# PROJECT 17
# INTERLEDGER AND RIPPLE
***SPRING TRIMESTER 2022***
**By**
***NIKHIL PATEL***

**Overview :** In this lab, I will study how to avoid money laundering by utilizing ripple, a complex cryptocurrency utilized by banks in 12-28-17 where one protocol, interledger, is used between two separate ledgers.

We use rippler money to generate a payment between the shopper and the consumer. We obtained a ripper address from the website ripple.com/build/xrp-test-net/. Here, we are constructing two addresses, one for the customer and one for the shopper. We may use one for the customer and leave the other.

## Procedure :

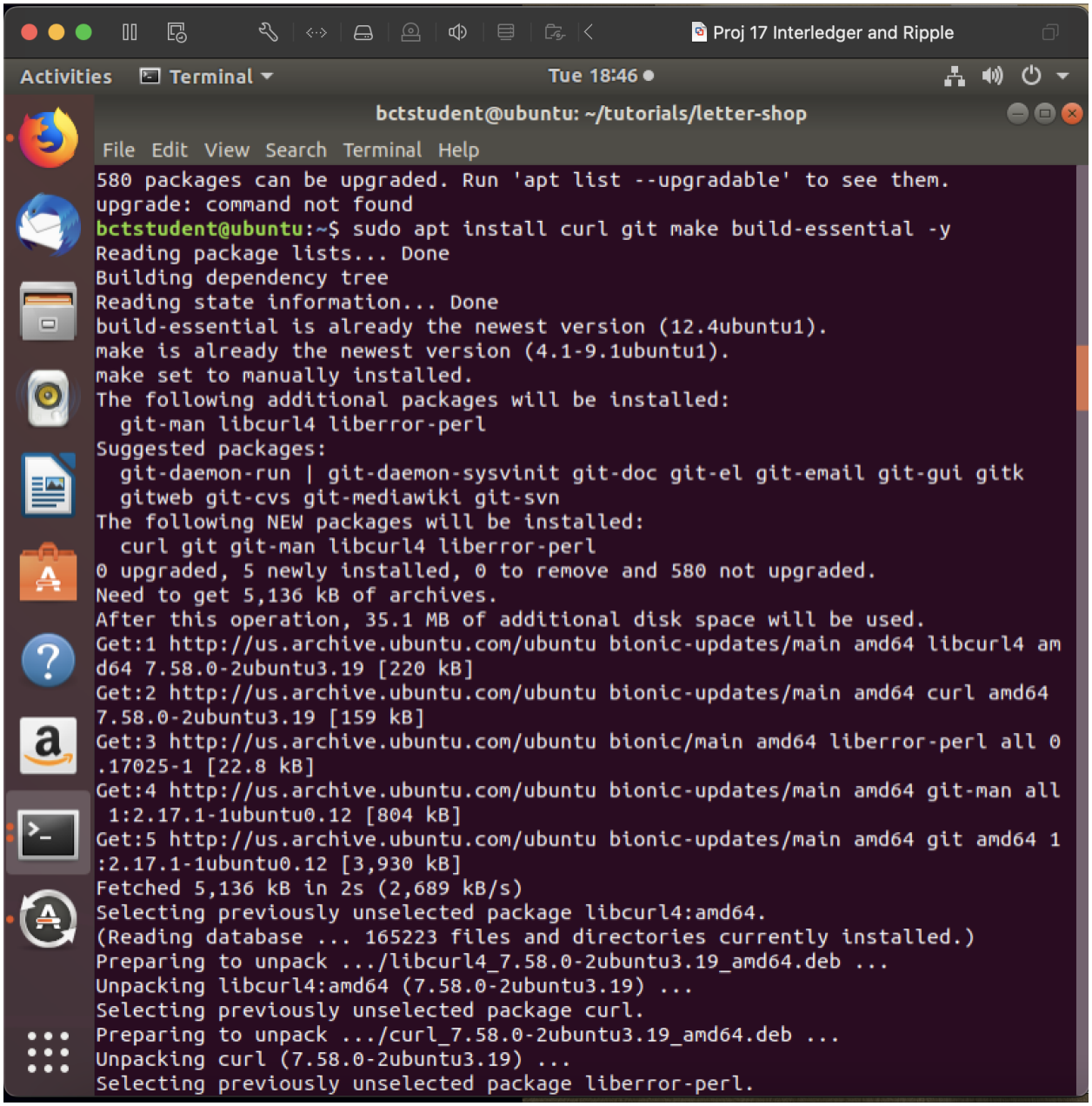
1. **Install Node**

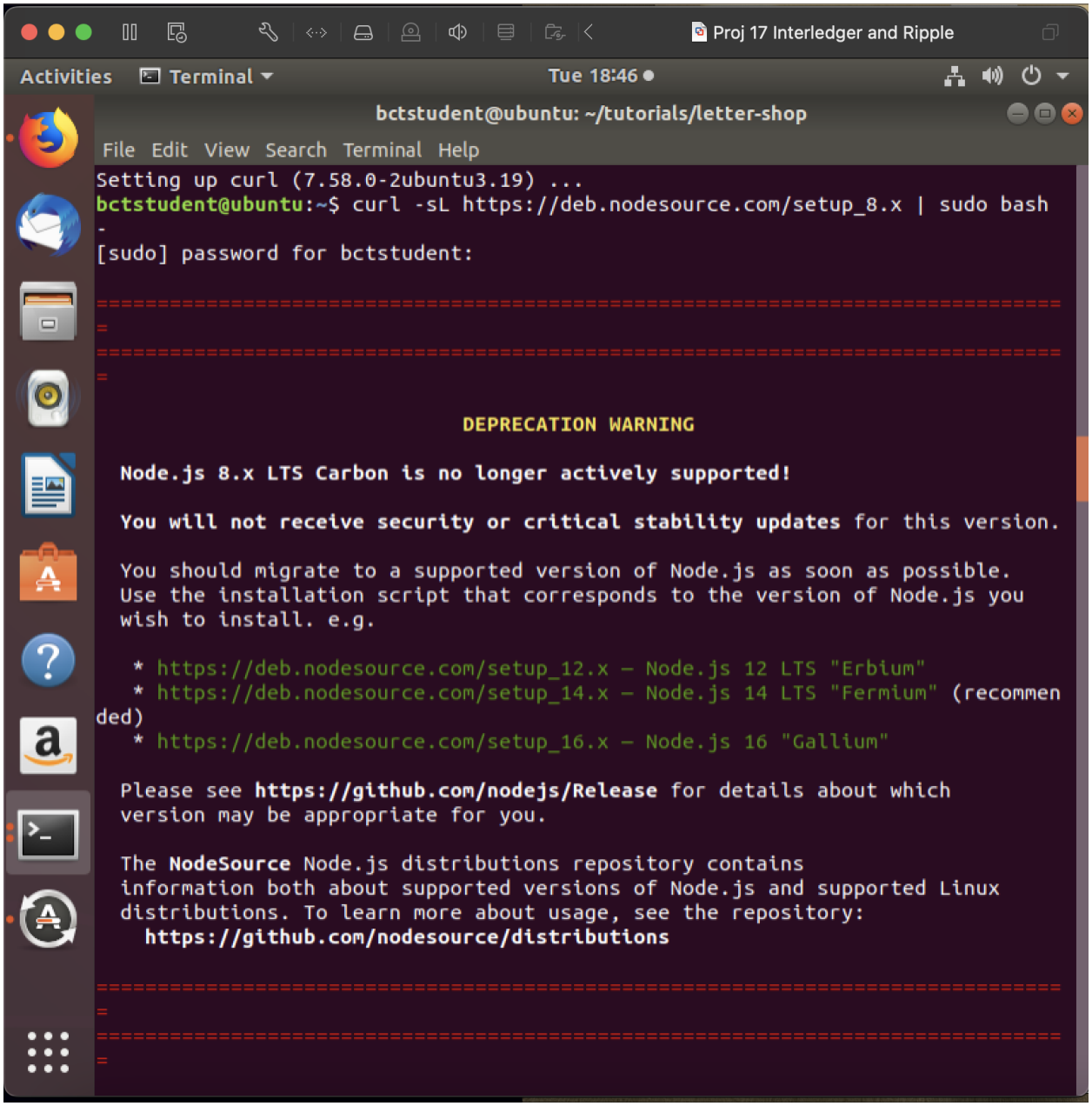
   Go to terminal window, execute the following command:

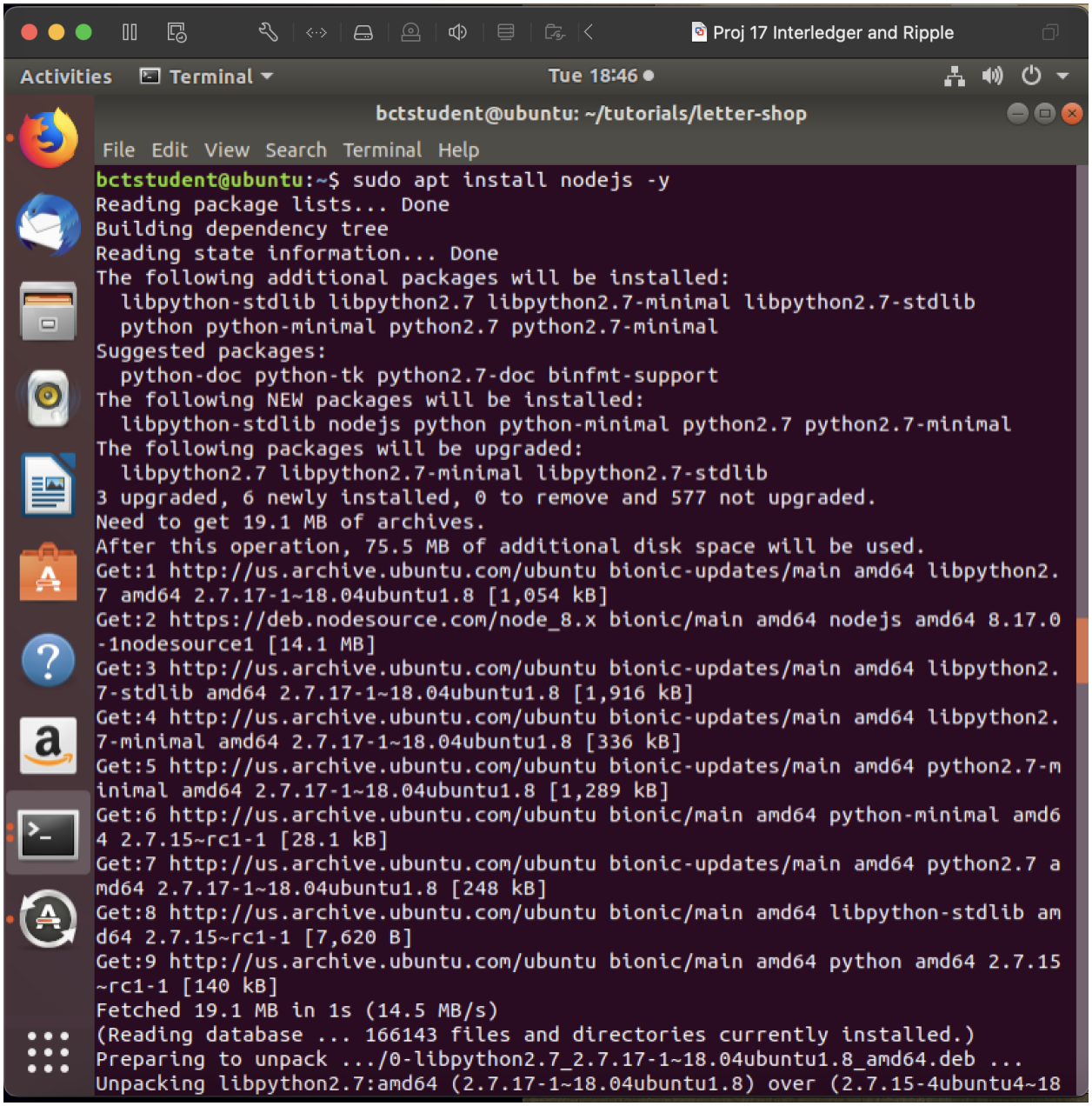
   ```
   sudo apt update
   sudo apt install curl git make build-essential -y
   curl -sL https://deb.nodesource.com/setup_8.x | sudo bash -
   sudo apt install nodejs -y
   node -v
   npm –v
   ```

   Note: If you get a lock error in /var/lib/dpkg/lock, you must delete that directory and try again with the following command:

   ```
   sudo rm /var/lib/dpkg/lock
   ```


Proj 17 Interledger and Ripple
Activities
Terminal
Tue 18:46
bctstudent@ubuntu: ~/tutorials/letter-shop
File Edit View Search Terminal Help
580 packages can be upgraded. Run 'apt list --upgradable' to see them.
upgrade: command not found
bctstudent@ubuntu:~$ sudo apt install curl git make build-essential -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
build-essential is already the newest version (12.4ubuntu1).
make is already the newest version (4.1-9.1ubuntu1).
make set to manually installed.
The following additional packages will be installed:
git-man libcurl4 liberror-perl
Suggested packages:
git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
curl git git-man libcurl4 liberror-perl
0 upgraded, 5 newly installed, 0 to remove and 580 not upgraded.
Need to get 5,136 kB of archives.
After this operation, 35.1 MB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcurl4 amd64 7.58.0-2ubuntu3.19 [220 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 curl amd64 7.58.0-2ubuntu3.19 [159 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu bionic/main amd64 liberror-perl all 0.17025-1 [22.8 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 git-man all 1:2.17.1-1ubuntu0.12 [804 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 git amd64 1:2.17.1-1ubuntu0.12 [3,930 kB]
Fetched 5,136 kB in 2s (2,689 kB/s)
Selecting previously unselected package libcurl4:amd64.
(Reading database ... 165223 files and directories currently installed.)
Preparing to unpack .../libcurl4_7.58.0-2ubuntu3.19_amd64.deb ...
Unpacking libcurl4:amd64 (7.58.0-2ubuntu3.19) ...
Selecting previously unselected package curl.
Preparing to unpack .../curl_7.58.0-2ubuntu3.19_amd64.deb ...
Unpacking curl (7.58.0-2ubuntu3.19) ...
Selecting previously unselected package liberror-perl.

