

Laptop Specification	
Manufacturer	Acer
Unit	Aspire 3 - A315-41-R287
OS	Ubuntu 18.04.1 LTS
OS Type	64 bit
CPU	AMD Ryzen 3 2200U
Ram	8 Gb
Kernel	Linux 4.15.0-43-generic (x86_64)

## **Installation:**

First we need to install all of the dependencies for hyperledger.

Open terminal then paste this:

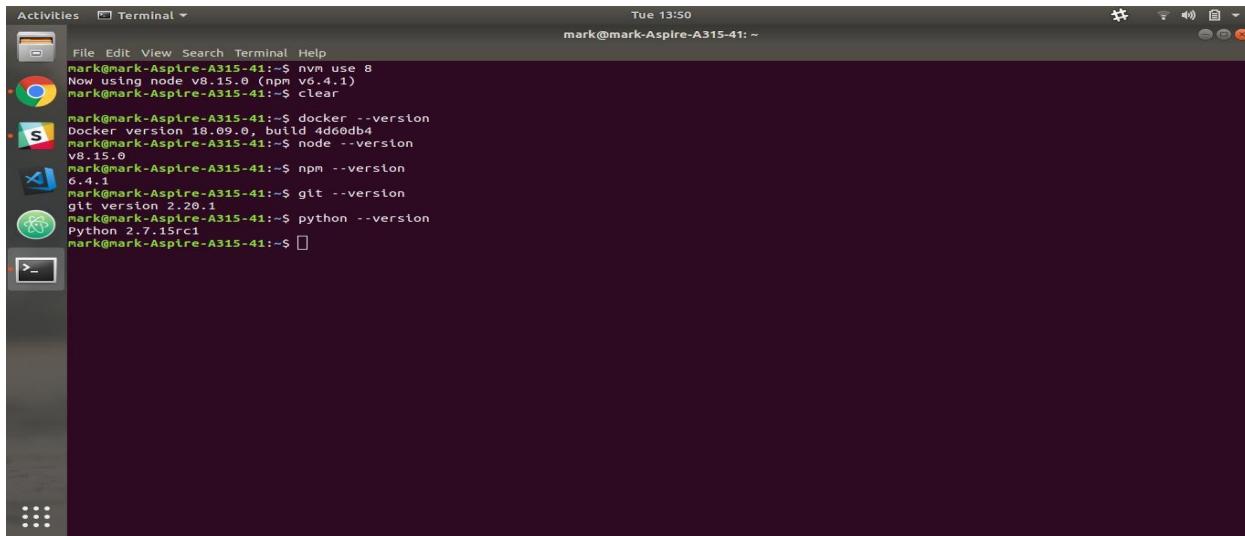
```
curl -o https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh  
chmod u+x prereqs-ubuntu.sh  
../prereqs-ubuntu.sh
```

wait for it to finish installation. Then type this:

```
docker -version  
npm -version  
git -version  
python -version  
node -version
```

so you can check the version of your development environment.

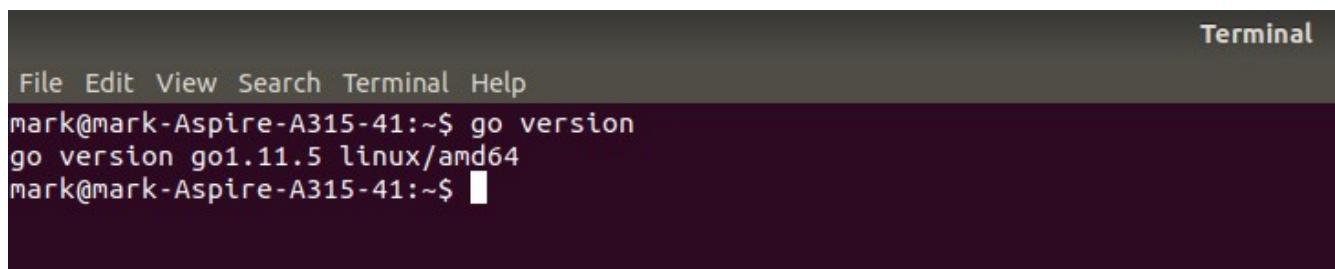
Mark John R. Frias



after that, install Go language by pasting this to your terminal:

```
wget https://storage.googleapis.com/golang/go1.9.2.linux-amd64.tar.gz && \
sudo tar -C /usr/local -xzf go1.9.2.linux-amd64.tar.gz && \
rm go1.9.2.linux-amd64.tar.gz && \
echo 'export PATH=$PATH:/usr/local/go/bin' | sudo tee -a /etc/profile && \
echo 'export GOPATH=$HOME/go' | tee -a $HOME/.bashrc && \
echo 'export PATH=$PATH:$GOROOT/bin:$GOPATH/bin' | tee -a $HOME/.bashrc && \
mkdir -p $HOME/go/{src,pkg,bin}
```

wait for it to finish then type go -version.

