Lesson 04 Demo 01

Installing and Creating a Node.js App

Objective: To install Node.js and create a server-side application using its extensive libraries and asynchronous features

Tools required: Visual Studio Code

Prerequisites: Basic Linux commands

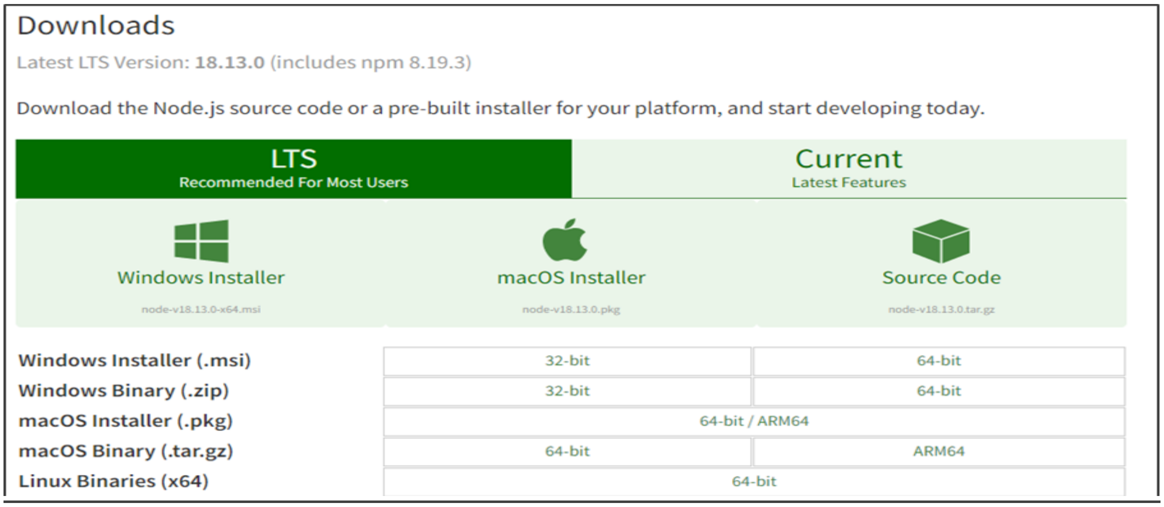
Steps to be followed:

1. Download Node.js
2. Install Node.js
3. Install the NPM package manager
4. Create a Node.js app

Note: Follow steps 1, 2, and 3 if Node.js and NPM are not installed within the lab or system.

Step 1: Download Node.js

1.1 Visit the official Node.js website, **https://nodejs.org/en/download/,** to download an executable file according to the operating system



Downloads

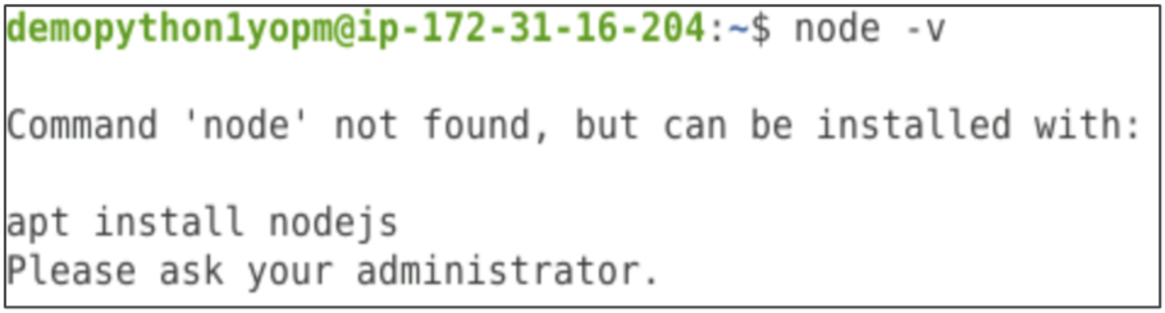
Latest LTS Version: **18.13.0** (includes npm 8.19.3)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	

