Name :- Jariwala Neel Nipulkumar
Roll No :- 12
MSc ICT Sem-3
Subject :- 304 - Open Source Web Developement

Assignment - 1

1. Node js : Introduction, features, execution Architecture

- Introduction :-

Node.js is an open-source and cross-Plateform javaScript runtime enviroment. It is a popular tool for almost any kind of Project.

- node js runs the V8 Java Script engine, the core of google crome, outside the Brocaser.

Fast Data Streaming	NO Buffering
Java Script Cross Plateform	Node js Event Driven
Highly Scalable	Single Threaded

* key features of Nodejs :-

1. Asynchronous & Event Driven :-

Node js library's APIS are all asynchronous in nature. A server built with node js never waits for data from an API After accessing an API, the server moves on to the next one.

2. Single Threaded :-

nodejs employs a single threaded architecture with event looping, making it very scalable.

3. Scalable :-

nodejs addresses one of the Processing concerns in software developm Scalable, nodejs can also handle concurrent requests efficient. It has a cluster module that manages load balancing for all CPU cores.

4. Quick Execution of code :-

nodejs makes use of the V8 Java Script runtime motors, which is also used by google Crome.

5. Cross Platform Compability :-
nodejs may be used on a variety of system, including windows, unix, linux, macos & mobile devices.

6. uses Javascript :-
Javascript is used by the nodejs library, which is another important aspect of nodejs from the engineer's perspective.

7. Fast Data Streaming :-
when data is transmitted in multiple streams, processing them takes a long time. nodejs processes data at a very fast rate.

8. no buffering :-
In a nodejs application, data is never buffered.

* nodejs execution Architecture :-

1. Event - loop :-

The heart of nodejs is an event loop. It is a continous loop that constantly checks for pending I/O operation, times & call block.

2. Blocking VS non-blocking :-

- Blocking :- Read data from file
  - Show data
  - Do other Tasks

```
var data = FS.readFileAsync ("...");
Console.log (data);
Console.log ("Do other task");
```

- Non-blocking :- Read data from file when data completed, show data
  - Do other tasks

3. Callback :-

Callback is an asynchronous equivalent for a function. A callback function is called at compilation of given tasks.

4. Event Queue :-

It holds all the call backs waiting to be executing event loop continously checks event Queue for pending callbacks

5. Event Emittior :-

Event Emittior class lives in the event module. Event emitter provides multiple methods like on and emit.

(2) Write a note on nodejs module.

- modules are Javascript libraries.
- A set of function you want to include in your application.
- They are two types of module

1. Built - in
2. External

1. Built in

- nodejs has a set of built in modules which you can use without any farther installation

- To include a module use the require() function with the name of the module.

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

(ii) built-in http module

- use the createServer() method to create an HTTP server.

- The function passed into the http.createServer() method, will be executed when someone tries to access the computer on port 8080.

- Content type :-

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

(3) write a note on nodejs Packages with examples.

- In nodejs a Package refers to a collection of modules, often accompanied by Package json file that contains method data about Package, its dependency version & other information.

- Package are managed & distributed using node Package manager.

- NPM is a default Package manager for nodis that Provides vast repos.

* Creating Packages :-

- Create new directory for your Package

- Initalize your Project by running 'npm init' command.

- This will create Package.json.

* Example

i) create a new directory for Package.

ii) navigate to directory by Command Prompt.

iii) run npm init command.

iv) create index.js

```
Const generator = (min, max) =>
{
    return math.floor (math.random() );
}

module exports = get random.
```

using this Packages

```
Const getrandom = require ( random
                            generator );
```

(4) use of Package.json & Package lock.json.

- Package.json :-

Package.json is present in the root directory of any node application module & is used to defines the properties of a package.

- Project information :-
  - → Name
  - → version
  - → Dependencies
  - → Licenses
  - → main file

- How to create Package.json :-

S-1 npm init

S-2 npm start

S-3 npm run Say hello

S-4 npm test

S-5 npm install

* Package-lock.json :-

- The Purpose of Package-lock.json is to ensure that the same dependency are installed consistently across different environments, such as developement & Production environments.

- npm introduction & commands with its use.

- node Package manager Provides two main functionalities :-

- Online repositiones for node.js Packages modules.

- A Package in node.js Contains all the file you need for a module.

- npm Commands :-

- To know version :-

$npm --version

- There is a simple Syntax to install any node.js module :-

$npm install <module name>

- Install a famous node.js web framework module called express :-

$npm install express.

- Globaly installed packages!

$npm install express -g

- To check all modules installed

$npm is -g

- Save dependency in package.json for future install :-

$npm i express --save

- uninstall a module :-

$npm uninstall express

- updating a module

$npm update express.

(6) Important Properties & methods of Servelet Programs.

(1) URL :-

It Provides utilities for URL Resolution & Parsing

- URL.parse (URL String)

(2) Process :-

- Process.argv
- Process.env
- Process.cwd()
- PM2 :- It is a Popular Process for mode.
- It needs to be installed first using npm.
- npm install -g pm2

3) Readline :-

It Provides an interface for reading data from a readable Stream.

- readline. Create interface

- rl. Question (Query, Callback)

4) FSI :-

- fs. readfile (Path [options], Callback

- fs. writefile (file, data [option]; Callback

5) Event :-

eventemitter. on (event, listner)

- Eventemitter. emit (event [args])

6) Buffer :-

It Provides way to work with binary data directly.

- buffer. alloc (size[File encoding])

- buffer from (array)

7) Console
- console.log([data][...args])
- console.error([data](...args))

8) QueryString
- Provides utility for Parsing & Formatting URL Query String.
- QueryString.Parse()
- QueryString.StringIF()

9) HTTP
- It Provides functionality for creating HTTP Servers & Clients.
- HTP.createServer()
- HTP.get()
- HTP.require()

10) V8:-
- V8.getHTPStatistics()

11) OS :-

- const os = require ('os');

- const. CPUInfo = os. cpus ( );

- const. totalmemory = os. totalmem ( );

12) Zlib :-

- zlib. gzip ( )

- zlib. unzip ( )