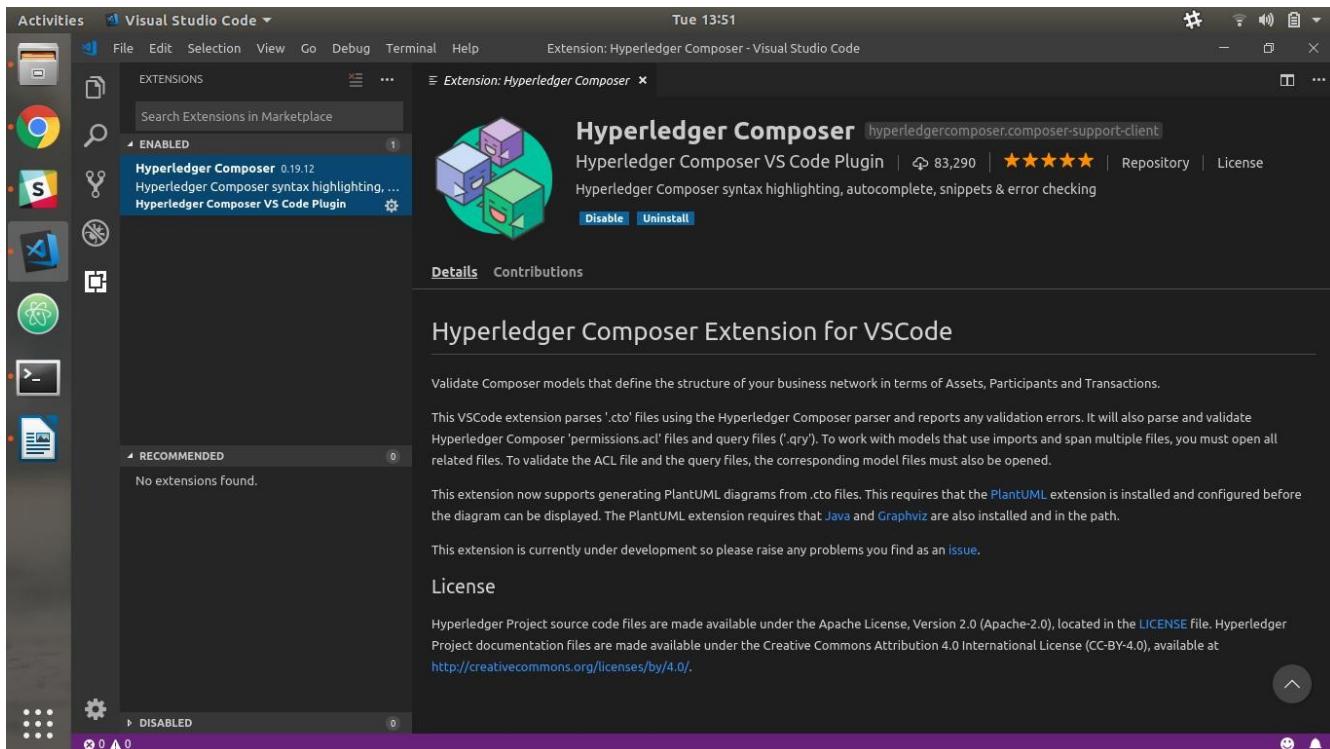


Then install vscode. Just paste this to your terminal:

```
sudo snap install vscode --classic
```

Mark John R. Frias

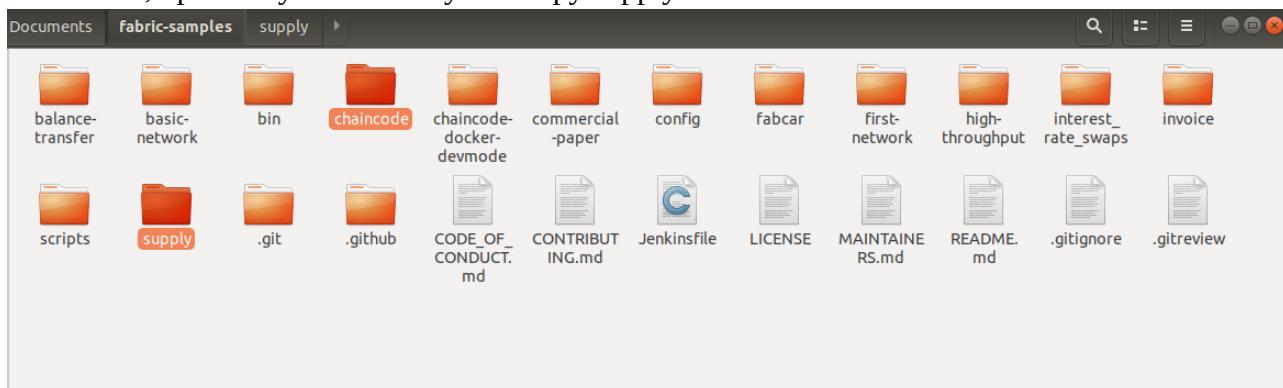
after installing open vscode then go to view => extension => on search bar type hyperledger composer then install it. You should see this:



## Hyperledger Activity:

### Mandatory:

1. First, if you have existing blockchain-training-labs folder in your directory. You should rename it so you wont have any problems when you clone another blockchain-training-labs in git hub.
2. Type in your terminal: git clone <https://github.com/khrandm/blockchain-training-labs>
3. Then, open the your directory and copy supply and chaincode folder.



Mark John R. Frias

4. Paste it to your fabric samples folder.
5. Open supply folder. Then, right click open terminal and type `npm install`.



A screenshot of a terminal window showing the output of an `npm install` command. The terminal shows the compilation of CXX files and the installation of various node modules like grpc and grpc-client. It ends with a warning about using node-pre-gyp for https download.

```
Terminal
File Edit View Search Terminal Help
CXX(target) Release/obj.target/pkcs11/src/pkcs11/param_rsa.o
CXX(target) Release/obj.target/pkcs11/src/pkcs11/param_ecdh.o
CXX(target) Release/obj.target/pkcs11/src/pkcs11/pkcs11.o
CXX(target) Release/obj.target/pkcs11/src/async.o
CXX(target) Release/obj.target/pkcs11/src/node.o
SOLINK_MODULE(target) Release/obj.target/pkcs11.node
COPY Release/pkcs11.node
make: Leaving directory '/home/mark/Documents/fabric-samples/supply/node_modules/pkcs11js/build'

> grpc@1.10.1 install /home/mark/Documents/fabric-samples/supply/node_modules/grpc-client/node_modules/grpc
> node-pre-gyp install --fallback-to-build --library=static_library

[grpc] Success: "/home/mark/Documents/fabric-samples/supply/node_modules/grpc-client/node_modules/grpc/src/node/extension_binary/node-v57-linux-x64-glibc/grpc_node.node" is installed via remote

> grpc@1.18.0 install /home/mark/Documents/fabric-samples/supply/node_modules/grpc
> node-pre-gyp install --fallback-to-build --library=static_library

node-pre-gyp WARN Using request for node-pre-gyp https download
```

Just `ctrl+c` if you get stuck here.

6. Paste this to your terminal: `./startFabric.sh`



A screenshot of a terminal window showing the execution of `./startFabric.sh`. The script creates a CLI, installs chaincode, and runs a chaincode invoke or query. It ends with instructions to run `npm install`, `node enrollAdmin.js`, and `node registerUser`.

```
Terminal
File Edit View Search Terminal Help
Creating cli ... done
2019-02-13 01:25:35.665 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escsc
2019-02-13 01:25:35.665 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
2019-02-13 01:25:39.734 UTC [chaincodeCmd] install -> INFO 003 Installed remote response:<status:200 payload:"OK" >
2019-02-13 01:25:40.159 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escsc
2019-02-13 01:25:40.160 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
2019-02-13 01:25:54.133 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200
Total setup execution time : 66 secs ...

Start by installing required packages run 'npm install'
Then run 'node enrollAdmin.js', then 'node registerUser'

The 'node invoke.js' will fail until it has been updated with valid arguments
The 'node query.js' may be run at anytime once the user has been registered
mark@mark-Aspire-A315-41:~/Documents/fabric-samples/supply$
```

