



# **DEVOPS INTERNSHIP**

## **2021**

**Assignment**  
**NodeJS & ReactJS**

**Name: Bijay Aashish Bhatta**

## Question Number 1 - NodeJS:

- a. To install NodeJS 16 on CentOS 8:

```
$ curl -fsSL https://rpm.nodesource.com/setup_16.x | sudo bash -  
$ sudo yum install -y nodejs
```

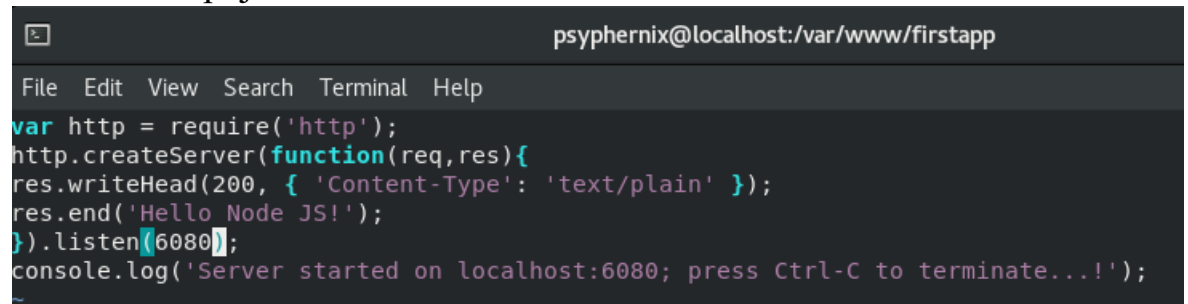
- b. To create two API using port 6080 and 7080 with messages “Hello Node JS” and “Node JS installed successfully” respectively:

- i) First App (port 6080):

```
$ sudo mkdir -p /var/www/firstapp  
$ cd /var/www/firstapp/  
$ sudo npm init
```

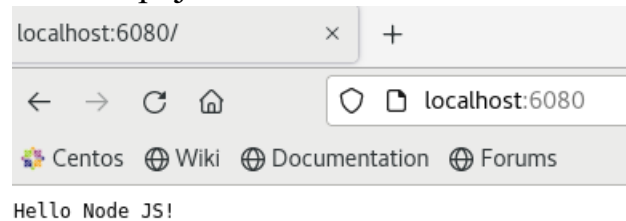
```
[psyphernix@localhost firstapp]$ sudo npm init  
This utility will walk you through creating a package.json file.  
It only covers the most common items, and tries to guess sensible defaults.  
  
See `npm help init` for definitive documentation on these fields  
and exactly what they do.  
  
Use `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: (firstapp)  
version: (1.0.0)  
description: First node js app for learning purpose  
entry point: (index.js)  
test command:  
git repository:  
keywords:  
author: Bijay Aashish Bhatta  
license: (ISC)  
About to write to /var/www/firstapp/package.json:  
  
{  
  "name": "firstapp",  
  "version": "1.0.0",  
  "description": "First node js app for learning purpose",  
  "main": "index.js",  
  "scripts": {  
    "test": "echo \"Error: no test specified\" && exit 1"  
  },  
  "author": "Bijay Aashish Bhatta",  
  "license": "ISC"  
}  
  
Is this OK? (yes)  
[psyphernix@localhost firstapp]$
```

\$ sudo vim api.js



```
psyphe@localhost:~/firstapp$ sudo vim api.js
var http = require('http');
http.createServer(function(req, res){
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('Hello Node JS!');
}).listen(6080);
console.log('Server started on localhost:6080; press Ctrl-C to terminate...!');
```

