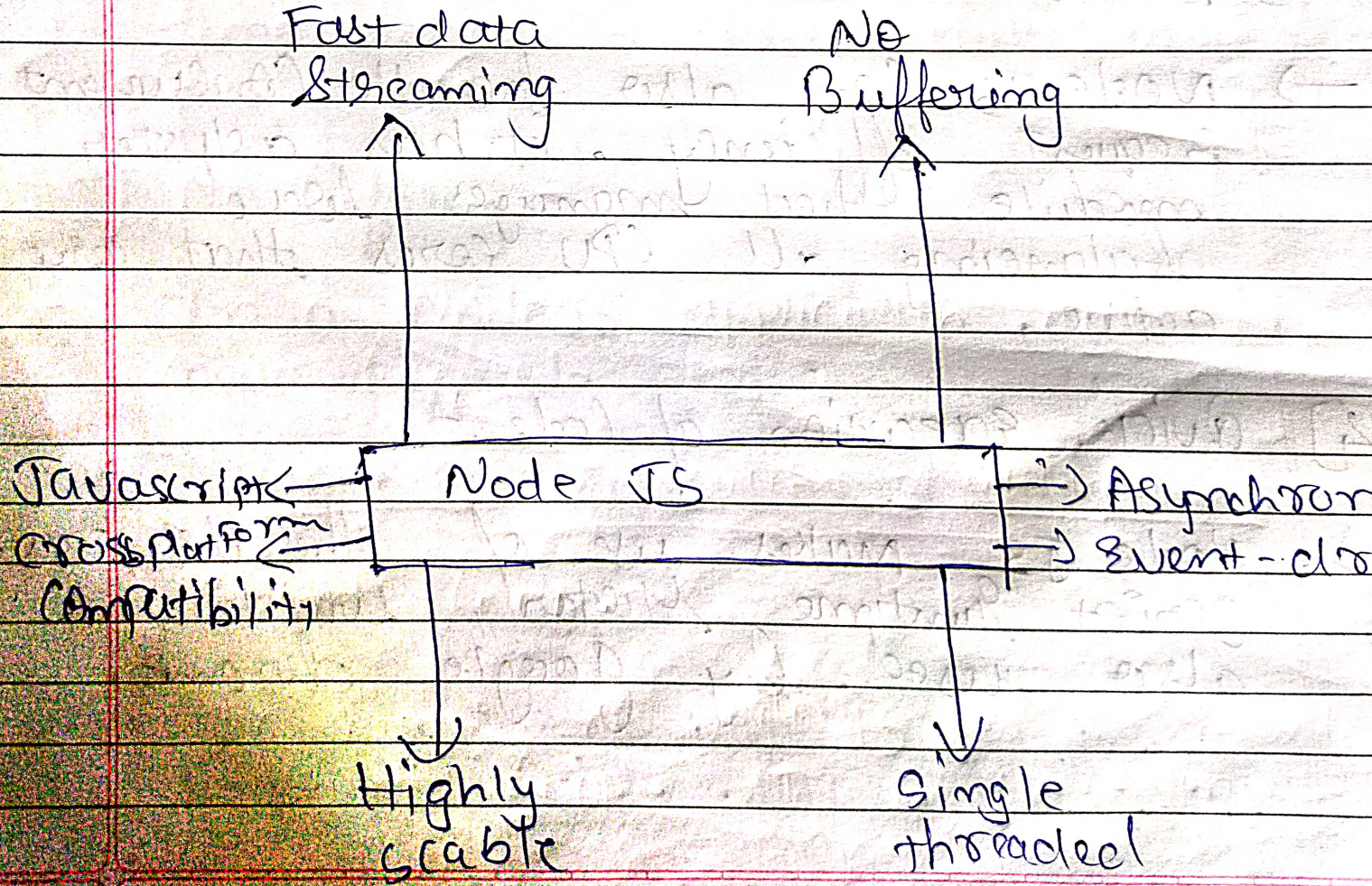
Name :- Panwala Shiv R
Roll No :-27
Subject :- Open Source web development
MSc ICT - Sem 3

Assignment :-1

1] Node JS :- Introduction, Features, execution Architecture

- Introduction :- Node-JS is an opensource and Cross Platform Javascript Runtime Environment. it is a popular tool for almost any kind of Project.

Node JS Features :-



=) Key Features :-

1] Asynchronous and Event-Driven :-

→ The Node.JS library's API's are Asynchronous nature. A Server built with Node never waits for data from and after accessing an API, the Server moves on to the next one.

2] Single-threaded :-

→ Node JS employs a single-threaded architecture with event looping making it very scalable.

3] Scalable :-

→ Node JS can also handle concurrent request efficiency. It has a cluster module that manages load balancing all CPU cores that are active.

