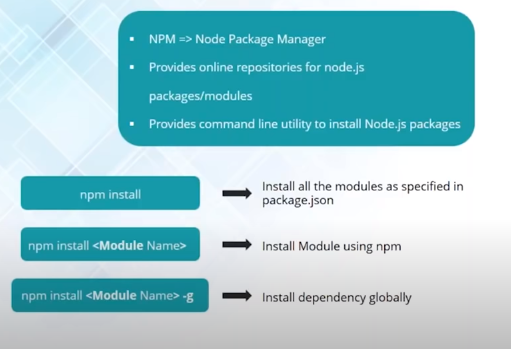
show after 2 sec.



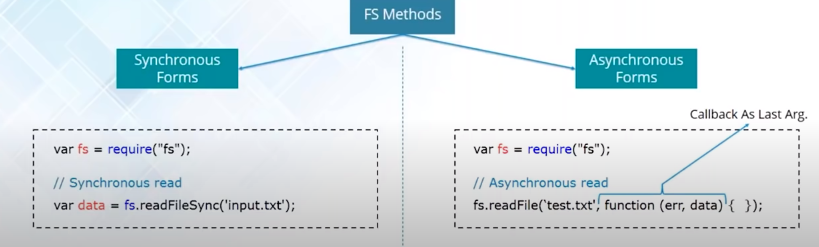
**NODE JS MODULES :-**

NPM :-

Node package manager. Online utility to download node js packages/dependencies.

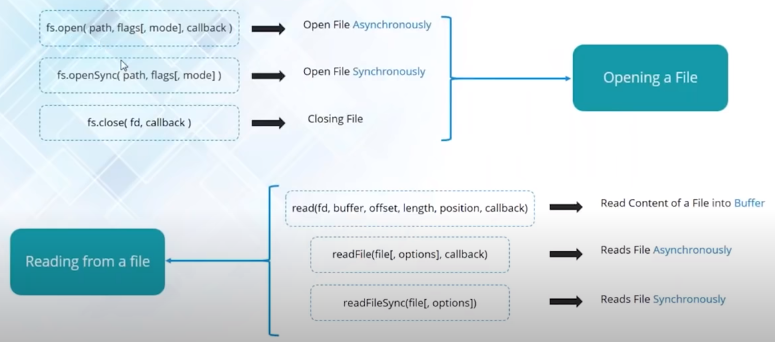


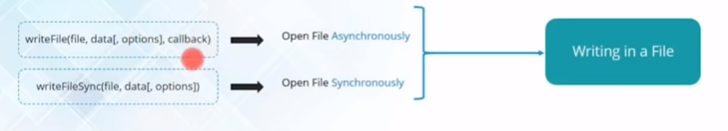
**NODE JS FILE SYSTEM MODULES :-**

****

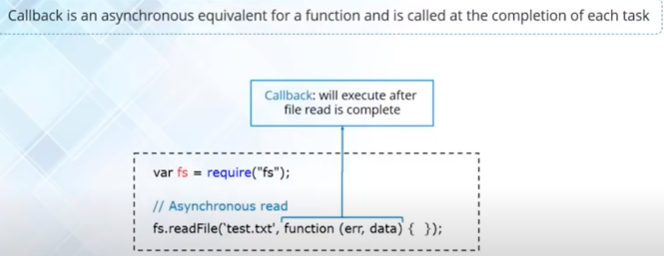
FS MODULE OPERATIONS :-

open | read | write | close

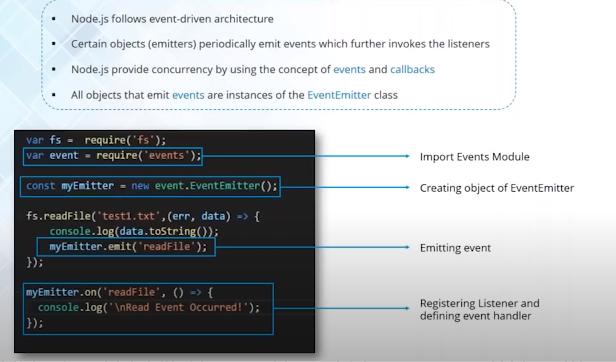


****

**NODE JS CALLBACK :-**



**NODE JS EVENTS :-**



We can emit our own event using EventEmitter() Method.

var event = require('events');

var callEmitterEvent = new event.EventEmitter();

var fs = require('fs');

fs.readFile('fileName', (err, data) => {

    console.log(data.toString());

    callEmitterEvent.emit('readBlog'); // function name

});

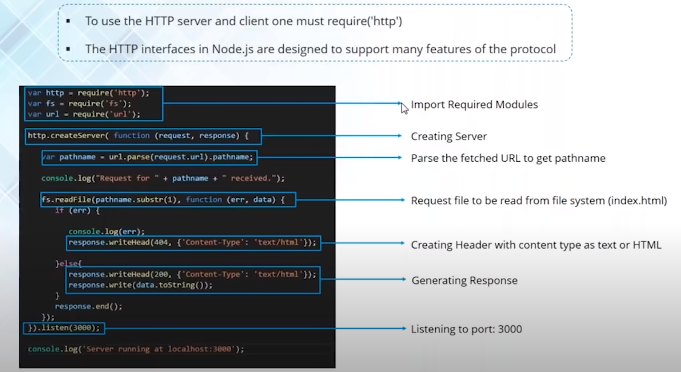
callEmitterEvent.on('readBlog', () => {

    console.log('This event is called');

});

**NODE JS HTTP MODULE :-**

Hypertext Transfer Protocall.



var http = require('http');

var fs = require('fs');

var url = require('url');

http.createServer((request, response) => {

    var pathName = url.parse(request.url).pathName;

    console.log("Request For" + pathName + "Received");

    fs.readFile(pathName.subString(1), (err, data) => {

        if(err) {

            console.log(err);

            response.writeHead(404, {'Content-Type': 'text/html'});

        } else {

            console.log(data.toString());

            response.writeHead(200, {'Content-Type': 'text/html'});

            response.write(data.toString());

        };

        response.end();

    });

}).listen(3000);

**SETTING UP NODE JS PROJECT**

1. install