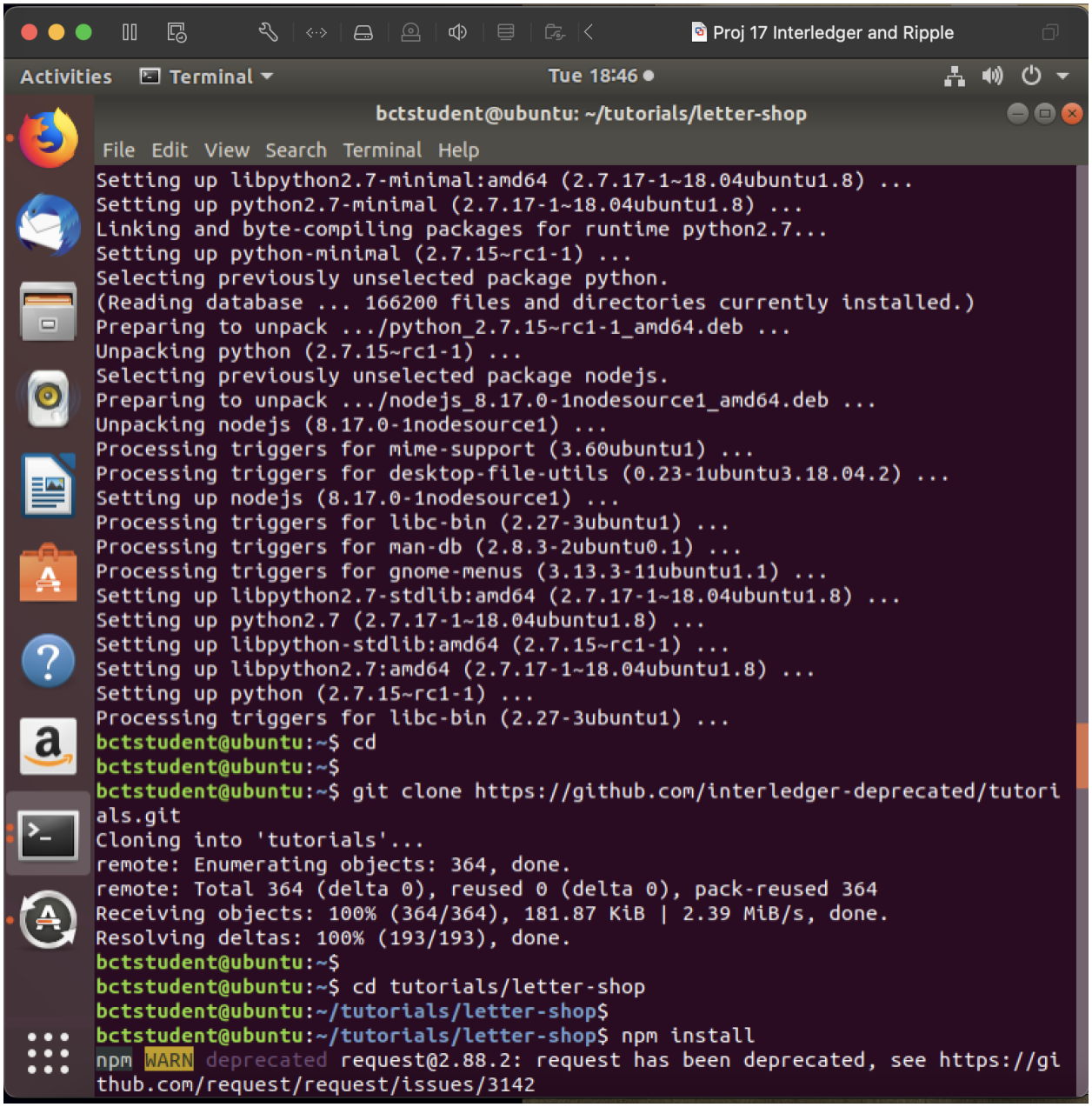

Proj 17 Interledger and Ripple
Activities
Terminal
Tue 18:46
bctstudent@ubuntu: ~/tutorials/letter-shop
File Edit View Search Terminal Help
Setting up curl (7.58.0-2ubuntu3.19) ...
bctstudent@ubuntu:~$ curl -sL https://deb.nodesource.com/setup_8.x | sudo bash
-
[sudo] password for bctstudent:
DEPRECATION WARNING
Node.js 8.x LTS Carbon is no longer actively supported!
You will not receive security or critical stability updates for this version.
You should migrate to a supported version of Node.js as soon as possible.
Use the installation script that corresponds to the version of Node.js you
wish to install. e.g.
* https://deb.nodesource.com/setup_12.x — Node.js 12 LTS "Erbium"
* https://deb.nodesource.com/setup_14.x — Node.js 14 LTS "Fermium" (recommen
ded)
* https://deb.nodesource.com/setup_16.x — Node.js 16 "Gallium"
Please see https://github.com/nodejs/Release for details about which
version may be appropriate for you.
The NodeSource Node.js distributions repository contains
information both about supported versions of Node.js and supported Linux
distributions. To learn more about usage, see the repository:
https://github.com/nodesource/distributions

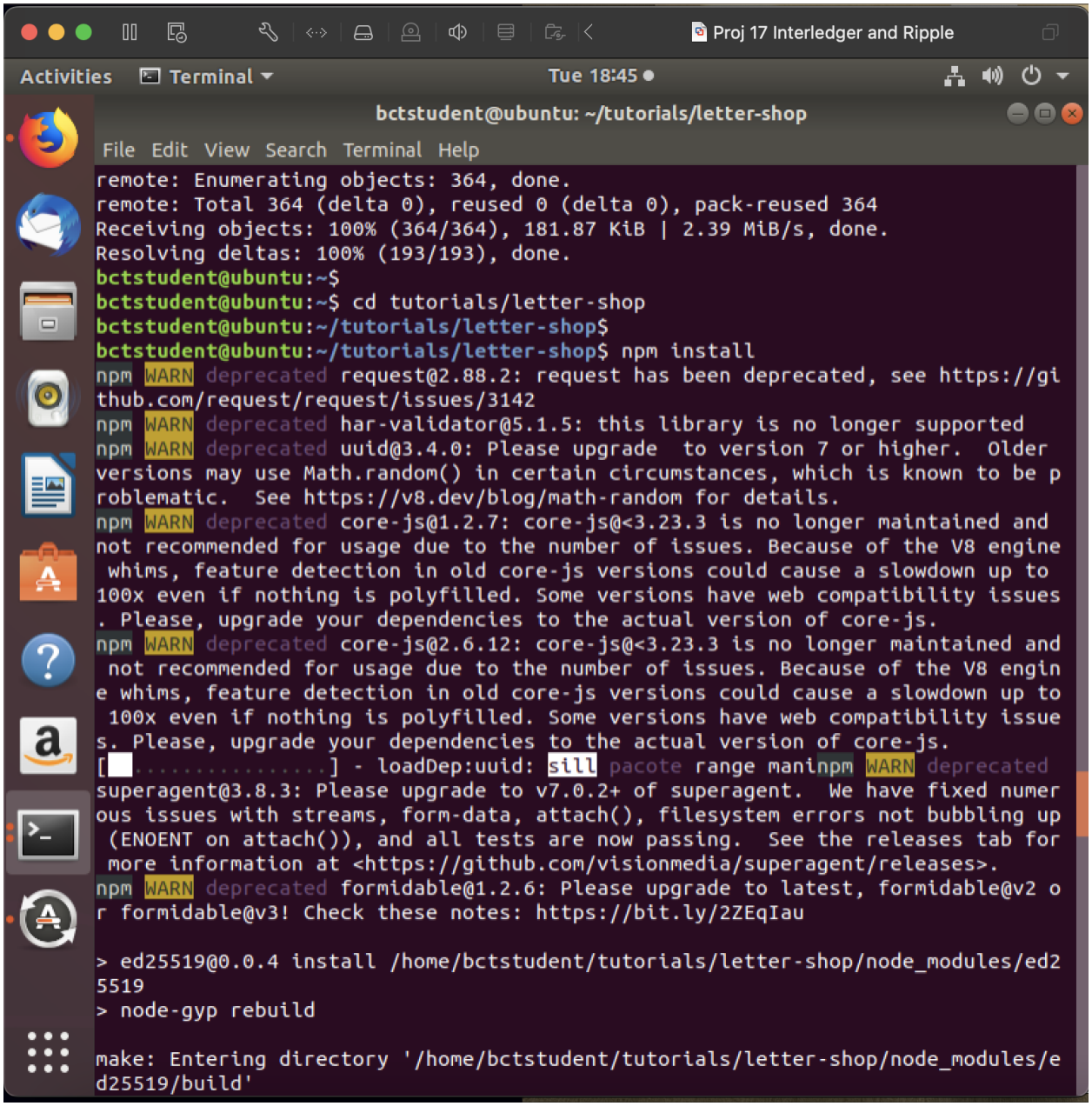

Proj 17 Interledger and Ripple
Activities
Terminal
Tue 18:46
bctstudent@ubuntu: ~/tutorials/letter-shop
File Edit View Search Terminal Help
bctstudent@ubuntu:~$ sudo apt install nodejs -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
libpython-stdlib libpython2.7 libpython2.7-minimal libpython2.7-stdlib
python python-minimal python2.7 python2.7-minimal
Suggested packages:
python-doc python-tk python2.7-doc binfmt-support
The following NEW packages will be installed:
libpython-stdlib nodejs python python-minimal python2.7 python2.7-minimal
The following packages will be upgraded:
libpython2.7 libpython2.7-minimal libpython2.7-stdlib
3 upgraded, 6 newly installed, 0 to remove and 577 not upgraded.
Need to get 19.1 MB of archives.
After this operation, 75.5 MB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libpython2.
7 amd64 2.7.17-1~18.04ubuntu1.8 [1,054 kB]
Get:2 https://deb.nodesource.com/node_8.x bionic/main amd64 nodejs amd64 8.17.0
-1nodesource1 [14.1 MB]
Get:3 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libpython2.
7-stdlib amd64 2.7.17-1~18.04ubuntu1.8 [1,916 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libpython2.
7-minimal amd64 2.7.17-1~18.04ubuntu1.8 [336 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 python2.7-m
inimal amd64 2.7.17-1~18.04ubuntu1.8 [1,289 kB]
Get:6 http://us.archive.ubuntu.com/ubuntu bionic/main amd64 python-minimal amd6
4 2.7.15~rc1-1 [28.1 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu bionic-updates/main amd64 python2.7 a
md64 2.7.17-1~18.04ubuntu1.8 [248 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu bionic/main amd64 libpython-stdlib am
d64 2.7.15~rc1-1 [7,620 B]
Get:9 http://us.archive.ubuntu.com/ubuntu bionic/main amd64 python amd64 2.7.15
~rc1-1 [140 kB]
Fetched 19.1 MB in 1s (14.5 MB/s)
(Reading database ... 166143 files and directories currently installed.)
Preparing to unpack .../0-libpython2.7_2.7.17-1~18.04ubuntu1.8_amd64.deb ...
Unpacking libpython2.7:amd64 (2.7.17-1~18.04ubuntu1.8) over (2.7.15-4ubuntu4~18

