SERVER SIDE DEVELOPMENT WITH NODE.JS

CAP 919

CA – 1

SET - B

Submitted by,
Name: Jelvin Joseph
Reg: 12000005
Roll: RDE439A28
Section: DE439
Bachelor of Computer Application

Question (1)

Complete the following questions:

a) Which function of Node.js module is used to create an HTTP server? Create a HTTP server that listens, i.e., waits on port 8080 of your computer. When port 8080 get accessed, write "Welcome to Lovely Professional University" back as a response.

b) Create a Node.js file that reads the HTML file, and return the content.

Answer:

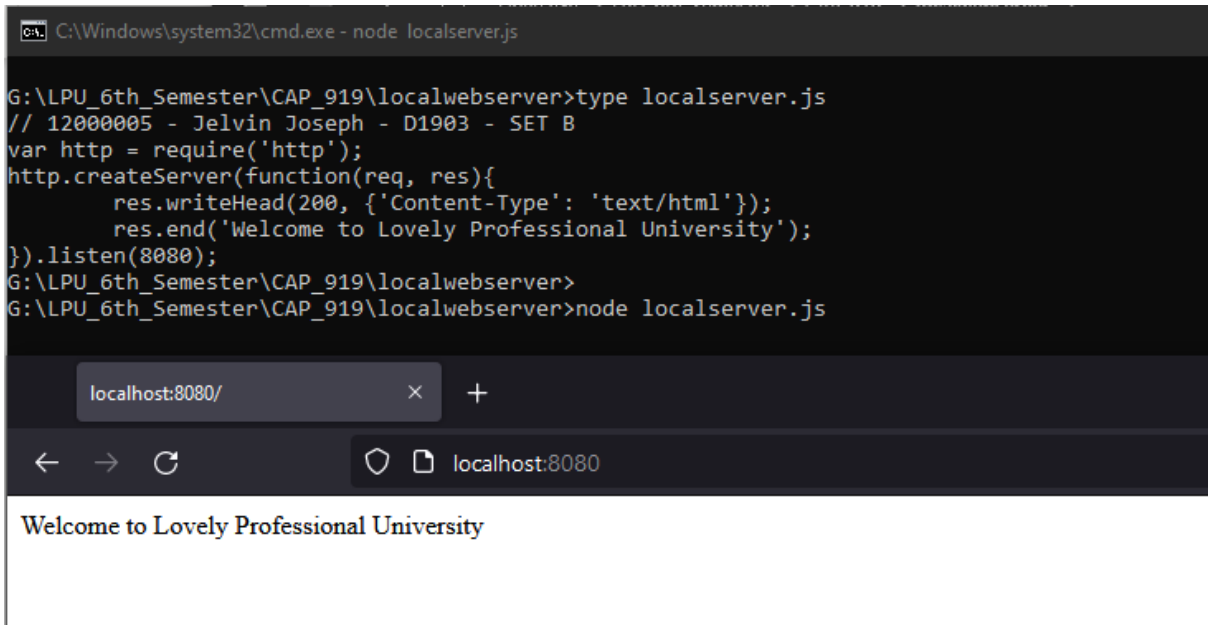
(a)

The http.createServer() function is used to create a HTTP server in Node.js. The createServer() function is part of http module which is a core module in Node.js.

We use the following code to create a http server using Node.js on port 8080 –

```
// 12000005 - Jelvin Joseph - DE439 - SET B
var http = require('http');
http.createServer(function(req, res){
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.end('Welcome to Lovely Professional University');
}).listen(8080);
```

<u>Output:</u>



(b)

To meet the requirements of the question, we need two files – A Node.js file to create a web server and a HTML file that will be read by the Node.js file and displayed to the users.