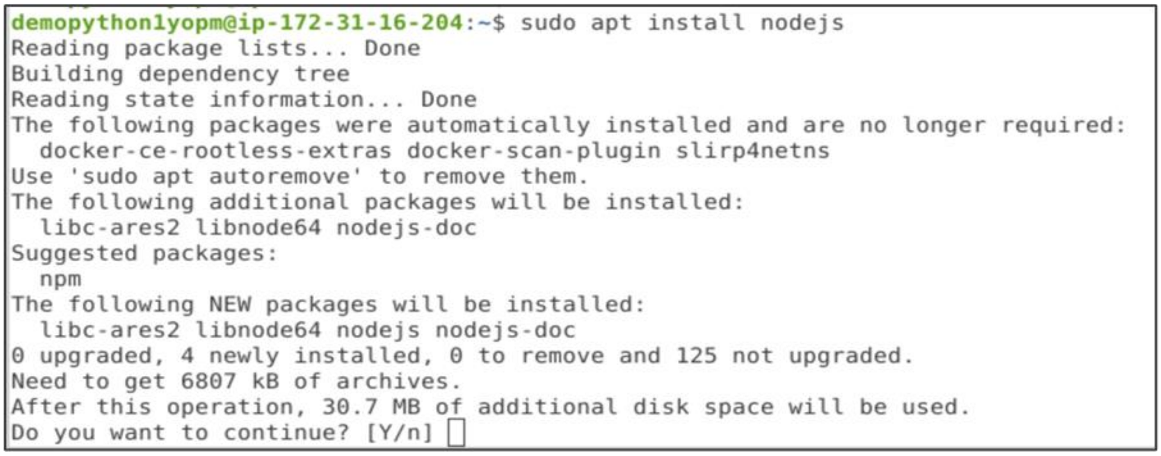
Step 2: Install Node.js

2.1 Open the terminal and execute the following command to check whether Node.js is installed in the system:
node -v



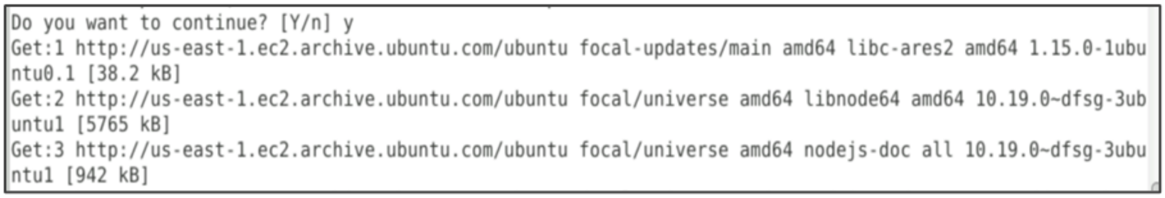
```
demopython1yopm@ip-172-31-16-204:~$ node -v

Command 'node' not found, but can be installed with:

apt install nodejs
Please ask your administrator.
```

2.2 Execute the following command to install Node.js on the system:
sudo apt install nodejs



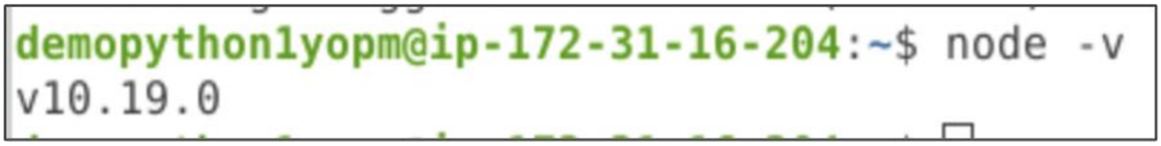
```
demopython1yopm@ip-172-31-16-204:~$ sudo apt install nodejs
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-ce-rootless-extras docker-scan-plugin slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libc-ares2 libnode64 nodejs-doc
Suggested packages:
  npm
The following NEW packages will be installed:
  libc-ares2 libnode64 nodejs nodejs-doc
0 upgraded, 4 newly installed, 0 to remove and 125 not upgraded.
Need to get 6807 kB of archives.
After this operation, 30.7 MB of additional disk space will be used.
Do you want to continue? [Y/n] 
```

2.3 Enter **y** to confirm the installation



```
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 libc-ares2 amd64 1.15.0-1ubuntu0.1 [38.2 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 libnode64 amd64 10.19.0~dfsg-3ubuntu1 [5765 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 nodejs-doc all 10.19.0~dfsg-3ubuntu1 [942 kB]
```