4] Quick Execution of Code :-

→ Node JS makes use of the V8 JavaScript runtime Motor, which is also used by Google Chrome.

5] Cross - Platform Compability :-

→ Node JS may be used on a variety of system, including windows, Unix, Linux, Mac OS and Node devices.

6] Uses Javascript :-

→ Javascript is used by the Node JS library, which is another important aspect of NodeJS from the engineering's perspective.

7] Fast Data Streaming :-

→ When data is transmitted in multiple streams, processing them takes a long time. Node JS proces data very fast rate.

8] No Buffering :-

→ In a NodeJS application, data is never buffered.

* Node JS Execution Architecture :-

1] Event loop :-

→ The heart of NodeJS is an event ~~Concurrency~~ Loop. It is a continous

Event that constantly checks for pending I/O operation, timers and call blocks

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 complete Show data
-) Do other tasks.

```
FS.readFile("Text.txt", function
(err, data) {
Console.log(data); });
Console.log("Do other task");
```

3) Call back :-

-) It is an asynchronous equivalent for a function. A callback function is called at completion of given task.

4) Event queue :-

-) It holds all the 'Call back' until

Page No.
Date

to be executing, event loop continously check event queue for pending callbacks.

5) Event Emitter :

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

2) Write a note on Node JS module.

→ Modules are Javascript libraries.

→ A set of functions you want to include in your application

→ Two types of modules

→ 1) Built in

2) External

* Built in :-

→ Node JS has a set of built in modules which you can use without any further installation.

* → var http = require('http');

i) Built in .http module

-> The http module can create an HTTP server that listens to server ports and gives a response back to client

-> Use the create server() Method to create an HTTP server.

-> if the response from the HTTP Server is supposed to be displayed as HTML, you should include HTTP header with the correct Content type.

3) write a note on Node JS package with example :-

-> Package are managed and distributed using Node package manager (NPM).

-> NPM is a default package manager for NodeJS that provides vast Reposit.

=> Creating Package :-

-> Create New directory for your package initialize your project by running NPM init Command it will create Package.JSON.

```
Const generator = (min, max) =>
{
  return Math.Floor (Math.random());
};
module.exports = getRandom
```

Using this Package =>

```
Const getRandom = require (Random generator);
Const RandomNos = getRandom (10);
Console.log (randomnos);
```

4) Use of package.JSON and package Lock.JSON.

* Package.JSON :-

-> Package.JSON is present in the root directory of any Node application and is used to define the properties of a Package.

-> Project information.

-> Name

-> Version

-> dependencies

-> License

-> Main File etc.

* How to Create package.JSON :-

S1 :- NPM init

S2 :- NPM Start

S3 :- NPM run .Say hello

S4 :- NPM Test

S5 :- NPM Install

=> Package .JSON

```
"scripts" : {
"start" : "Node Event2.JS",
"Say-hello" : "echo "Hello.World"",
"bash-hello" : "bash hello.sh"
```

* Package.Lock.JSON :-

-> The purpose package-lock.JSON is to ensure that the same dependencies are installed consistently across different environment, such as development and production environment.

5] NPM introduction and Command's with its use.

-> Node package manages provides two main functionalities.

-> Online Repositinies for Node.JS packages modules.

-> A package in NodeJS contains all the files you need for a module

=> NPM Commands :-

- TO know Versions :
  $ NPM - Version

- Install a famous Node JS web Framework module called Expressed
  $ NPM Install Express

- Globally installed package $ NPM install Express -g

- Uninstalling a module :
  $ NPM Uninstall express

- Updating a module
  $ Npm update express

- Search a package name using NPM
  $ NPM search express

Page No.
Date

6) Important Properties and Methods and Relevant programs.

Ans:-

(i) URL:-

-> IT provides Utilities for URL Resolution and Parsing.

-> URL.parse (URL string [parse Query String [Slashes Denote:]])

(ii) process

-> process.argv

-> process.cwd

-> process.cwd():

- PM2: IT is a popular process for mode.

-> NPM install -g pm2

(iii) Readline:-

-> Readline.createInterface

-> OL. Question

(iv) FS :-

→ Fs.readfile (callback

→ Fs.writeFile (callback

(v) Events :-

→ Event.emitter.on

→ Event emitter.emit

(vi) BUFFer

→ BUFFer.alloc

→ BUFFer.From

(vii) Console

→ Console.log

→ Console.error

(viii) Query string :-

→ Query String.parse()

→ QueryString.Stringify()

(iv) HTTP :-

-> HTTP : create server()

-> HTTP . get()

-> HTTP . require ()

(i) v8 :-

-> v8 . get HTTP . statistic()

(ii) OS :-

-> const OS = . require ('OS.');

-> const . CPU INFO = OS . CPUS();

(iii) zlib :-

-> z.lib : gzip()

-> zlib : Unzip () .