Caching Server

Operating System: Ubuntu

AWS or local installation

The Verification, QR Code, and Caching server can either be all installed on separate computers, or all run on the same computer. These instructions will outline how to set up the Caching server as a separate server, and at the end there will be instructions on how to modify the server if you want to run them all on the same computer for simplicity.

Each Node.js server requires `npm` be installed, as well as all the dependencies for the project.

If you have already set up the QRCodeServer and intend on running all Node.js servers on the one computer, then skip to the end of this document.

Install npm

The first step is to install npm so that the Node.js servers can run. Run the following command in a new terminal

```
sudo apt-get install npm
```

We now need to download the web applications from GitHub. Perform the following command in a terminal in your `home` directory.

```
git clone https://github.com/SwinburneBlockchain/WebApplications
```

This copies the Node.js web server from the GitHub link. If you plan on running each Node.js server on individual servers then you may delete the `QRCodeServer.js` and `QRCodeServer.js` files that were downloaded along with the `CachingServer.js` file. Otherwise continue onto the next step.

To initialise this application, you will need to move into the WebApplications folder created when you cloned the GitHub link, and initialise npm in this folder.

```
cd WebApplications
npm init
```

You will now need to edit the `package.json` file that was downloaded from GitHub. Open up the `package.json` file, and delete the following line:

```
"mongodb": "^3.2.16"
```

Now follow these instructions on installing MongoDB on Ubuntu

https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-16-04

If you are having trouble starting MongoDB after following these instructions, see Appendix 1.

You should now be able to run the following command which installs the remaining dependencies for the web server:

```
npm install
```

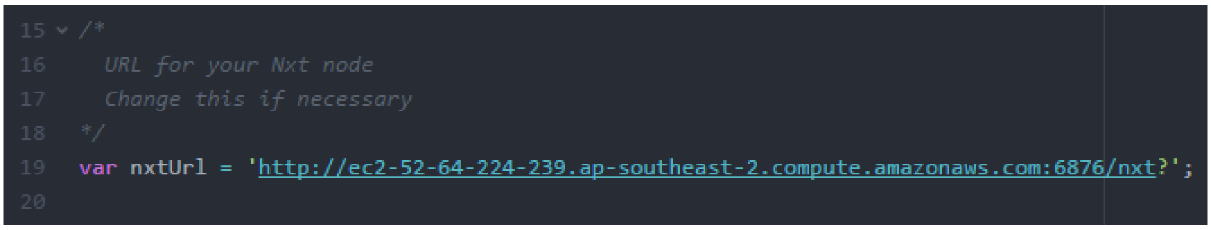
Once everything has been installed, run the Verification Server by entering the following command:

```
node CachingServer.js
```

You may get errors if you have not yet set up the Verification Server and the QR Code server, as the Caching Server contacts both of these servers. If so, please complete setting up of these servers.

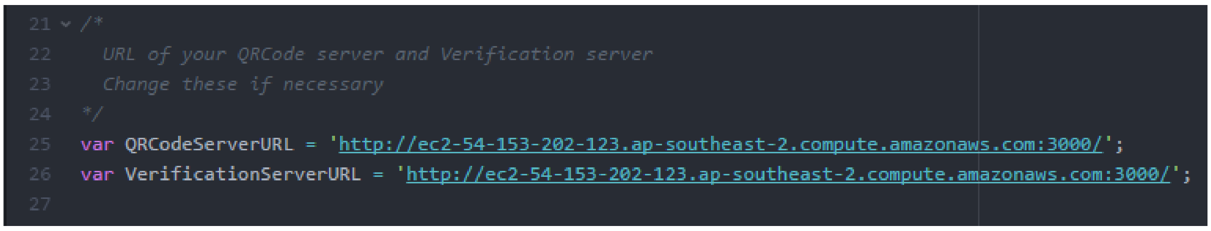
Modifying Program Constants

There are several variables at the beginning of `CachingServer.js` which need to be modified to suit your setup. The first is on line 19, and is the Nxt address of your online Blockchain node created in the “Set up Nxt Nodes” documentation.



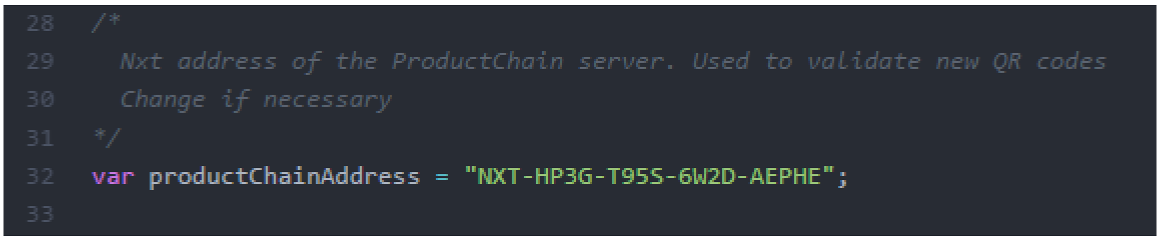
```
15 /*
16   URL for your Nxt node
17   Change this if necessary
18 */
19 var nxtUrl = 'http://ec2-52-64-224-239.ap-southeast-2.compute.amazonaws.com:6876/nxt?';
20
```