1. Node.js file

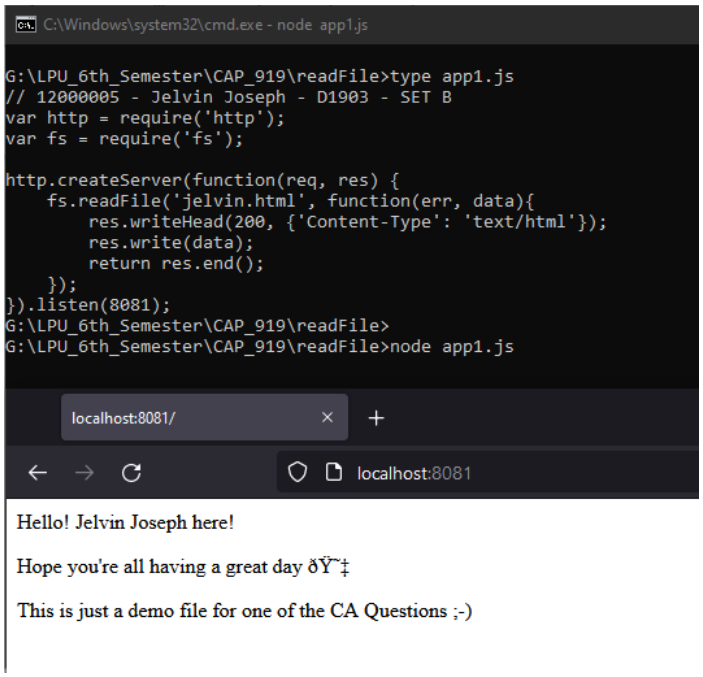
```
// 12000005 - Jelvin Joseph - DE439 - SET B
var http = require('http');
var fs = require('fs');

http.createServer(function(req, res) {
    fs.readFile('jelvin.html', function(err, data){
        res.writeHead(200, {'Content-Type': 'text/html'});
        res.write(data);
        return res.end();
    });
}).listen(8081);
```

2. HTML file

```
<html>
    <body>
        <p>Hello! Jelvin Joseph here!</p>
        <p>Hope you're all having a great day 😇</p>
        <p>This is just a demo file for one of the CA Questions ;-)</p>
    </body>
</html>
```

<u>Output:</u>



Question (2)

Complete the following questions:

a) How do you run with Nodemon? Write commands for installing nodemon both locally and globally.

b) Write a Program using Node.js to create own Module for simple calculating module that calculates various operations given below:

1. **function for adding numbers**
2. **function for subtracting numbers**
3. **function for multiplying numbers**
4. **function for dividing numbers**

Answer:

(a)

Before we can run Nodemon, we need to install it first.

To install nodemon locally we use the following npm (node package manager) command:

```
C:\Windows\system32\cmd.exe

G:\LPU_6th_Semester\CAP_919>npm install nodemon
```

To install nodemon globally we use the following npm (node package manager) command:

```
C:\Windows\system32\cmd.exe

G:\LPU_6th_Semester\CAP_919>npm install -g nodemon

changed 116 packages, and audited 117 packages in 10s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

G:\LPU_6th_Semester\CAP_919>
```

- ✓ The -g flag is used to install nodemon globally.

Nodemon documentation can be found at the following website: https://www.npmjs.com/package/nodemon

(b)

First, we create a custom module to perform simple arithmetic calculations. The code is mentioned below:

```
// 12000005 - Jelvin Joseph - DE439 - SET B
// filename: ca1q2bmod.js

function add(a, b) {
    return a + b
}

function substract(a, b) {
    return a - b
}

function multiply(a , b) {
    return a * b
}

function divide(a, b) {
    return a / b
}

module.exports.add = add
module.exports.substract = substract
module.exports.multiply = multiply
module.exports.divide = divide
```

Next, we create the Node.js file which we'll be running:

```
// 12000005 - Jelvin Joseph - DE439 - SET B

var ca1q2bmod = require('./ca1q2bmod.js')
a = 20
b = 30
c = 40
add = ca1q2bmod.add(a, b)
substract = ca1q2bmod.substract(b, a)
multiply = ca1q2bmod.multiply(c, b)
divide = ca1q2bmod.divide(c, a)

console.log("The sum of", a , "and", b , "is", add)
console.log("The difference of", b , "and", a , "is", substract)
console.log("The product of", c , "and", b , "is", multiply)
console.log("The quotient of", c , "and", a , "is", divide)
```