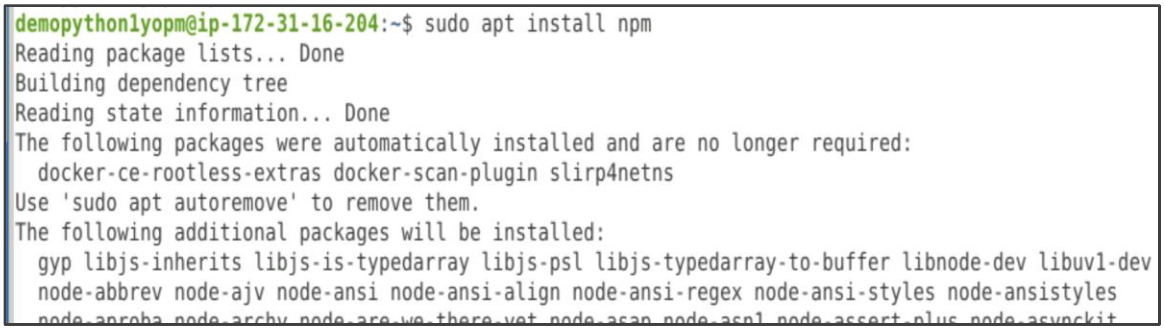
2.4 Execute the following command to check the installed version of Node.js:
node -v



```
demopython1yopm@ip-172-31-16-204:~$ node -v
v10.19.0
```

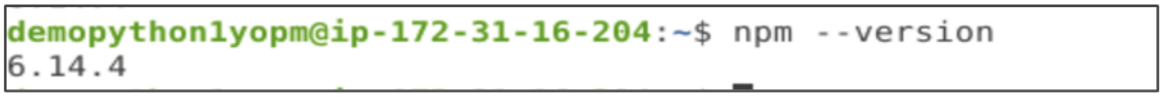
Step 3: Install the NPM package manager

3.1 Execute the following command to install the NPM package manager:
sudo apt install npm



```
demopython1yopm@ip-172-31-16-204:~$ sudo apt install npm
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-ce-rootless-extras docker-scan-plugin slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  gyp libjs-inherits libjs-is-typedarray libjs-psl libjs-typedarray-to-buffer libnode-dev libuv1-dev
  node-abbrev node-ajv node-ansi node-ansi-align node-ansi-regex node-ansi-styles node-ansistyles
  node-aproba node-archy node-are-we-there-yet node-asap node-asn1 node-assert-plus node-asynckit
```

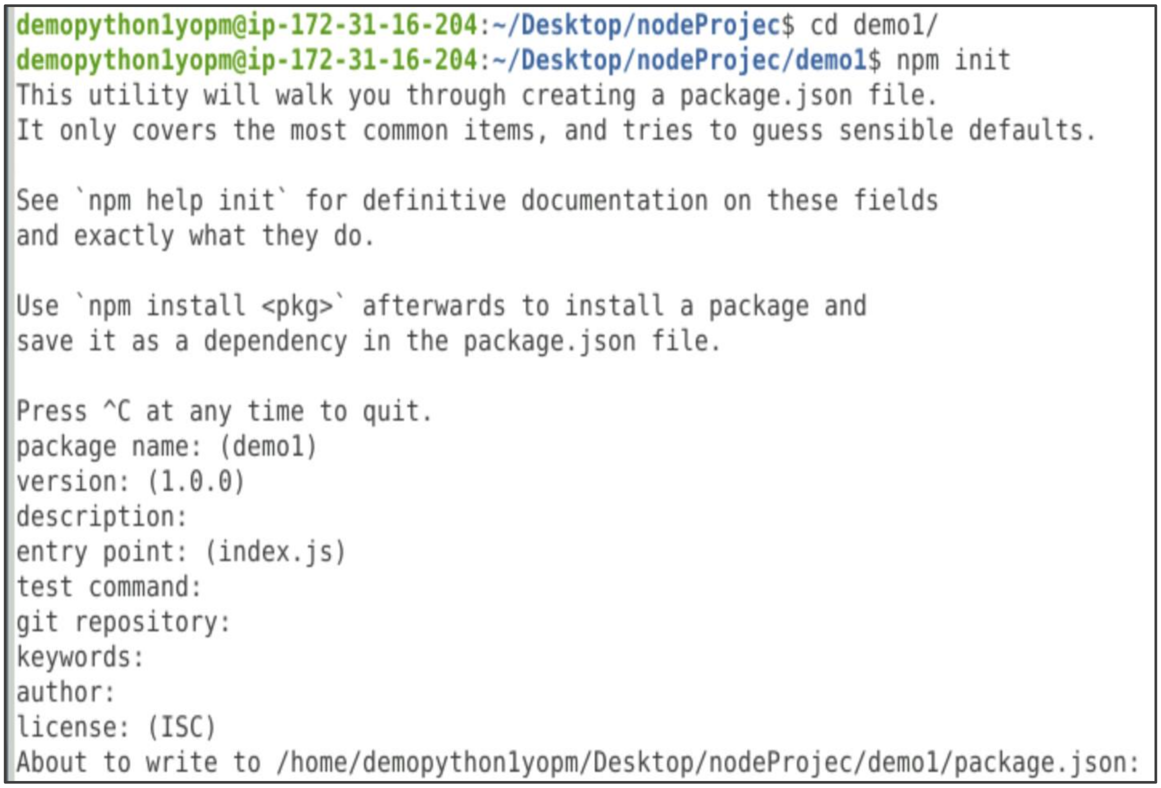
3.2 Verify the installation of the NPM package manager by executing the following command:
npm --version



```
demopython1yopm@ip-172-31-16-204:~$ npm --version
6.14.4
```