## Mark John R. Frias

### 7. then, node enrollUser.js

```
Terminal
File Edit View Search Terminal Help

Start by installing required packages run 'npm install'
Then run 'node enrollAdmin.js', then 'node registerUser'

The 'node invoke.js' will fail until it has been updated with valid arguments
The 'node query.js' may be run at anytime once the user has been registered

mark@mark-Aspire-A315-41:~/Documents/fabric-samples/supply$ node enrollAdmin.js
  Store path:/home/mark/Documents/fabric-samples/supply/hfc-key-store
Successfully enrolled admin user "admin"
Assigned the admin user to the fabric client ::{"name":"admin","mspid":"Org1MSP"
,"roles":null,"affiliation":"","enrollmentSecret":"","enrollment":{"signingIdentity":"58849635ef38d6cb56da85e1207f4b04c0fc9024e8c2a3adf36fafdb21f0c42","identity":{"certificate":"-----BEGIN CERTIFICATE-----\nMIICAjCCAAgAwIBAgIUGVsPfRG00sM4PK1adH1YkmpYnEswCgYIKoZizj0Eawlw\nnczELMAKGa1UEBhMCVVMxEzARBgTCKnhG1mb3juwExfjAUbgNVBACTDVNhnb18cmFuY2LzY28xGTAXBgNVBAoTEG9yZzEuZxhbxBsZ55jb20xHdAabgNV\nBAMTlE2Nhlmp9yZzEuZxhbxBsZ55jb20wHcNMtkwMjEzMDEyND4whcNMj4wMjEzMDEy\nnOTAwjhAHQ8wdQYDvQOLEwZjbGlbnqxDjAMBgNVBAMTBWFKbwLMFkwEwHkoZi\\nzj0CAQYIKoZizj0DAQcDQgAESEJWmc0M05dYiaMscp2dCorbdn3ew7CTtKgx1b\\nt8X1j01Y0t0FNUN5cdnDM1JKGPXG8MCqqNBkxu8xLhg06NsMGowDgYDVR0PAOH/nBAQDAgeAMwGA1UdEwEB/wQCMIAwHgQYDVR00BBYEfkldGc0+R4JYpvnabs7bkn91\\n8PysMcsgA1Udiw0kMCKAIE15gg3ndtruuLoM2nAYUDFBNNarRst3du5alc2XkL8\\nAoGCCqGSM49BAMCA0gAEMUCIQRg5492egf59PLPdA7fDVRffs2Sfoxc09D52\\nHt4FwIgSTmp1VQfJDTfVypQViq1GItqZ45vSSS4FKN6t3CU=\\n----END CERTIFICATE----\\n"}}

mark@mark-Aspire-A315-41:~/Documents/fabric-samples/supply$
```



### 8. followed by node registerUser.js

```
Terminal
File Edit View Search Terminal Help

Store path:/home/mark/Documents/fabric-samples/supply/hfc-key-store
Successfully enrolled admin user "admin"
Assigned the admin user to the fabric client ::{"name":"admin","mspid":"Org1MSP"
,"roles":null,"affiliation":"","enrollmentSecret":"","enrollment":{"signingIdentity":"58849635ef38d6cb56da85e1207f4b04c0fc9024e8c2a3adf36fafdb21f0c42","identity":{"certificate":"-----BEGIN CERTIFICATE-----\nMIICAjCCAAgAwIBAgIUGVsPfRG00sM4PK1adH1YkmpYnEswCgYIKoZizj0Eawlw\nnczELMAKGa1UEBhMCVVMxEzARBgTCKnhG1mb3juwExfjAUbgNVBACTDVNhnb18cmFuY2LzY28xGTAXBgNVBAoTEG9yZzEuZxhbxBsZ55jb20xHdAabgNV\nBAMTlE2Nhlmp9yZzEuZxhbxBsZ55jb20wHcNMtkwMjEzMDEyND4whcNMj4wMjEzMDEy\nnOTAwjhAHQ8wdQYDvQOLEwZjbGlbnqxDjAMBgNVBAMTBWFKbwLMFkwEwHkoZi\\nzj0CAQYIKoZizj0DAQcDQgAESEJWmc0M05dYiaMscp2dCorbdn3ew7CTtKgx1b\\nt8X1j01Y0t0FNUN5cdnDM1JKGPXG8MCqqNBkxu8xLhg06NsMGowDgYDVR0PAOH/nBAQDAgeAMwGA1UdEwEB/wQCMIAwHgQYDVR00BBYEfkldGc0+R4JYpvnabs7bkn91\\n8PysMcsgA1Udiw0kMCKAIE15gg3ndtruuLoM2nAYUDFBNNarRst3du5alc2XkL8\\nAoGCCqGSM49BAMCA0gAEMUCIQRg5492egf59PLPdA7fDVRffs2Sfoxc09D52\\nHt4FwIgSTmp1VQfJDTfVypQViq1GItqZ45vSSS4FKN6t3CU=\\n----END CERTIFICATE----\\n"}}

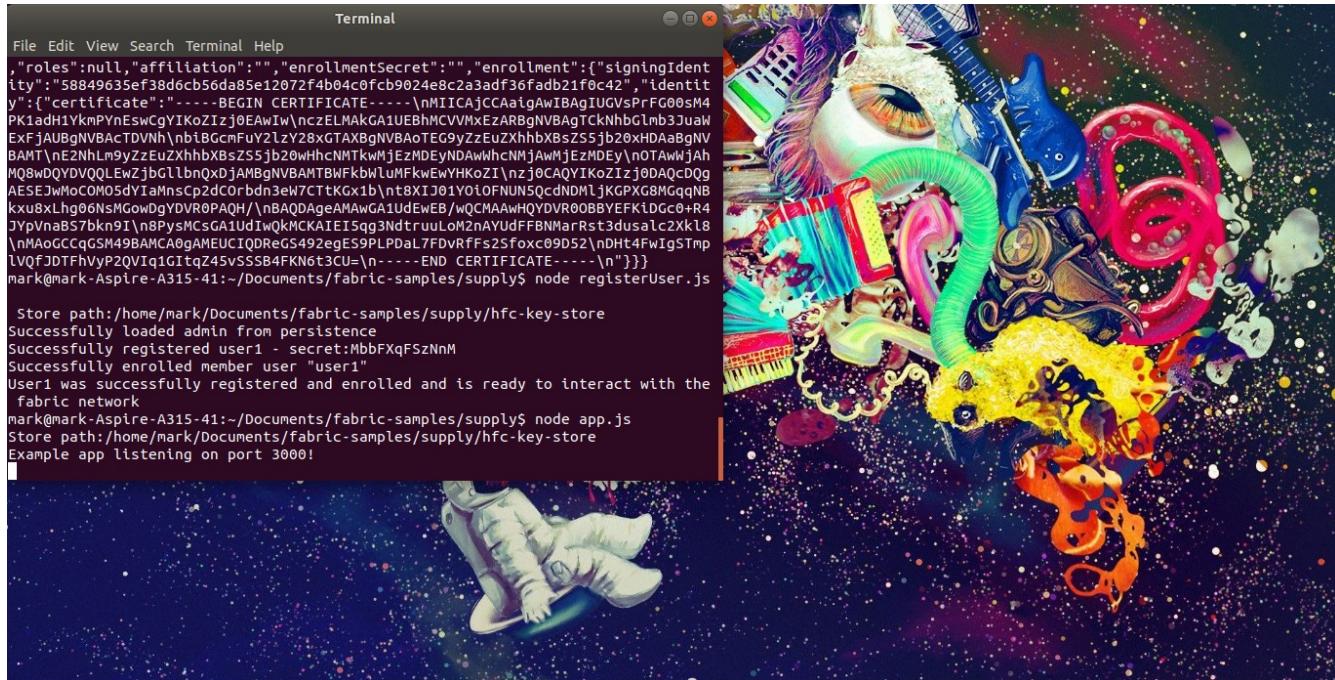
mark@mark-Aspire-A315-41:~/Documents/fabric-samples/supply$ node registerUser.js

  Store path:/home/mark/Documents/fabric-samples/supply/hfc-key-store
Successfully loaded admin from persistence
Successfully registered user1 - secret:MbbrFxqFSzNm
Successfully enrolled member user "user1"
User1 was successfully registered and enrolled and is ready to interact with the
fabric network
mark@mark-Aspire-A315-41:~/Documents/fabric-samples/supply$
```