Output using nodemon:

```
C:\Windows\system32\cmd.exe - "node"  "C:\Users\Tatsuya\AppData\Roaming\npm\\node_modules\nodemon\bi

G:\LPU_6th_Semester\CAP_919\CA1Q2B>nodemon driver.js
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node driver.js`
The sum of 20 and 30 is 50
The difference of 30 and 20 is 10
The product of 40 and 30 is 1200
The quotient of 40 and 20 is 2
[nodemon] clean exit - waiting for changes before restart
```

Output without using nodemon:

```
C:\Windows\system32\cmd.exe

G:\LPU_6th_Semester\CAP_919\CA1Q2B>node driver.js
The sum of 20 and 30 is 50
The difference of 30 and 20 is 10
The product of 40 and 30 is 1200
The quotient of 40 and 20 is 2

G:\LPU_6th_Semester\CAP_919\CA1Q2B>
```

Question (3)

Complete the following question.

What is Node.js modules? You need to explain the below mentioned modules by taking example for each:

1. **Core Modules (HTTP Module, File System Module)**
2. **Local Modules (Create your own module and export that into your Node.js application)**
3. **Third-party Modules (Yargs Module)**

Answer:

When we talk about Node.js modules the first thing that comes into mind is that it's similar to a JavaScript library. A module can be defined as a group of functions that are grouped together and can be used in a Node.js application by using the require() function. Modules are used to provide reusable codes in a Node.js application.

Modules in Node.js can be classified into three types, they are:

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

- Core Modules

  Core Modules are modules that are shipped by default with Node.js. When you install Node.js on your system, these core modules are installed by default. These modules are also known as “Built-in” modules as you don’t need to install separately.

  You do however need to import the module first before you can use it, and to import a core module like any other module you need to use the require() function.

  Example:

  var http = require(‘http’)

  Firstly, we are going to look at the “http” module which is a core module that comes shipped with Node.js. The http module is used to transfer data over Hyper Text Transfer Protocol. We can use http methods such as http.createServer() to create a Node.js web server on our local system. We can then access this server with the help of a web browser and going to http://localhost:<port_number_of_server>

  Example:

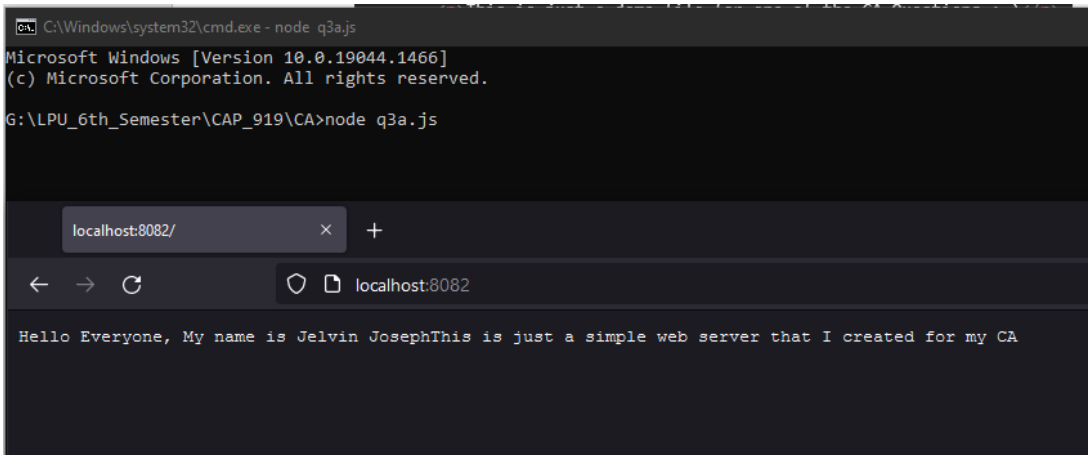
  http://localhost:8080

```
// 12000005 - Jelvin Joseph - DE439 - SET B

var http = require('http'); //importing http core module

//creating a Node.js web server on port 8082 using createServer method
http.createServer(function (req, res) {
    //2 lines below are used to write lines to the browser
    res.write('Hello Everyone, my name is Jelvin Joseph') //my name ofcourse
    res.write('This is just a simple web server that I created for my CA');
    res.end(); //response end
}).listen(8082); //specifying the local listening port
```