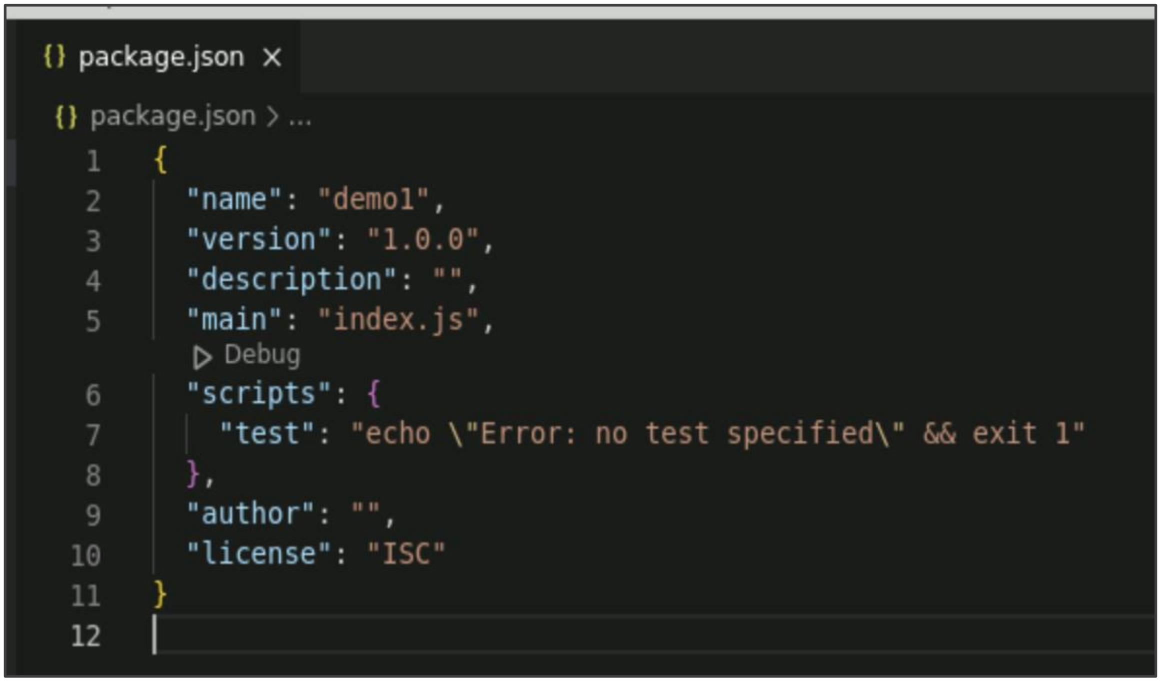
Step 4: Create a Node.js app

4.1 Navigate inside the project directory and initialize the node environment using the following commands:
cd demo1
npm init



```
demopython1yopm@ip-172-31-16-204:~/Desktop/nodeProjec$ cd demo1/
demopython1yopm@ip-172-31-16-204:~/Desktop/nodeProjec/demo1$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (demo1)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to /home/demopython1yopm/Desktop/nodeProjec/demo1/package.json:
```