### 9. Last, node app.js

## Mark John R. Frias



10. Then, open your postman and click send and you should see this:

```
[{"Key": "INVOICE0", "Record": "{'billedTo': 'OEM', \"goodReceived\": \"False\", \"invoiceAmount\": 200000, \"invoiceDate\": '02/07/19', \"isPaid\": \"False\", \"itemDescription\": \"Processor\", \"paidAmount\": 0, \"repaid\": \"False\", \"repaymentAmount\": 0}'}"], [{"Key": "INVOICE1", "Record": "{'billedTo': 'OEM', \"goodReceived\": \"False\", \"invoiceAmount\": 220000, \"invoiceDate\": '02/08/19', \"isPaid\": \"False\", \"itemDescription\": 'SSD', \"paidAmount\": 0, \"repaid\": \"False\", \"repaymentAmount\": 0}"}], [{"Key": "INVOICE2", "Record": "{'billedTo': 'OEM', \"goodReceived\": \"False\", \"invoiceAmount\": 140000, \"invoiceDate\": '02/09/19', \"isPaid\": \"False\", \"itemDescription\": 'RAM', \"paidAmount\": 0, \"repaid\": \"False\", \"repaymentAmount\": 0}"}], [{"Key": "INVOICE3", "Record": "{'billedTo': 'OEM', \"goodReceived\": \"False\", \"invoiceAmount\": 50000, \"invoiceDate\": '02/10/19', \"isPaid\": \"False\", \"itemDescription\": 'HDD', \"paidAmount\": 0, \"repaid\": \"False\", \"repaymentAmount\": 0}"}]
```

11. Now, we will push data using postman. Change the <sup>1</sup>"GET" to "POST" and then type this in the url bar <sup>2</sup>localhost:3000/invoice.

Mark John R. Frias

12. Click on <sup>3</sup>Body and select <sup>4</sup>x-www-form-urlencoded then click <sup>5</sup>Bulk Edit

13. Then, paste this:

```
invoicenumber:INVOICE6
billedto:OEM
invoicedate:02/08/19
invoiceamount:10000
itemdescription:KEYBOARD
goodreceived:False
ispaid:False
paidamount:0
repaid:False
repaymentamount:0
```

it should look like this:

The screenshot shows the Postman application interface. At the top, there are tabs for File, Edit, View, Help, New, Import, Runner, and a search bar. Below the tabs, there's a workspace titled "My Workspace" with an "Invite" button. The main area shows a history of requests on the left and a detailed view of a POST request to "localhost:3000/invoice" on the right.

**Request Details:**

- Method: POST
- URL: localhost:3000/invoice
- Headers: (1)
- Body (x-www-form-urlencoded):  
invoicenumber:INVOICE6  
billedto:OEM  
invoicedate:02/08/19  
invoiceamount:10000  
itemdescription:KEYBOARD  
goodreceived:False  
ispaid:False  
paidamount:0  
repaid:False  
repaymentamount:0

**Response Details:**

- Status: 200 OK
- Time: 1047 ms
- Size: 1.09 KB
- Body (Pretty):

```
1 [{"Key": "INVOICE0", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "200000", "invoiceDate": "02/07/19", "isPaid": "False", "itemDescription": "SSD", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}, {"Key": "INVOICE1", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "220000", "invoiceDate": "02/08/19", "isPaid": "False", "itemDescription": "SSD", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}, {"Key": "INVOICE2", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "14000", "invoiceDate": "02/09/19", "isPaid": "False", "itemDescription": "RAM", "paidAmount": "0"}]}
```

14. Then, click send and it will return success

## Mark John R. Frias

The screenshot shows the Postman application interface. On the left, there's a sidebar titled "History" with a "Clear all" button and a "Save Responses" toggle. The main area displays a sequence of API requests:

- A GET request to `localhost:3000/invoice`.
- A GET request to `localhost:3000/`.
- A POST request to `localhost:3000/invoice` with the following JSON body:

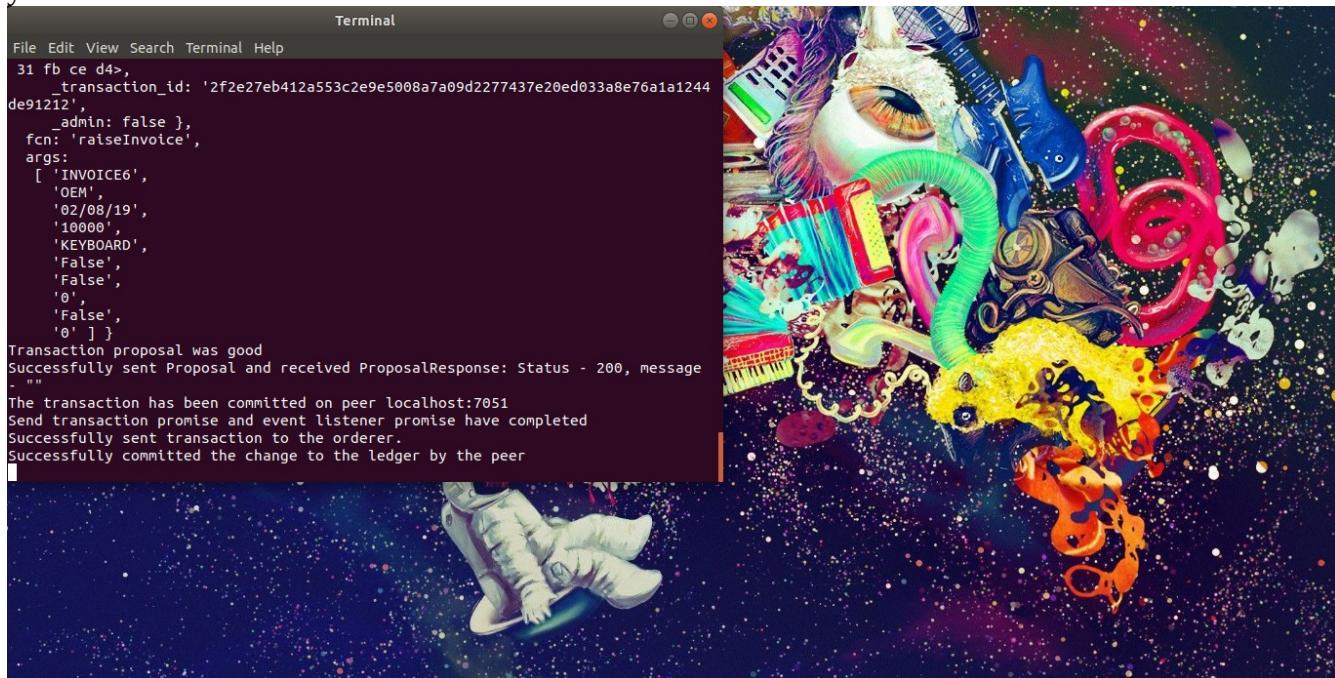
```
goodreceived:False  
ispaid:False  
paidamount:0  
repaid:False  
repaymentamount:0
```

The "Body" tab is selected, showing the JSON response:

```
1 = {  
2   "result": "success"  
3 }
```

At the bottom, status information is displayed: Status: 200 OK, Time: 3044 ms, Size: 264 B.

your terminal would looklike this:



Mark John R. Frias

15. click on Key-Value edit then change “POST” to “PUT” and follow this:

The screenshot shows the Postman application interface. At the top, there are tabs for File, Edit, View, Help, New, Import, Runner, and a search bar. Below the header, there's a workspace with three requests listed: GET localhost:3000/invoice, GET localhost:3000/, and PUT localhost:3000/. The PUT request is selected. The main area shows a PUT method against localhost:3000/invoice. The 'Body' tab is active, showing a table with key-value pairs. The table has columns for KEY, VALUE, and DESCRIPTION. The rows are:

KEY	VALUE	DESCRIPTION
invoicenumber	INVOICE6	
billedto	OEM	
invoicedate	02/08/19	
invoiceamount	10000	
itemdescription	KEYBOARD	
goodreceived	True	
ispaid	False	
paidamount	0	
repaid	False	
repaymentamount	0	
Key	Value	Description

At the bottom, there are buttons for Learn, Build, Browse, and a help icon.

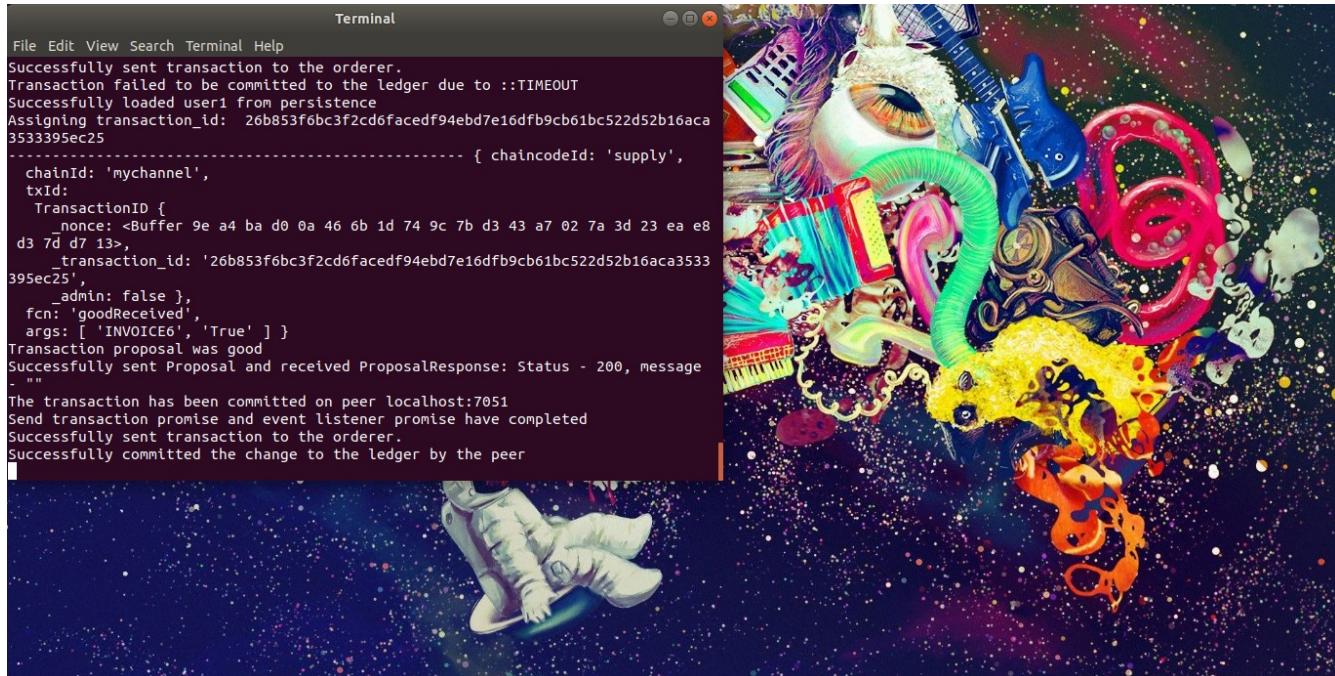
we will change the value of goodreceived to true if the good is received and then, click send:

The screenshot shows the Postman application interface after sending the PUT request. The status bar at the bottom indicates a 200 OK response with a time of 2987 ms and a size of 264 B. The response body is displayed in JSON format:

```
1: {  
2:   "result": "success"  
3: }
```

The rest of the interface is similar to the previous screenshot, showing the history and other requests.