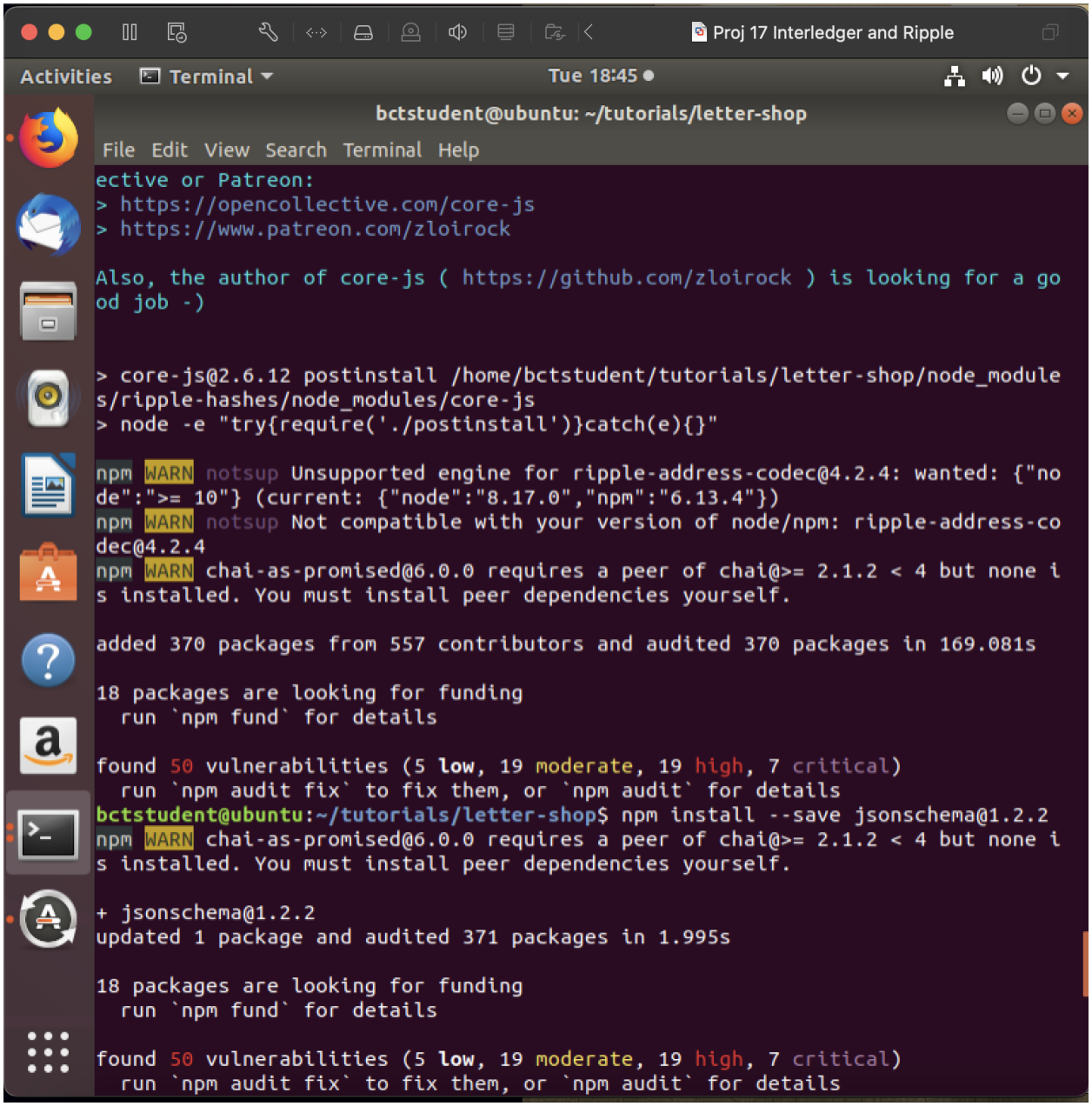
2. **Getting the code**

Go to terminal window, execute the following command:

**cd**
**git clone https://github.com/interledger-deprecated/tutorials.git**
**cd tutorials/letter-shop**
**npm install**




Proj 17 Interledger and Ripple
Activities
Terminal
Tue 18:45
bctstudent@ubuntu: ~/tutorials/letter-shop
File Edit View Search Terminal Help
remote: Enumerating objects: 364, done.
remote: Total 364 (delta 0), reused 0 (delta 0), pack-reused 364
Receiving objects: 100% (364/364), 181.87 KiB | 2.39 MiB/s, done.
Resolving deltas: 100% (193/193), done.
bctstudent@ubuntu:~$
bctstudent@ubuntu:~$ cd tutorials/letter-shop
bctstudent@ubuntu:~/tutorials/letter-shop$
bctstudent@ubuntu:~/tutorials/letter-shop$ npm install
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
npm WARN deprecated core-js@1.2.7: core-js@<3.23.3 is no longer maintained and not recommended for usage due to the number of issues. Because of the V8 engine whims, feature detection in old core-js versions could cause a slowdown up to 100x even if nothing is polyfilled. Some versions have web compatibility issues. Please, upgrade your dependencies to the actual version of core-js.
npm WARN deprecated core-js@2.6.12: core-js@<3.23.3 is no longer maintained and not recommended for usage due to the number of issues. Because of the V8 engine whims, feature detection in old core-js versions could cause a slowdown up to 100x even if nothing is polyfilled. Some versions have web compatibility issues. Please, upgrade your dependencies to the actual version of core-js.
[..................] - loadDep:uuid: sill pacote range maninpm WARN deprecated superagent@3.8.3: Please upgrade to v7.0.2+ of superagent. We have fixed numerous issues with streams, form-data, attach(), filesystem errors not bubbling up (ENOENT on attach()), and all tests are now passing. See the releases tab for more information at <https://github.com/visionmedia/superagent/releases>.
npm WARN deprecated formidable@1.2.6: Please upgrade to latest, formidable@v2 or formidable@v3! Check these notes: https://bit.ly/2ZEqIau
> ed25519@0.0.4 install /home/bctstudent/tutorials/letter-shop/node_modules/ed25519
> node-gyp rebuild
make: Entering directory '/home/bctstudent/tutorials/letter-shop/node_modules/ed25519/build'




3. **Running the Shop: First Attempt**

Go to terminal window, execute the following command:

**npm install --save jsonschema@1.2.2**
**node shop.js**

Because we need to add account numbers to the code in the file "plugins.js," you notice an error message, as shown below.

## 4. Getting Ripple Addresses

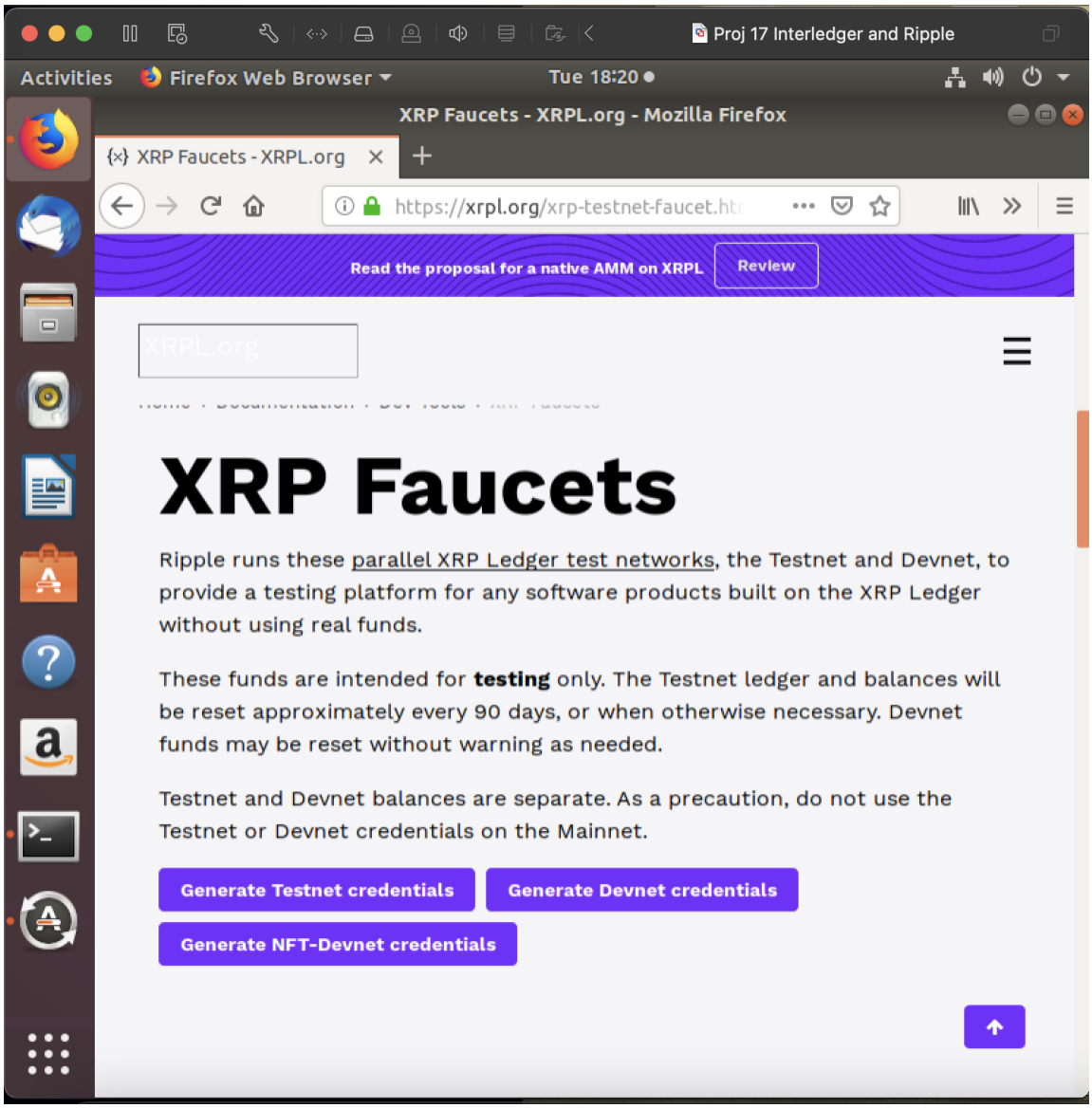
Go to web browser, and enter the link address mentioned below:

https://ripple.com/build/xrp-test-net/ or

https://xrpl.org/xrp-test-net-faucet.html

Click the generate test credential button. Copy address and secret value to text file for later use. These will be your **SHOPPER** credentials.

Click the generate test credential button. Copy address and secret value to text file for later use. These will be your **CUSTOMER** credentials.



5. **Adding Credentials to plugins.js**

Go to terminal window, execute the following command:

**cp plugins.js plugins.js.bak**
**nano plugins.js**

Add two slashes to the start of each line to comment the lines out.
Remove the / from the beginning of each line.
Copy and paste the ADDRESS, SECRET, and values into the lines for both the SHOP and the CUSTOMER. Take note that the address is for the Account.

6. **Running the Shop: Second Attempt**

Go to terminal window, execute the following command:
**node shop.js**

## 7. Connecting to the Ripple Account

Go to terminal window, execute the following command:

**cp shop.js shop.js.bak1**
**nano shop.js**

The code ends with a comment saying "Do something...", as shown below.

Paste this code at the end of the file to connect it with ripple account so that shop can receive fund:

```
console.log(` 1. Connecting to an account to accept payments...`)

plugin.connect().then(function () {
  // Get ledger and account information from the plugin
  const ledgerInfo = plugin.getInfo()
  const account = plugin.getAccount()

  console.log(`    - Connected to ledger: ${ledgerInfo.prefix}`)
  console.log(`    -- Account: ${account}`)
  console.log(`    -- Currency: ${ledgerInfo.currencyCode}`)
  console.log(`    -- CurrencyScale: ${ledgerInfo.currencyScale}`)

  // Convert our cost (10) into the right format given the ledger scale
  const normalizedCost = cost / Math.pow(10,
parseInt(ledgerInfo.currencyScale))

console.log(` 2. Starting web server to accept requests...`)
 console.log(`    - Charging ${normalizedCost} ${ledgerInfo.currencyCode}`)
```

```
// Handle incoming web requests...

// Handle incoming transfers...

})
```

8. **Running the Shop: Third Attempt**

Go to terminal window, execute the following command:

**node shop.js**

9. **Handling Web Requests**

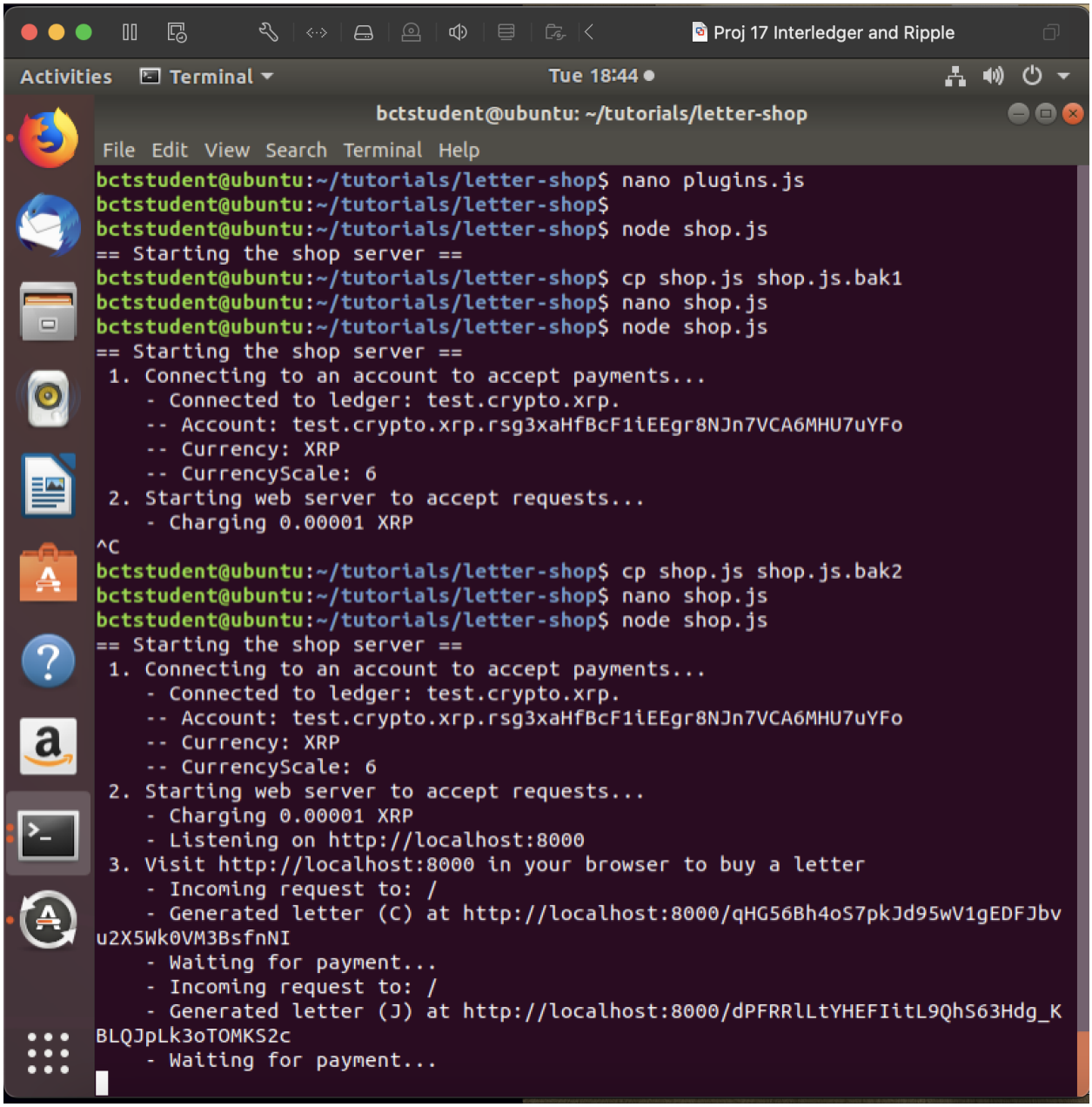
Go to terminal window, execute the following command:

**cp shop.js shop.js.bak2**

**nano shop.js**

Paste this code after the "// Handle incoming web requests" line and before the "// Handle incoming transfers…":

```
// Handle incoming web requests

http.createServer(function (req, res) {
  // Browsers are irritating and often probe for a favicon, just ignore
        if (req.url.startsWith(`/favicon.ico`)) {
          res.statusCode = 404
            res.end()
             return
            }

              console.log(`    - Incoming request to: ${req.url}`)
    const requestUrl = url.parse(req.url)

            if (requestUrl.path === `/`) {
```