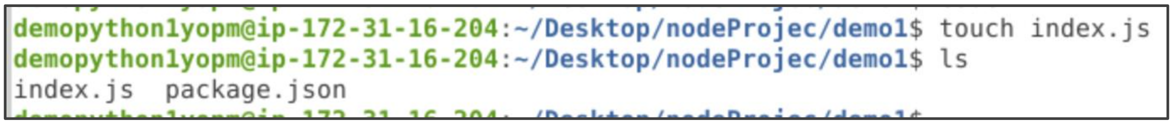
Note: Once the NPM is initialized, it will ask for details like the package name, version, description, and so on. These details can be skipped by using the **-y** flag in the **npm init** command.



```
{
  "name": "demo1",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

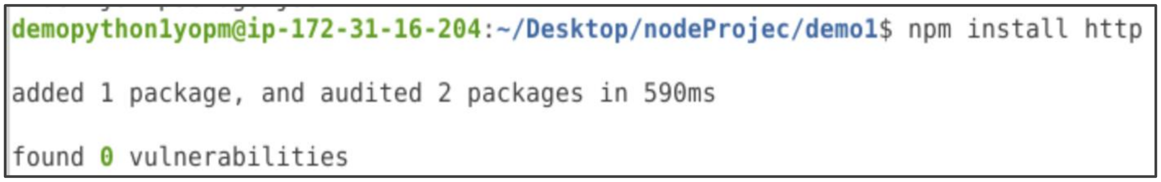
After execution, the **package.json** file will be created in the folder. This file contains all the information about the installed packages in the Node.js project.

4.2 Execute the following command to create a new file named **index.js**:
touch index.js



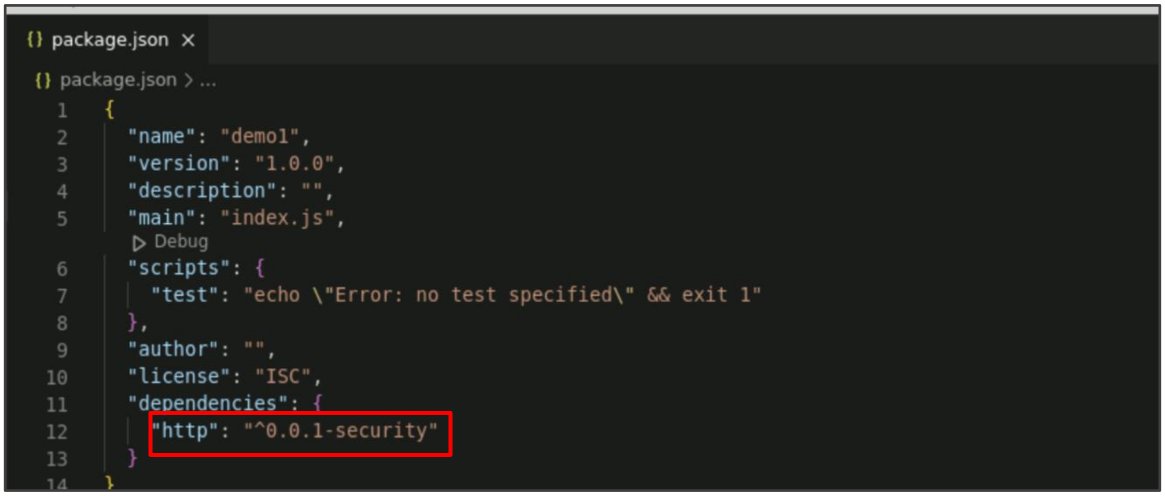
```
demopython1yopm@ip-172-31-16-204:~/Desktop/nodeProjec/demo1$ touch index.js
demopython1yopm@ip-172-31-16-204:~/Desktop/nodeProjec/demo1$ ls
index.js  package.json
```

4.3 Execute the following command to install the HTTP package within the Node.js project:
npm install http



```
demopython1yopm@ip-172-31-16-204:~/Desktop/nodeProjec/demo1$ npm install http