The second modifications are on lines 25 and 26. These are the URLs of your QR Code Server and your Verification Server. If you have set up these on the same computer then they will be the same (similar to below).



```
21 /*
22   URL of your QRCode server and Verification server
23   Change these if necessary
24 */
25 var QRCodeServerURL = 'http://ec2-54-153-202-123.ap-southeast-2.compute.amazonaws.com:3000/';
26 var VerificationServerURL = 'http://ec2-54-153-202-123.ap-southeast-2.compute.amazonaws.com:3000/';
27
```

Finally on line 32, you will need to enter the Nxt address of your ProductChain account, created in “Set up Nxt Nodes” documentation. If you have used the same Nxt address that we used, then this will be the same.

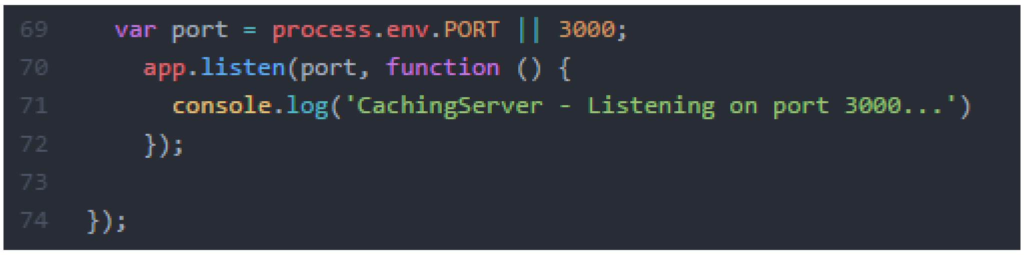


```
28 /*
29   Nxt address of the ProductChain server. Used to validate new QR codes
30   Change if necessary
31 */
32 var productChainAddress = "NXT-HP3G-T95S-6W2D-AEPHE";
33
```

Running all Node.js servers on the same computer

There are several modifications that need to be made to the `CachingServer.js` file if you wish to run all the servers on the same computer.

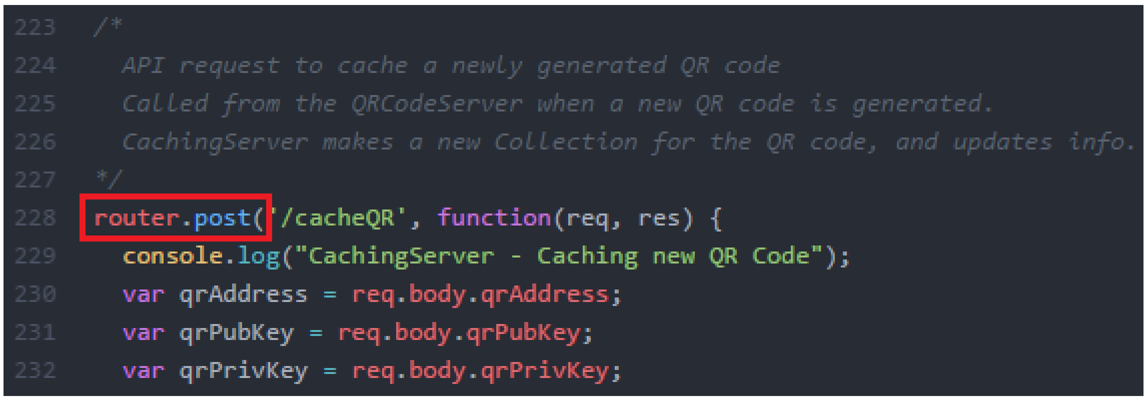
The first change is that you will need to comment out lines 69 - 74. This is the part which starts the Node.js server on port 3000. You cannot have multiple Node.js servers running on port 3000.



```
69      var port = process.env.PORT || 3000;
70        app.listen(port, function () {
71          console.log('CachingServer - Listening on port 3000...')
72        });
73
74    });
```

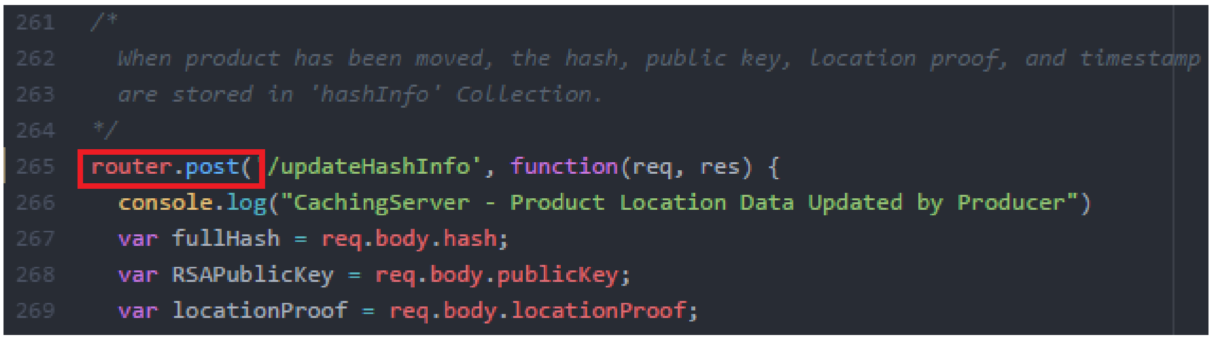
Next you will need to change several lines so that it they read `router.post` instead of `app.post`.