**// Request for a letter with no attached fulfillment**

**// Respond with a 402 HTTP Status Code (Payment Required)**
**res.statusCode = 402**

**// Generate a preimage and its SHA256 hash,**
**// which we'll use as the fulfillment and condition, respectively, of the**
**// conditional transfer.**
**const fulfillment = crypto.randomBytes(32)**
**const condition = sha256(fulfillment)**

```
      // Get the letter that we are selling
      const letter = ('ABCDEFGHIJKLMNOPQRSTUVWXYZ')
      .split('')[(Math.floor(Math.random() * 26))]

        console.log(`    - Generated letter (${letter}) ` +
        `at http://localhost:8000${req.url}${base64url(fulfillment)}`)

      // Store the fulfillment (indexed by condition) to use when we get paid
        fulfillments[base64url(condition)] = fulfillment

    // Store the letter (indexed by the fulfillment) to use when the customer
        // requests it
        letters[base64url(fulfillment)] = letter

        console.log(`    - Waiting for payment...`)

        res.setHeader(`Pay`, `${cost} ${account} ${base64url(condition)}`)

        res.end(`Please send an Interledger payment of` +
          ` ${normalizedCost} ${ledgerInfo.currencyCode} to ${account}`
+
          ` using the condition ${base64url(condition)}\n` +
        `> node ./pay.js ${account} ${cost} ${base64url(condition)}`)
      } else {
        // Request for a letter with the fulfillment in the path

        // Get fulfillment from the path
        const fulfillmentBase64 = requestUrl.path.substring(1)

        // Lookup the letter we stored previously for this fulfillment
        const letter = letters[fulfillmentBase64]

        if (!letter) {
      // We have no record of a letter that was issued for this fulfillment

          // Respond with a 404 HTTP Status Code (Not Found)
          res.statusCode = 404

          console.log('    - No letter found for fulfillment: ' +
                                        fulfillmentBase64)

          res.end(`Unrecognized fulfillment.`)
        } else {
          // Provide the customer with their letter
```

```
      res.end(`Your letter: ${letter}`)

      console.log(` 5. Providing paid letter to customer ` +
                  `for fulfillment ${fulfillmentBase64}`)
    }
  }
}).listen(8000, function () {
  console.log(`    - Listening on http://localhost:8000`)
   console.log(` 3. Visit http://localhost:8000 in your browser ` +
                                          `to buy a letter`)
})
```

**10. Running the Shop: Fourth Attempt**

Go to terminal window, execute the following command:

**node shop.js**

The shop connects to a ledger, and says it's starting a web server,

**11. Paying for a Letter: First Attempt**

Go to web browser, and enter the link address mentioned below:

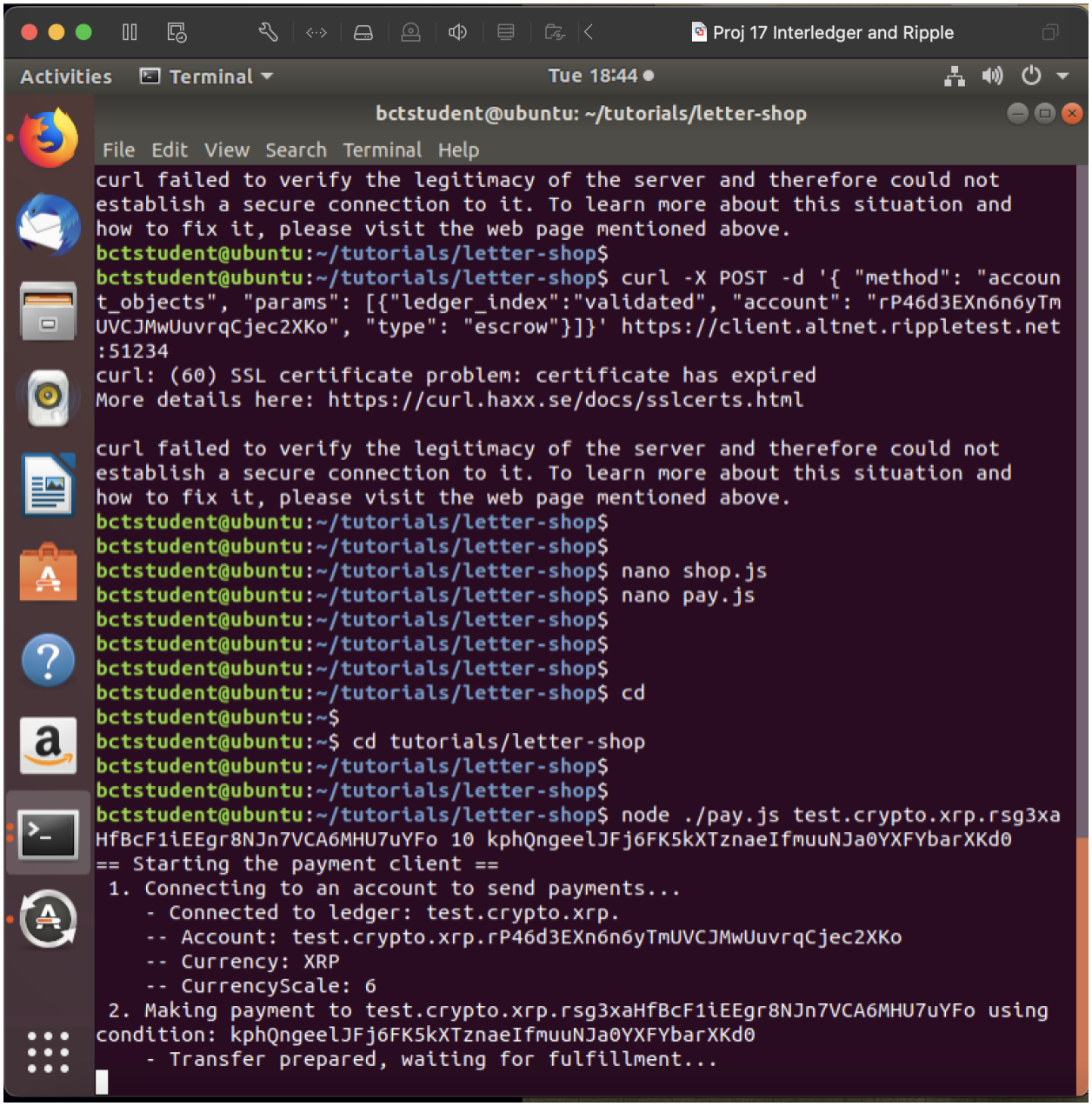
http://localhost:8000

Go to terminal window, execute the following command:

**cd**

**cd tutorials/letter-shop**

After that, copy the command from the browser window and execute it in the Terminal window.




**12. Connecting to an Account**

On the terminal, execute :

**nano pay.js**

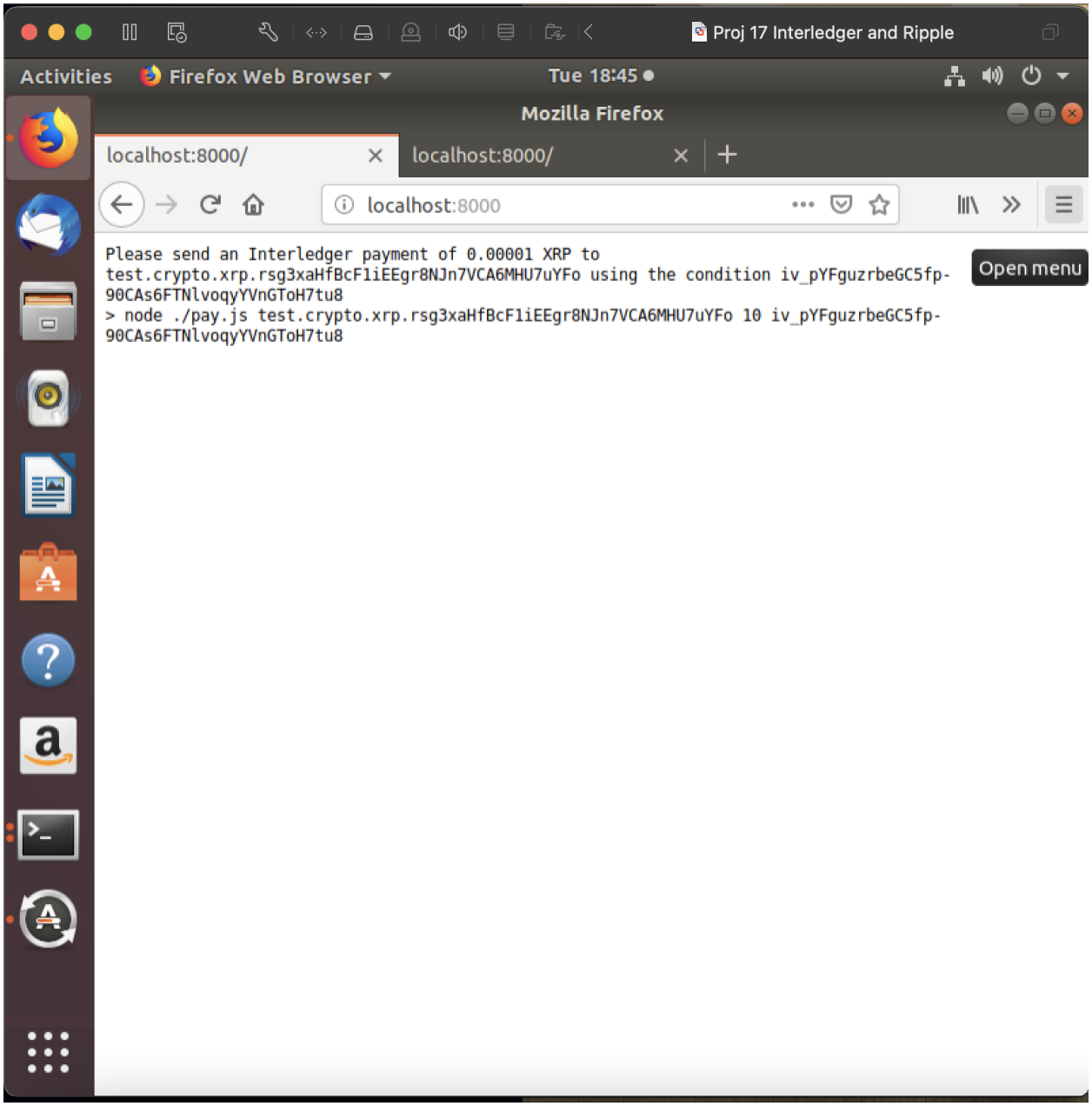
At the bottom, paste in this code:

```
console.log(` 1. Connecting to an account to send payments...`)

plugin.connect().then(function () {
  const ledgerInfo = plugin.getInfo()
  const account = plugin.getAccount()
  console.log(`    - Connected to ledger: ${ledgerInfo.prefix}`)
  console.log(`    -- Account: ${account}`)
  console.log(`    -- Currency: ${ledgerInfo.currencyCode}`)
  console.log(`    -- CurrencyScale: ${ledgerInfo.currencyScale}`)

  // Make payment...

  // Listen for fulfillments...

})
```