\$ node api.js

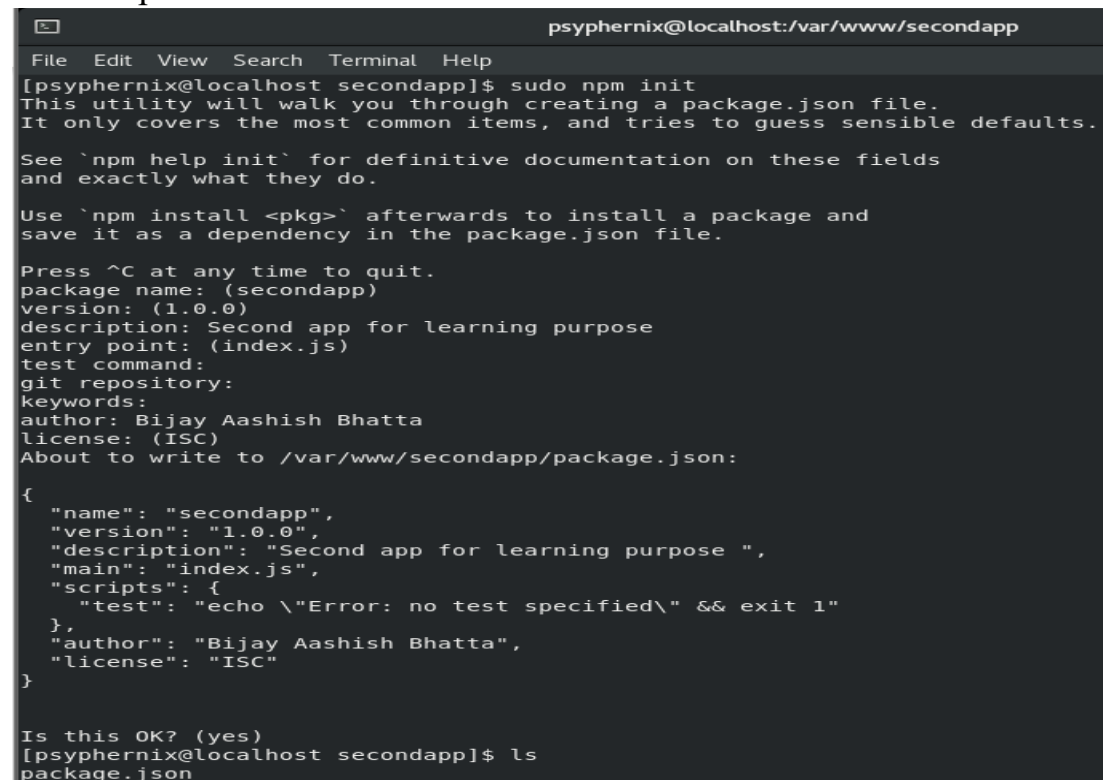


ii) Second app (port 6080):

\$ sudo mkdir -p /var/www/secondapp

\$ cd /var/www/secondapp

\$ sudo npm init



```
psyphe@localhost:~/secondapp$ sudo 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.

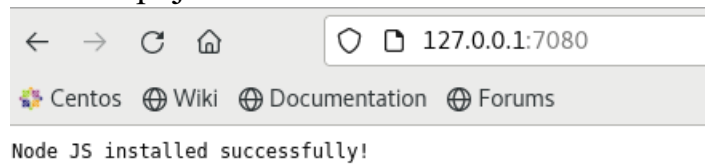
Press ^C at any time to quit.
package name: (secondapp)
version: (1.0.0)
description: Second app for learning purpose
entry point: (index.js)
test command:
git repository:
keywords:
author: Bijay Aashish Bhatta
license: (ISC)
About to write to /var/www/secondapp/package.json:
{
  "name": "secondapp",
  "version": "1.0.0",
  "description": "Second app for learning purpose ",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Bijay Aashish Bhatta",
  "license": "ISC"
}

Is this OK? (yes)
[psyphe@localhost secondapp]$ ls
package.json
```