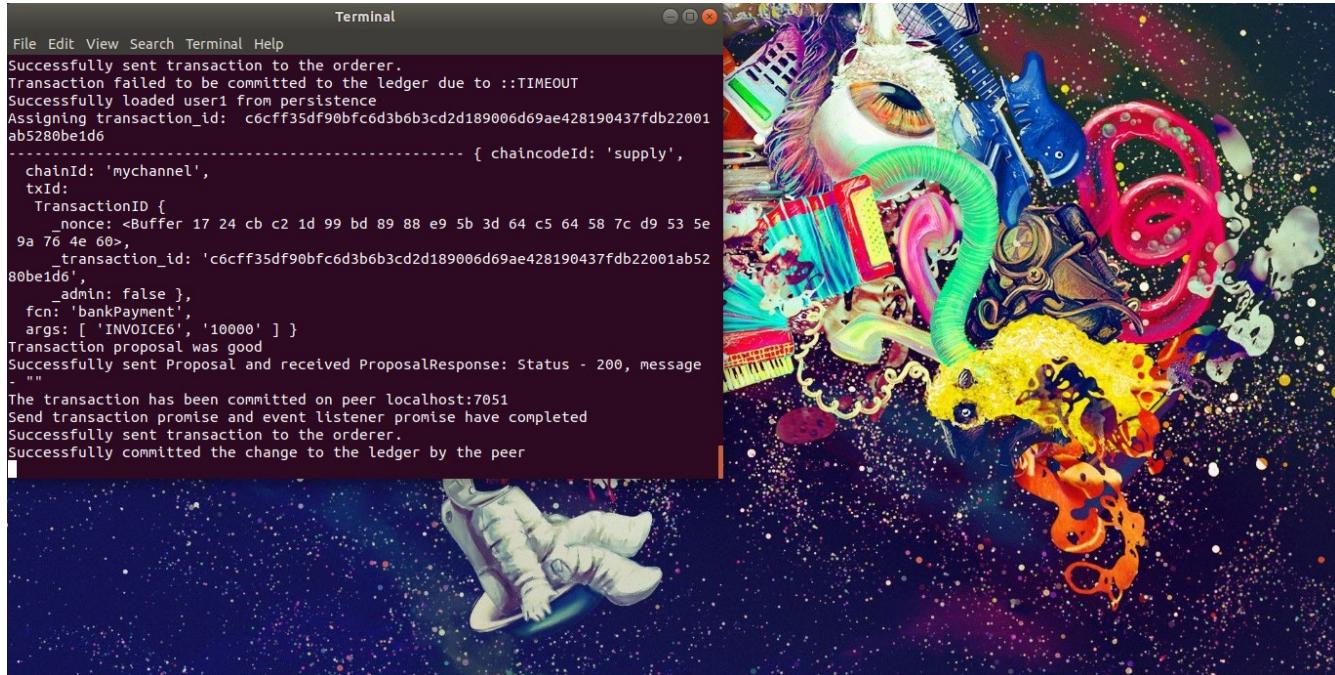
## Mark John R. Frias



Now, we will paid the good. Uncheck goodreceived and check paidamount and change the value from 0 to 10000 then click send.

A screenshot of the Postman application. The interface includes a top navigation bar with 'File', 'Edit', 'View', 'Help', and various tool icons. Below the navigation is a search bar and a 'My Workspace' dropdown. The main area is divided into sections: 'History' (listing previous requests), 'Collections' (empty), and a central workspace for the current request. The workspace shows a 'PUT' method for the URL 'localhost:3000/invoice'. The request body is a JSON object with fields: 'billedto' (OEM), 'invoicedate' (02/08/19), 'invoiceamount' (10000), 'itemdescription' (KEYBOARD), 'goodreceived' (True), 'ispaid' (False), 'paidamount' (10000, checked), 'repaid' (False), 'repaymentamount' (0), and 'Key' (empty). Below the body are tabs for 'Body', 'Cookies', 'Headers (7)', and 'Test Results'. The 'Test Results' tab shows a status of '200 OK' and a JSON response: { "result": "success" }. At the bottom are buttons for 'Send', 'Save', and 'Download', along with other application controls.

## Mark John R. Frias



Change “PUT” to “GET” and the url to localhost:3000 then click send to see the updated record.

File Edit View Search Terminal Help

Transaction proposal was good

Successfully sent Proposal and received ProposalResponse: Status - 200, message - ""

The transaction has been committed on peer localhost:7051

Send transaction promise and event listener promise have completed

Successfully sent transaction to the orderer.