Output:



File System Module

```
// 12000005 - Jelvin Joseph - DE439 - SET B
var http = require('http');
var fs = require('fs');

http.createServer(function(req, res) {
    fs.readFile('lpu.html', function(err, data){
        res.writeHead(200, {'Content-Type': 'text/html'});
        res.write(data);
        return res.end();
    });
}).listen(8088);
```

- Local Module
  Local modules are the modules we create ourselves. They are not a core module and they are not a third-party module. These modules are made specifically by the programmer to meet certain requirements for the Node.js application. There are situations where the programmer cannot find a certain functionality in any of the core modules or third-party modules, it is at this time when we make our own "local modules" to add these functionalities that are needed but not available as part of a core module or third-party module.

  - Creating a local module

```
// 12000005 - Jelvin Joseph - DE439 - SET B
// filename: q3bmod.js
function area(length, breadth) {
    return length * breadth
}
module.exports.area = area
```

- Creating a Node.js application to use the above local module

```
// 12000005 - Jelvin Joseph - DE439 - SET B
var q3bmod = require('./q3bmod.js')

console.log("Area of rectangle with 200 and 300:", q3bmod.area(200,
300))
console.log("Area of rectangle with 100 and 200:", q3bmod.area(100,
200))
console.log("Area of rectangle with 123 and 456:", q3bmod.area(123,
456))
```

Output:

```
C:\Windows\system32\cmd.exe

G:\LPU_6th_Semester\CAP_919\CA>node q3b.js
Area of rectangle with 200 and 300: 60000
Area of rectangle with 100 and 200: 20000
Area of rectangle with 123 and 456: 56088

G:\LPU_6th_Semester\CAP_919\CA>
```

- Third-party Modules
  Third-party modules are modules which are not part of the core modules and is not a local module. Third-party modules are downloaded from the internet using Node Package Manager (NPM). There are two ways to install a third-party module on your system.
  1. Local installation
  2. Global installation

  Local installation is where you install the module only into your project directory. It is not available to be used in any other project directories.

  Global installation is where you install the module globally on your system. When you install a module globally on your system, then you can use the module in any project directory without needing to install the module again and again.

  Eg:

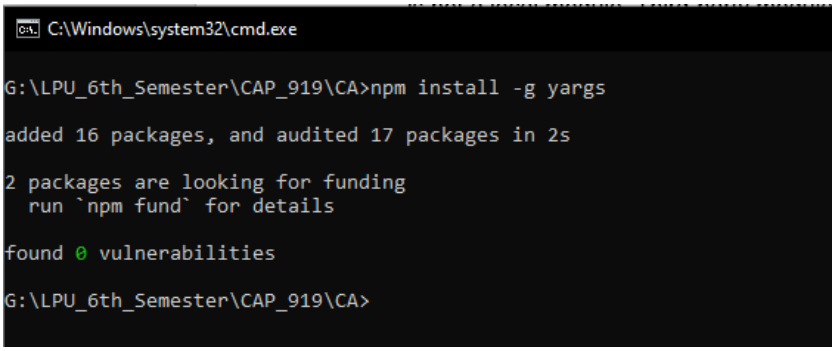
  npm install @angular/cli

  npm install -g @angular/cli

  In the above example, the first installation is a local installation and the second installation is a global installation.

We'll look at another example which is "Yargs Module". To install Yargs globally, we use the following command in command line prompt.

npm i -g yargs



X ---------- X