\$ sudo vim api.js

```
psyphernix@localhost:/var/www/secondapp
File Edit View Search Terminal Help
var http = require('http');
http.createServer(function(req,res){
res.writeHead(200, { 'Content-Type': 'text/plain' });
res.end('Node JS installed successfully!');
}).listen(7080);
console.log('Server started on localhost:7080; press Ctrl-C to terminate...!');
```

\$ node api.js



c. To install PM2:

\$ curl -sL https://dl.yarnpkg.com/rpm/yarn.repo | sudo tee  
/etc/yum.repos.d/yarn.repo

\$ sudo yum install yarn

\$ sudo npm i -g pm2

\$ sudo pm2 start api.js -i 4 (from both firstapp and secondapp  
directory)

```
[PM2] Spawning PM2 daemon with pm2_home=/root/.pm2
[PM2] PM2 Successfully daemonized
[PM2] Starting /var/www/firstapp/api.js in cluster_mode (4 instances)
[PM2] Done.
```

id	name	mode	♾	status	cpu	memory
0	api	cluster	0	online	0%	46.1mb
1	api	cluster	0	online	0%	44.6mb
2	api	cluster	0	online	0%	41.6mb
3	api	cluster	0	online	0%	39.8mb

```
[psyphernix@localhost firstapp]$
```

```
[psyphernix@localhost secondapp]$ sudo pm2 start api.js -i 4
[PM2] Starting /var/www/secondapp/api.js in cluster_mode (4 instances)
[PM2] Done.
```

id	name	mode	♾	status	cpu	memory
0	api	cluster	0	online	0%	45.3mb
1	api	cluster	0	online	0%	45.7mb
2	api	cluster	0	online	0%	45.9mb
3	api	cluster	0	online	0%	45.7mb
4	api	cluster	0	online	0%	46.1mb
5	api	cluster	0	online	0%	44.8mb
6	api	cluster	0	online	0%	41.3mb
7	api	cluster	0	online	0%	31.8mb

```
[psyphernix@localhost secondapp]$
```

- d. To delete all clusters (4 of both app's) one-by-one:

```
$ sudo pm2 delete 0
```

```
$ sudo pm2 delete 1
```

```
$ sudo pm2 delete 2
```

```
$ sudo pm2 delete 3
```

```
$ sudo pm2 delete 4
```

```
$ sudo pm2 delete 5
```

```
$ sudo pm2 delete 6
```

```
$ sudo pm2 delete 7
```

id	name	mode	♾	status	cpu	memory
4	api	cluster	0	online	0%	45.1mb
5	api	cluster	0	online	0%	47.9mb
6	api	cluster	0	online	0%	45.5mb
7	api	cluster	0	online	0%	45.3mb

```
[psyphernix@localhost secondapp]$ sudo pm2 delete 4
[PM2] Applying action deleteProcessId on app [4](ids: [ '4' ])
[PM2] [api](4) ✓
```

id	name	mode	♾	status	cpu	memory
5	api	cluster	0	online	0%	48.2mb
6	api	cluster	0	online	0%	45.5mb
7	api	cluster	0	online	0%	45.6mb

```
[psyphernix@localhost secondapp]$ sudo pm2 delete 5
[PM2] Applying action deleteProcessId on app [5](ids: [ '5' ])
[PM2] [api](5) ✓
```

id	name	mode	♾	status	cpu	memory
6	api	cluster	0	online	0%	44.8mb
7	api	cluster	0	online	0%	45.6mb

```
[psyphernix@localhost secondapp]$ sudo pm2 delete 6
[PM2] Applying action deleteProcessId on app [6](ids: [ '6' ])
[PM2] [api](6) ✓
```

id	name	mode	♾	status	cpu	memory
7	api	cluster	0	online	0%	45.1mb

```
[psyphernix@localhost secondapp]$ sudo pm2 delete 7
[PM2] Applying action deleteProcessId on app [7](ids: [ '7' ])
[PM2] [api](7) ✓
```

id	name	mode	♾	status	cpu	memory
----	------	------	---	--------	-----	--------

```
[psyphernix@localhost secondapp]$
```