Successfully committed the change to the ledger by the peer

Successfully loaded user1 from persistence

Query has completed, checking results

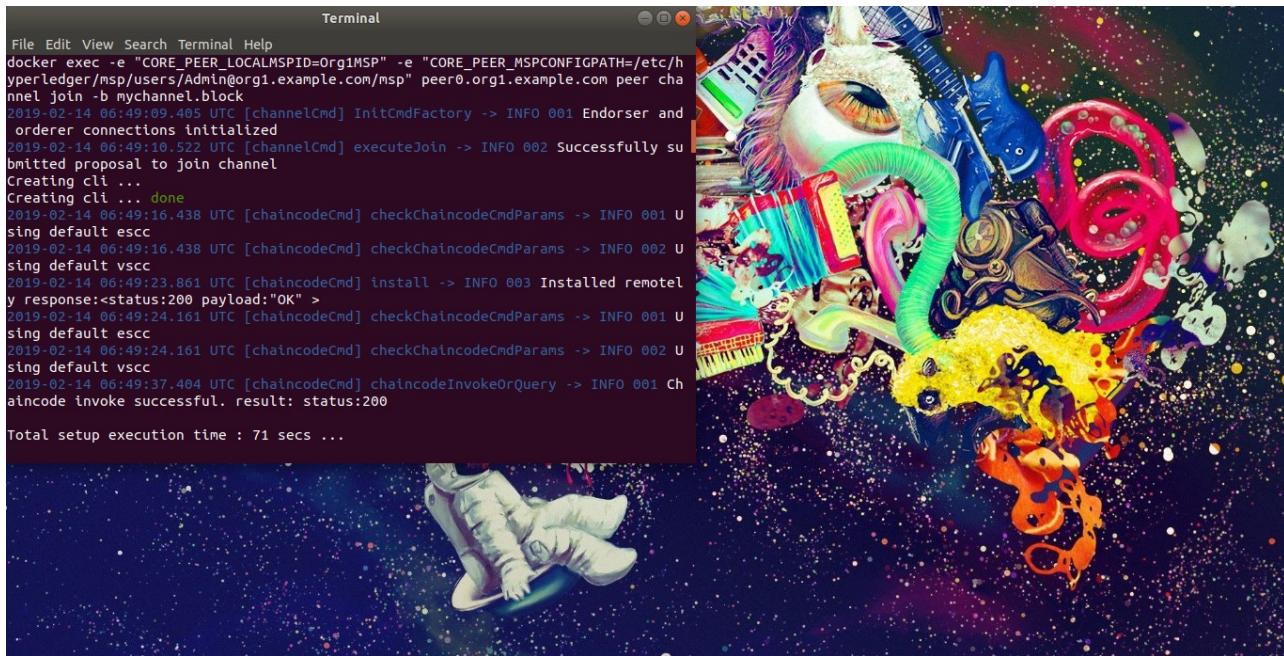
Response is [{"Key": "INVOICE0", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "200000", "invoiceDate": "02/07/19", "isPaid": "False", "itemDescription": "Processor", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}}, {"Key": "INVOICE1", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "22000", "invoiceDate": "02/08/19", "isPaid": "False", "itemDescription": "SSD", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}}, {"Key": "INVOICE2", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "14000", "invoiceDate": "02/09/19", "isPaid": "False", "itemDescription": "RAM", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}}, {"Key": "INVOICE3", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "50000", "invoiceDate": "02/10/19", "isPaid": "False", "itemDescription": "HDD", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}}, {"Key": "INVOICE4", "Record": {"billedTo": "OEM", "goodReceived": "True", "invoiceAmount": "10000", "invoiceDate": "02/08/19", "isPaid": "True", "itemDescription": "KEYBOARD", "paidAmount": "10000", "repaid": "False", "repaymentAmount": "0"}}, {"Key": "INVOICE5", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "200000", "invoiceDate": "02/07/19", "isPaid": "False", "itemDescription": "SSD", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}}, {"Key": "INVOICE6", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "14000", "invoiceDate": "02/09/19", "isPaid": "False", "itemDescription": "RAM", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}}, {"Key": "INVOICE7", "Record": {"billedTo": "OEM", "goodReceived": "False", "invoiceAmount": "50000", "invoiceDate": "02/10/19", "isPaid": "False", "itemDescription": "HDD", "paidAmount": "0", "repaid": "False", "repaymentAmount": "0"}}, {"Key": "INVOICE8", "Record": {"billedTo": "OEM", "goodReceived": "True", "invoiceAmount": "10000", "invoiceDate": "02/08/19", "isPaid": "True", "itemDescription": "KEYBOARD", "paidAmount": "10000", "repaid": "False", "repaymentAmount": "0"}}]

## Optional:

1. Open terminal then “git clone <https://github.com/khrandom/blockchain-training-labs>”
2. Open the folder that you cloned and paste it into your fabric-samples folder.
3. Then download the required library for our chaincode. Type this into your terminal:

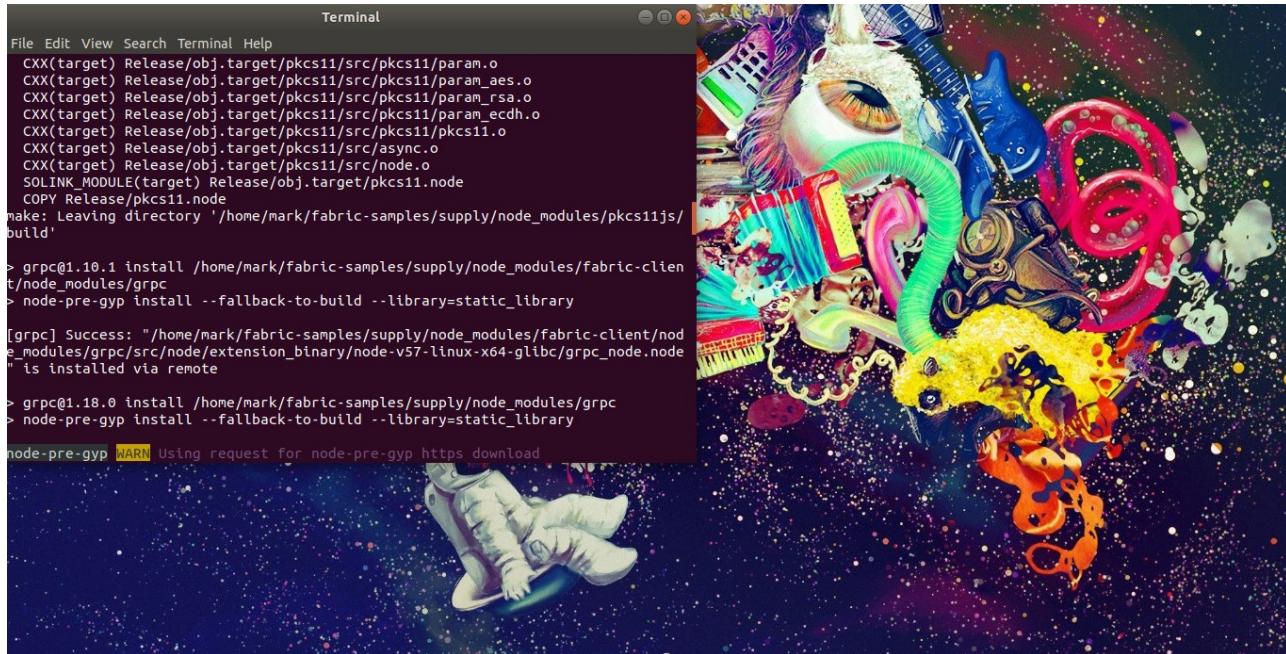
```
go get github.com/golang/protobuf/proto  
go get github.com/hyperledger/fabric/common/attrmgr  
go get github.com/pkg/errors  
go get github.com/hyperledger/fabric/core/chaincode/lib/cid
```

4. After downloading the library. Go to home/go/src/github.com/
5. Copy the 3 folder then paste it into your fabric-sample/chaincode.
6. Back to supply folder right click then open terminal and type this:  
.startFabric.sh  
should look like this:



7. npm install

## Mark John R. Frias



should look like this:

NOTE: if you get stuck just press **ctrl+c** then you may proceed.