Line #228



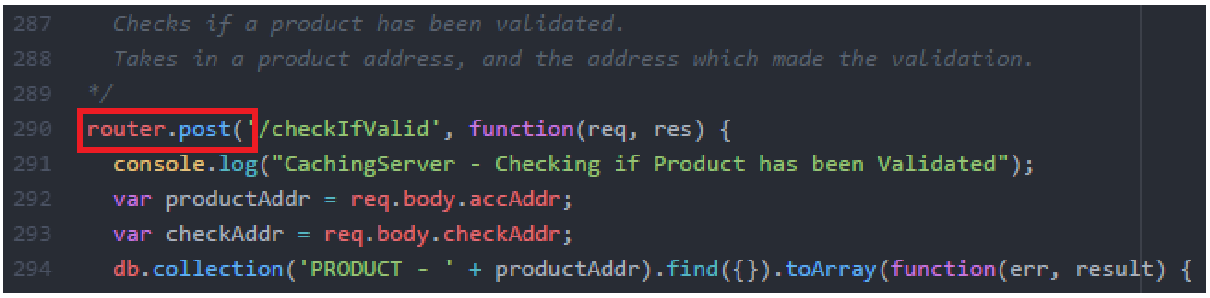
```
223   /*
224     API request to cache a newly generated QR code
225     Called from the QRCodeServer when a new QR code is generated.
226     CachingServer makes a new Collection for the QR code, and updates info.
227   */
228   router.post('/cacheQR', function(req, res) {
229     console.log("CachingServer - Caching new QR Code");
230     var qrAddress = req.body.qrAddress;
231     var qrPubKey = req.body.qrPubKey;
232     var qrPrivKey = req.body.qrPrivKey;
```

Line #265



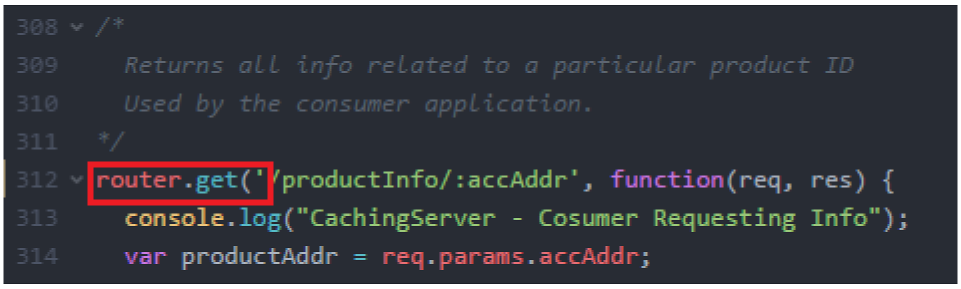
```
261   /*
262     When product has been moved, the hash, public key, location proof, and timestamp
263     are stored in 'hashInfo' Collection.
264   */
265   router.post('/updateHashInfo', function(req, res) {
266     console.log("CachingServer - Product Location Data Updated by Producer")
267     var fullHash = req.body.hash;
268     var RSAPublicKey = req.body.publicKey;
269     var locationProof = req.body.locationProof;
```

Line #290



```
287     Checks if a product has been validated.
288     Takes in a product address, and the address which made the validation.
289   */
290   router.post('/checkIfValid', function(req, res) {
291     console.log("CachingServer - Checking if Product has been Validated");
292     var productAddr = req.body.accAddr;
293     var checkAddr = req.body.checkAddr;
294     db.collection('PRODUCT - ' + productAddr).find({}).toArray(function(err, result) {
```

Line #312



```
308  /*
309    Returns all info related to a particular product ID
310    Used by the consumer application.
311  */
312  router.get('/productInfo/:accAddr', function(req, res) {
313    console.log("CachingServer - Cosumer Requesting Info");
314    var productAddr = req.params.accAddr;
```

That is all the modifications required to the `CachingServer.js` file. Once all additional Node.js servers have been set up, the program can be run by running `node QRCodeServer.js` from the terminal.

Appendix 1 – Trouble with MongoDB

This may resolve issues with starting MongoDB

Open a terminal and enter the following command:

```
sudo nano /etc/systemd/system/mongodb.service
```

Now copy the following text into the newly created file:

```
[Unit]
Description=High-performance, schema-free document-oriented
database
After=network.target

[Service]
User=mongodb
ExecStart=/usr/bin/mongod --quiet --config /etc/mongod.conf

[Install]
WantedBy=multi-user.target
```

You should now be able to start MongoDB through the following command:

```
sudo systemctl start mongodb
```

To permanently enable Mongodb, enter the following command:

```
sudo systemctl enable mongodb
```