## 13. Paying for a Letter: Second Attempt

Go to the terminal window, execute the same command you copied from the browser.

The client connects to an account, but then just hangs


Proj 17 Interledger and Ripple
Activities
Firefox Web Browser
Tue 18:45
Mozilla Firefox
localhost:8000/
localhost:8000/
localhost:8000
Please send an Interledger payment of 0.00001 XRP to test.crypto.xrp.rsg3xaHfBcF1iEEgr8NJn7VCA6MHU7uYFo using the condition iv_pYFguzrbeGC5fp-90CAs6FTNlvoqyYVnGToH7tu8
> node ./pay.js test.crypto.xrp.rsg3xaHfBcF1iEEgr8NJn7VCA6MHU7uYFo 10 iv_pYFguzrbeGC5fp-90CAs6FTNlvoqyYVnGToH7tu8
Open menu

## 14. Connecting to an Account

There is more code missing from the ***pay.js*** file. We will add some of them.

Execute : **nano pay.js**

Near the bottom, after the "// Make payment..." line, paste in this code:

```
console.log(` 2. Making payment to ${destinationAddress} ` +
                        `using condition: ${condition}`)

 // Send the transfer
 plugin.sendTransfer({
  to: destinationAddress,
  amount: destinationAmount,
  executionCondition: condition,
  id: uuid(),
  from: plugin.getAccount(),
  ledger: plugin.getInfo().prefix,
  ilp: base64url(IlpPacket.serializeIlpPayment({
   amount: destinationAmount,
   account: destinationAddress
  })),
  expiresAt: new Date(new Date().getTime() + 1000000).toISOString()
 }).then(function () {
  console.log('    - Transfer prepared, waiting for fulfillment...')
 }, function (err) {
  console.error(err.message)
 })
```

**Press Ctrl+X, Y, Enter to save the file.**

## 15. Paying for a Letter: Third Attempt

Go to the terminal window, execute the same command you copied from the browser.

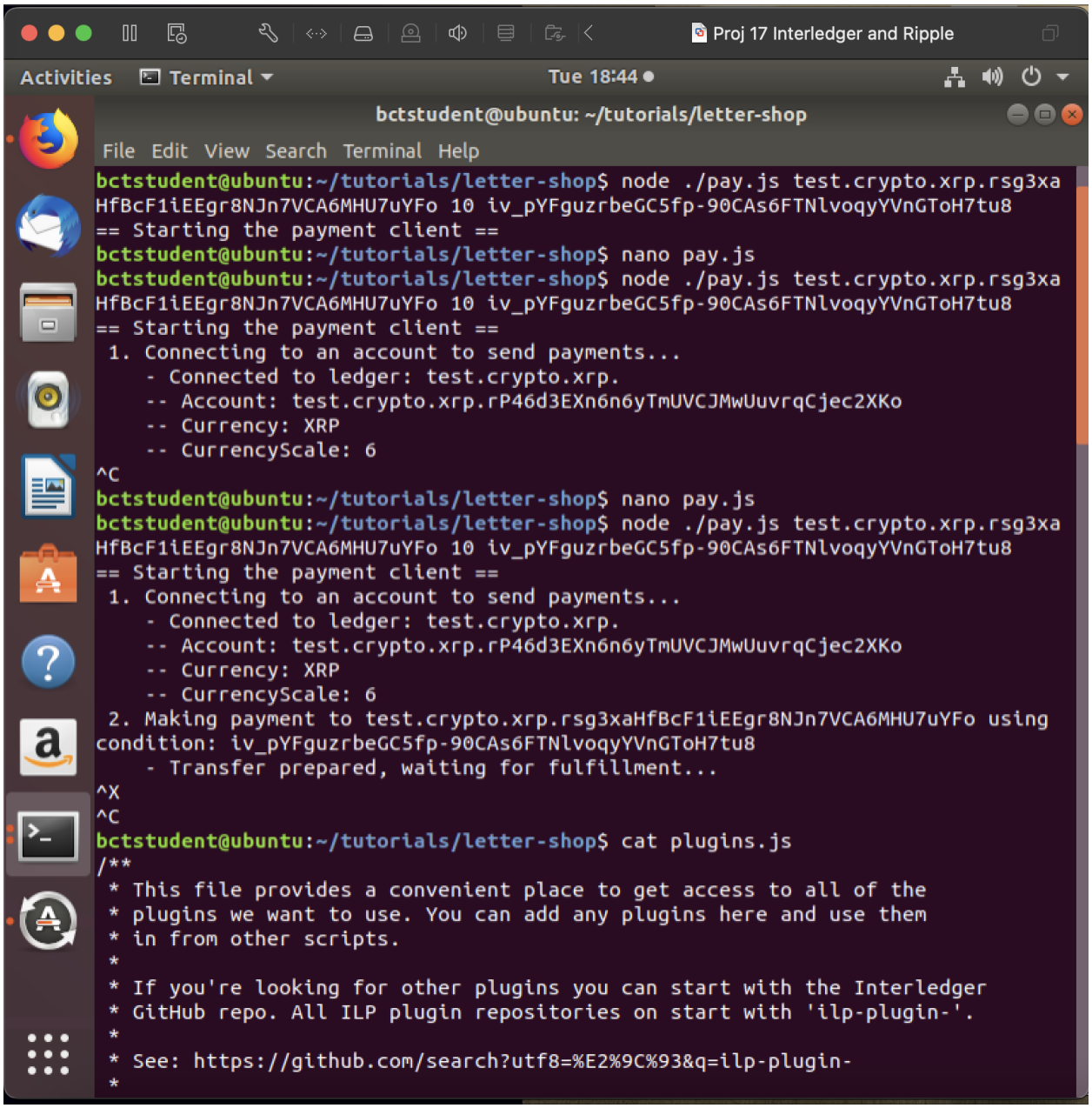
The client makes the payment, but stops with a "waiting for fulfillment" message

## 16. Finding Your CUSTOMER Account Number

Go to the terminal window (second terminal tutorials/letters-shop ),execute :

**cat plugins.js**

The CUSTOMER account number appears at the end of the output.