added 1 package, and audited 2 packages in 590ms

found 0 vulnerabilities
```

4.4 Open the **package.json** file in VS Code to view the information regarding the installed HTTP package

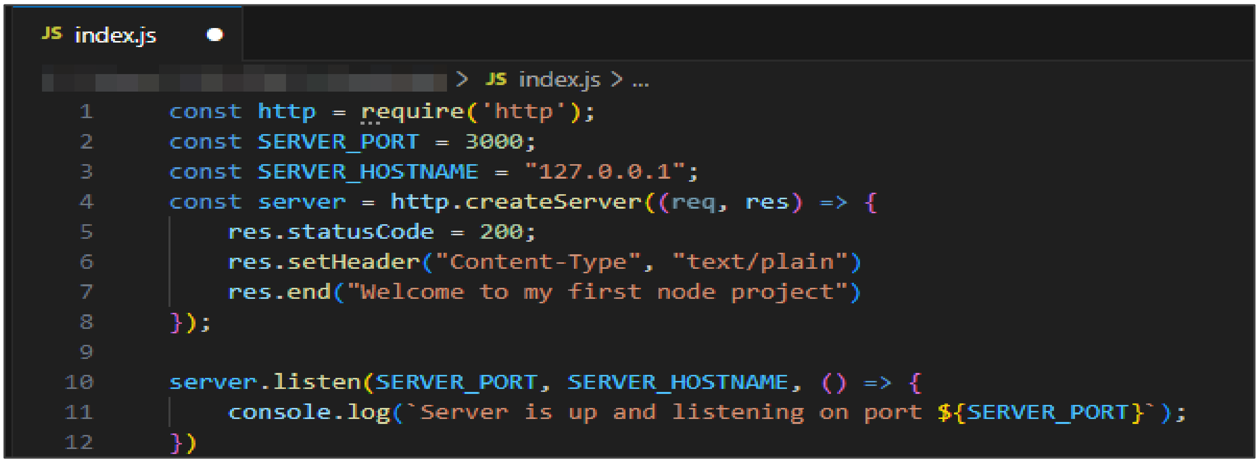


```
{
  "name": "demo1",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "http": "^0.0.1-security"
  }
}
```

4.5 Write the following code using the HTTP package within the index.js file:

const http = require('http');
const SERVER_PORT = 3000;
const SERVER_HOSTNAME = "127.0.0.1";
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader("Content-Type", "text/plain")
  res.end("Welcome to my first node project")
});

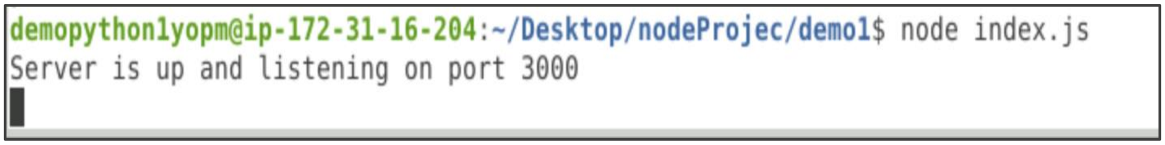
server.listen(SERVER_PORT, SERVER_HOSTNAME, () => {
  console.log(`Server is up and listening on port ${SERVER_PORT}`);
})



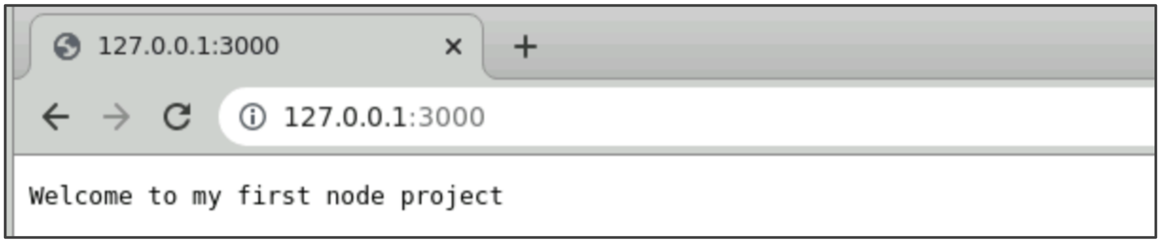
```
const http = require('http');
const SERVER_PORT = 3000;
const SERVER_HOSTNAME = "127.0.0.1";
const server = http.createServer((req, res) => {
    res.statusCode = 200;
    res.setHeader("Content-Type", "text/plain")
    res.end("Welcome to my first node project")
});

server.listen(SERVER_PORT, SERVER_HOSTNAME, () => {
    console.log(`Server is up and listening on port ${SERVER_PORT}`);
})
```

4.6 Run the following command to start the server:

node index.js



```
demopython1yopm@ip-172-31-16-204:~/Desktop/nodeProjec/demo1$ node index.js
Server is up and listening on port 3000
```



```
Welcome to my first node project
```

By following these steps, you have successfully installed Node.js and created a server-side application using its extensive libraries and asynchronous features.