## Question Number 2 - ReactJS:

- a. To install ReactJS:

```
$ sudo npm -g install create-react-apps
```

```
[psyphernix@localhost secondapp]$ sudo npm -g install create-react-app
npm WARN deprecated tar@2.2.2: This version of tar is no longer supported, and will not receive security
updates. Please upgrade asap.

added 67 packages, and audited 68 packages in 6s

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

2 high severity vulnerabilities

Some issues need review, and may require choosing
a different dependency.

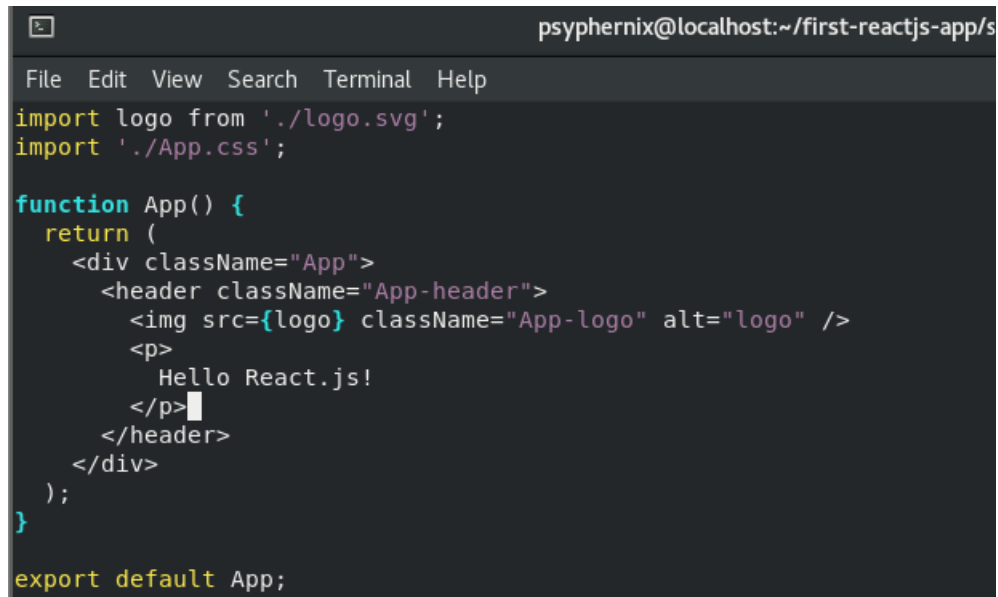
Run `npm audit` for details.
[psyphernix@localhost secondapp]$
```

- b. To create a React Application and print message "Hello React.js":

```
$ create-react-app first-reactjs-app
```

```
$ cd first-reactjs-app/src
```

```
$ sudo vim App.js
```



```
File Edit View Search Terminal Help
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Hello React.js!
        </p>
      </header>
    </div>
  );
}

export default App;
```

```
$ cd ..
```