### 17. Finding Your Transfer on the Ledger

Go to new terminal,replacing **YOUR-SENDING-ADDRESS** with your own CUSTOMER address:

**curl -X POST -d '{ "method": "account_objects", "params": [{"ledger_index":"validated", "account": "YOUR-SENDING-ADDRESS", "type": "escrow"}]}' https://client.altnet.rippletest.net:51234**

## 18. Accepting the Payment

There is more code missing from the shop.js file. We will add some of them.

```
nano shop.js
```

Near the bottom, after the "//Handle incoming transfers..." line, paste in this code:

```
// Handle incoming payments

plugin.on('incoming_prepare', function (transfer) {
  if (parseInt(transfer.amount) < 10) {
    // Transfer amount is incorrect
    console.log(`    - Payment received for the wrong amount ` +
                          `(${transfer.amount})... Rejected`)

    const normalizedAmount = transfer.amount /
                    Math.pow(10, parseInt(ledgerInfo.currencyScale))

    plugin.rejectIncomingTransfer(transfer.id, {
      code: 'F04',
      name: 'Insufficient Destination Amount',
      message: `Please send at least 10 ${ledgerInfo.currencyCode},` +
            `you sent ${normalizedAmount}`,
      triggered_by: plugin.getAccount(),
      triggered_at: new Date().toISOString(),
      forwarded_by: [],
      additional_info: {}
    })
  } else {
    // Lookup fulfillment from condition attached to incoming transfer
    const fulfillment = fulfillments[transfer.executionCondition]

    if (!fulfillment) {
      // We don't have a fulfillment for this condition
      console.log(`    - Payment received with an unknown condition: ` +
                                `${transfer.executionCondition}`)

      plugin.rejectIncomingTransfer(transfer.id, {
        code: 'F05',
        name: 'Wrong Condition',
        message: `Unable to fulfill the condition:  ` +
                                `${transfer.executionCondition}`,
```

```
    triggered_by: plugin.getAccount(),
    triggered_at: new Date().toISOString(),
    forwarded_by: [],
    additional_info: {}
  })
 }

 console.log(` 4. Accepted payment with condition ` +
                          `${transfer.executionCondition}.`)
 console.log(`    - Fulfilling transfer on the ledger ` +
                   `using fulfillment: ${base64url(fulfillment)}`)

 // The ledger will check if the fulfillment is correct and
 // if it was submitted before the transfer's rollback timeout
 plugin.fulfillCondition(transfer.id, base64url(fulfillment))
  .catch(function () {
   console.log(`    - Error fulfilling the transfer`)
  })
 console.log(`    - Payment complete`)
}
})
```

## 19. Adding the Event Listener to the Client

There is more code missing from the pay.js file. We will add some of them.

**nano pay.js**

Near the bottom, after the "// Listen for fulfillments..." line, paste in this code:

```
// Handle fulfillments
 plugin.on('outgoing_fulfill', function (transferId, fulfillmentBase64) {
  console.log('    - Transfer executed. Got fulfillment: ' +
                                    fulfillmentBase64)
  console.log(` 3. Collect your letter at ` +
                `http://localhost:8000/${fulfillmentBase64}`)
  plugin.disconnect()
  process.exit()
 })
```

## 20. Buying a Letter

Go to the web browser, execute :

http://localhost:8000

**Note the command got as an output from above step.**

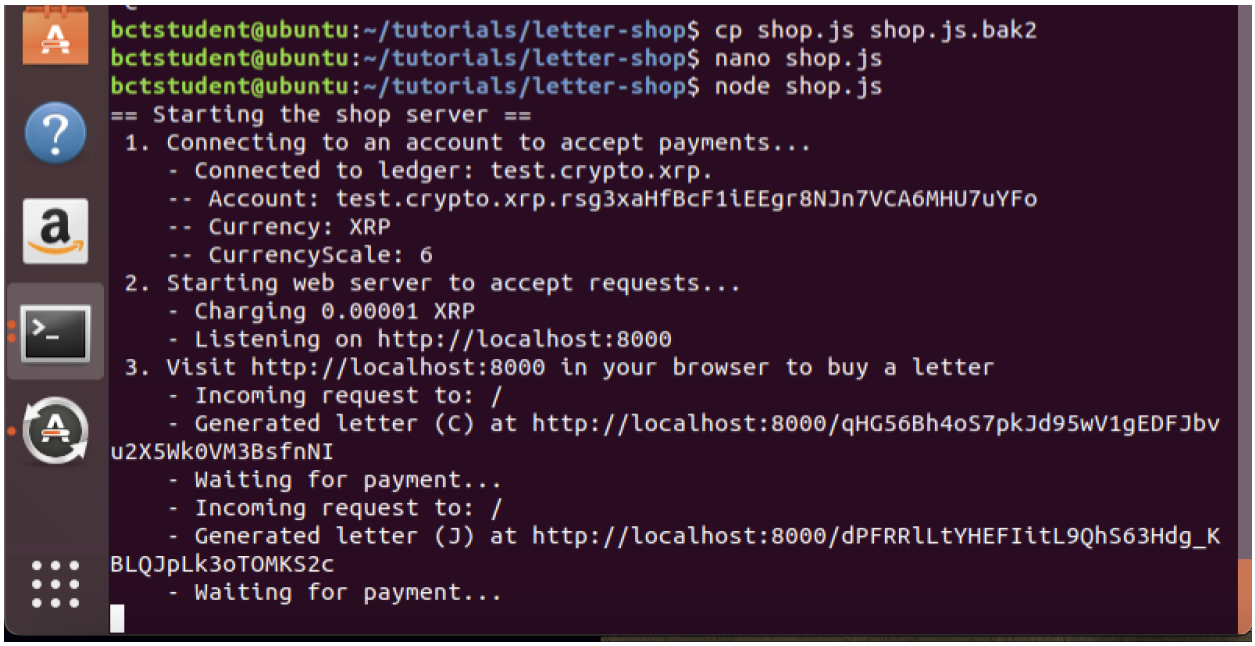
Go to the terminal window (second terminal tutorials/letters-shop ),execute :

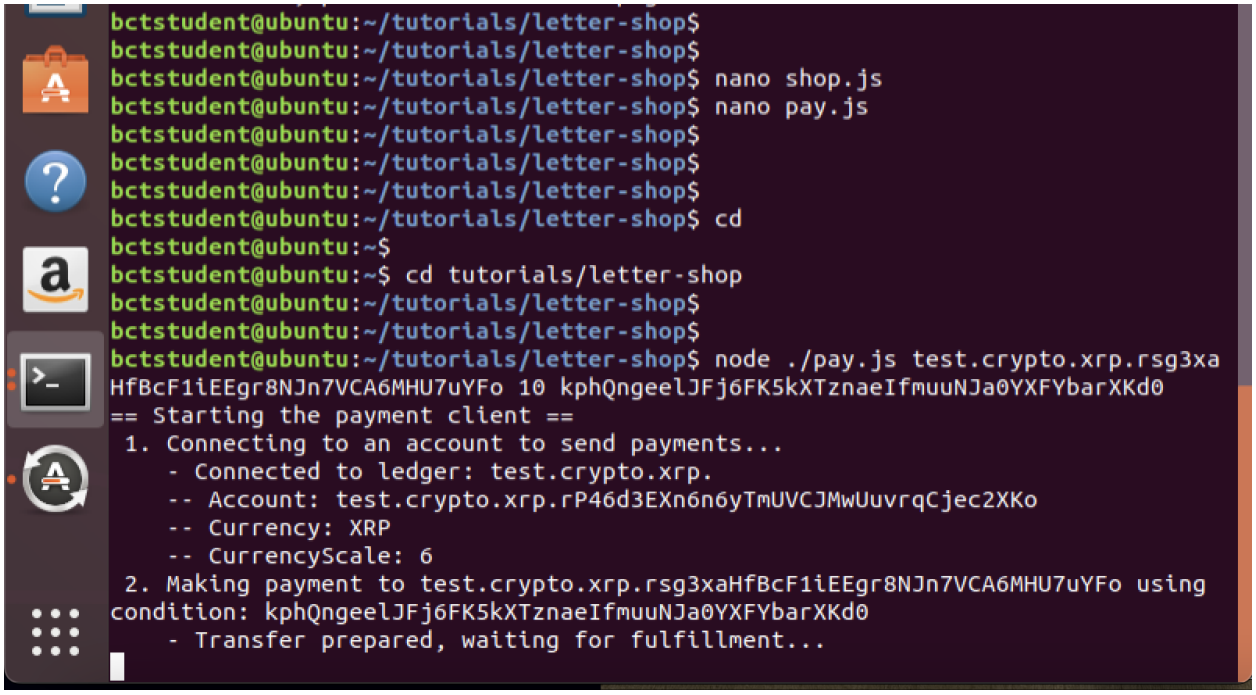
**cd**

**cd tutorials/letter-shop**

Now copy the command from the browser window and execute it in the Terminal window.

**node shop.js**






**CONCLUSION :** To summarize, we can generate money by utilizing the shop file and the xrp (ripple cryptocurrency) file, but we are using the test net here. When we initially ran shop.js, we discovered that we needed to provide shopper and customer account credentials, keys, and addresses. So we can get those details from the ripple.com website, and once we add both addresses, we run shop.js again, and this time the server successfully starts, so we add receive fund code in shop.js and run again, and now the shopper can receive funds from global requests; however, we still cannot receive web requests, so we add handling web request code in shop.js and run shop.js again.

We can now send money to that shoper's address by adding pay code to pay.js and running it. Then we add the transfer code to pay.js and run it again. So we execute the pay command and wait to see if we are paid. Because it did not function in this case, we used the curl API command to examine the jason output and saw that the status was successful. Then we added the fulfillment code to pay.js and ran it again, but we still didn't finish because Rippler crashed by 92 percent or for some other reason that we didn't discover.