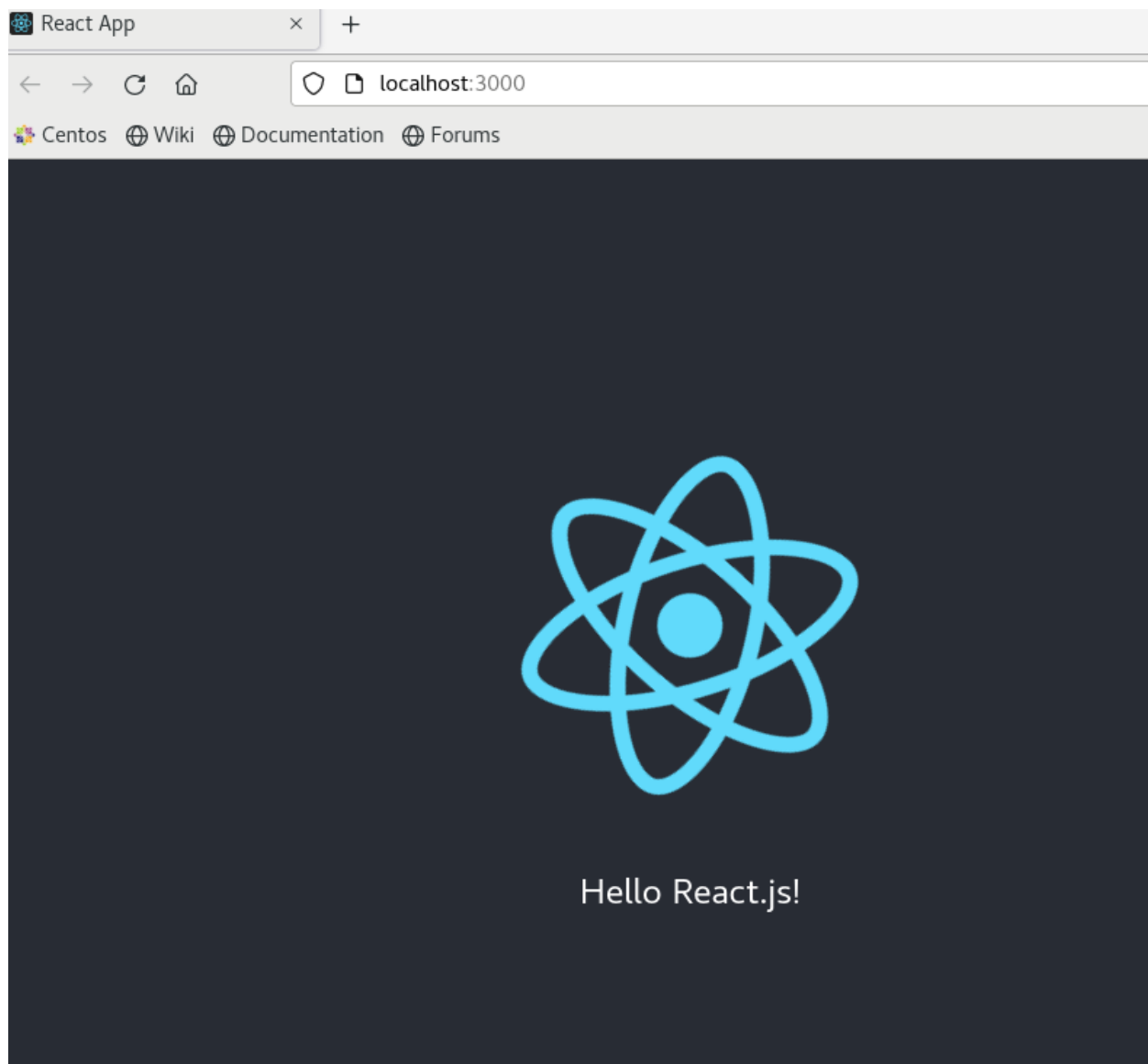
```
$ npm start
```

```
psyphernix@localhost:~/first-reactjs-app
File React App View Search Terminal Help
Compiled successfully!

You can now view first-reactjs-app in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.1.65:3000

Note that the development build is not optimized.
To create a production build, use yarn build.
```



- c. To change the default port from 3000 to 3001:

```
$ sudo vim .env
```

```
psyphernix@localhost:~/first-reactjs-app
File Edit View Search Terminal Help
PORT=3001
```

```
$ npm start
```

```
psyphernix@localhost:~/first-reactjs-app
File Edit View React App Terminal Help
Compiled successfully!

You can now view first-reactjs-app in the browser.

Local:      http://localhost:3001
On Your Network: http://192.168.1.65:3001

Note that the development build is not optimized.
To create a production build, use yarn build.
```