8. Type node enrollAdmin.js

Mark John R. Frias

```
Terminal
File Edit View Search Terminal Help
^C
mark@mark-Aspire-A315-41:~/fabric-samples/supply$ node enrollAdmin.js
  Store path:/home/mark/fabric-samples/supply/hfc-key-store
Successfully enrolled admin user "admin"
Assigned the admin user to the fabric client ::>{"name":"admin","mspid":"Org1MSP"
,"roles":null,"affiliation":"","enrollmentSecret":"","enrollement":{"signingIdentity":{"certificate":"
-----BEGIN CERTIFICATE-----\nMIICAjCCAigAwIBAgIURNyDRR8YrjeF
ZWFLFO5jduQftg0CgyIKoIZj0EwiW\nczELMAkGA1UEBHMVCVMyEzARBgNVBAgTCNhbGImb3JuaWEf
jAUbgNVBAcTVDNh\nbiBGcmFuylzY28gTAXBgNVBAoTE9yZzEuXhbxBsZ5jb20xHDA8gNV
BAMT\nE2NhLm9yZzEuXhbxBsZ5jb20wHhcNMjE0MDY0NjAwKhcNMjE0MDY1NjMTAwJah
MQBwDQYDVQQLEwZjBgIibnQxDjAMBgNVBAMTBWFKbwIuMFkwEwYHKoZI\nnzj0CAQYIKoZIj0DAQcDg
AEwvmlHjPfcOGdzsqy8sMpDHKEsg3PWUE0IeUP\nvxveCEDIWp7OTnA5P0kdng8crRsfDhnlrjw/E
9yJhs0w4anshGowDgDVROPAQH/\nbAQdAgeAMAWGA1UeEWEB/wQCMIAwQYDVR0BBYEF1jDTT0t+
8tR1dyu6HHzel\nuUMCMCSGA1UDiwQkMCKAIE15gq3NDtruuLoM2nAYUDFBNNMarRst3dusalc2XkL8
\nnAoGCCqGSM49BAMCAgAMEUCIQCT+8tDxuNFLvssJuthbqSknniLH08mkvzJ39\\nRp7kRAIg5spq
LugrbR542mRLsDCYdAyqa0Ua0drNoH80BJHyo=n-----END CERTIFICATE-----\n"}}
mark@mark-Aspire-A315-41:~/fabric-samples/supply$ node registerSupplier.js
  Store path:/home/mark/fabric-samples/supply/hfc-key-store
Successfully loaded admin from persistence
Successfully registered supplier - secret:YcmpMaNQpmI
Successfully enrolled member user "supplier"
supplier was successfully registered and enrolled and is ready to interact with
the fabric network
```



should look like this.

9. node registerSupplier.sh

10. node registerOem.sh

11. node registerBank.sh

```
Terminal
File Edit View Search Terminal Help
mark@mark-Aspire-A315-41:~/fabric-samples/supply$ node registerSupplier.js
  Store path:/home/mark/fabric-samples/supply/hfc-key-store
Successfully loaded admin from persistence
Successfully registered supplier - secret:YcmpMaNQpmI
Successfully enrolled member user "supplier"
supplier was successfully registered and enrolled and is ready to interact with
the fabric network
mark@mark-Aspire-A315-41:~/fabric-samples/supply$ node registerOEM.js
  Store path:/home/mark/fabric-samples/supply/hfc-key-store
Successfully loaded admin from persistence
Successfully registered oem - secret:VExtosmMQeft
Successfully enrolled member user "oem"
oem was successfully registered and enrolled and is ready to interact with the
fabric network
mark@mark-Aspire-A315-41:~/fabric-samples/supply$ node registerBank.js
  Store path:/home/mark/fabric-samples/supply/hfc-key-store
Successfully loaded admin from persistence
Successfully registered bank - secret:DItqyLXVJjWl
Successfully enrolled member user "bank"
bank was successfully registered and enrolled and is ready to interact with the
fabric network
mark@mark-Aspire-A315-41:~/fabric-samples/supply$ node app.js
  Store path:/home/mark/fabric-samples/supply/hfc-key-store
Example app listening on port 3000!
  Successfully loaded supplier from persistence
```



you should see this.

12. node app.js

Mark John R. Frias

terminal should output this:

Store path:/home/mark/fabric-samples/supply/hfc-key-store

Example app listening on port 3000!

13. we will check if its running. Open postman type in URL bar localhost:3000/invoice click header then add in content-type: user then value is supplier and change "GET" to "POST"

The screenshot shows a Postman interface with a POST request to 'localhost:3000/invoice'. The 'Headers' tab is selected, displaying two entries: 'Content-Type' with value 'application/x-www-form-urlencoded' and 'user' with value 'supplier'. Other tabs like 'Params', 'Authorization', 'Body', 'Pre-request Script', and 'Tests' are visible. Buttons for 'Send' and 'Save' are at the top right.

should look like this.

14. Then click on body and click Bulk Edit then paste this:

invoicenumber:INVOICE6

billedto:OEM

invoicedate:02/08/19

invoiceamount:10000

itemdescription:KEYBOARD

goodreceived:False

ispaid:False

paidamount:0

repaid:False

repaymentamount:0

15. click send and wait for the request to finish and it will return success

The screenshot shows a Postman interface with a POST request to 'localhost:3000/invoice'. The 'Body' tab is selected, showing a key-value edit field with the following data:  
invoicenumber:INVOICE6  
billedto:OEM  
invoicedate:02/08/19  
invoiceamount:10000  
itemdescription:KEYBOARD  
goodreceived:False  
ispaid:False  
paidamount:0  
repaid:False  
repaymentamount:0

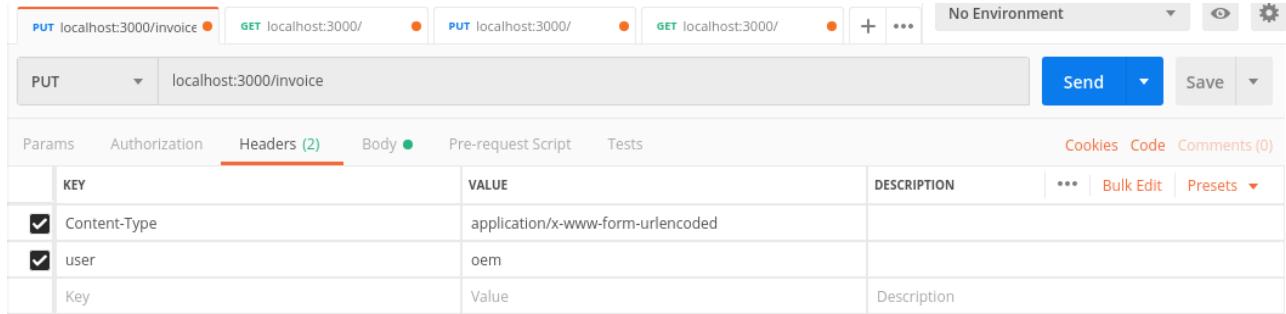
At the bottom, the status is shown as 'Status: 200 OK Time: 2957 ms Size: 264 B'. Below that, there are tabs for 'Pretty', 'Raw', 'Preview', and 'JSON'. The JSON preview shows a single object with a 'result' key.

```
1 ↴ {  
2   "result": "success"  
3 }
```

Mark John R. Frias

your terminal should look like this:

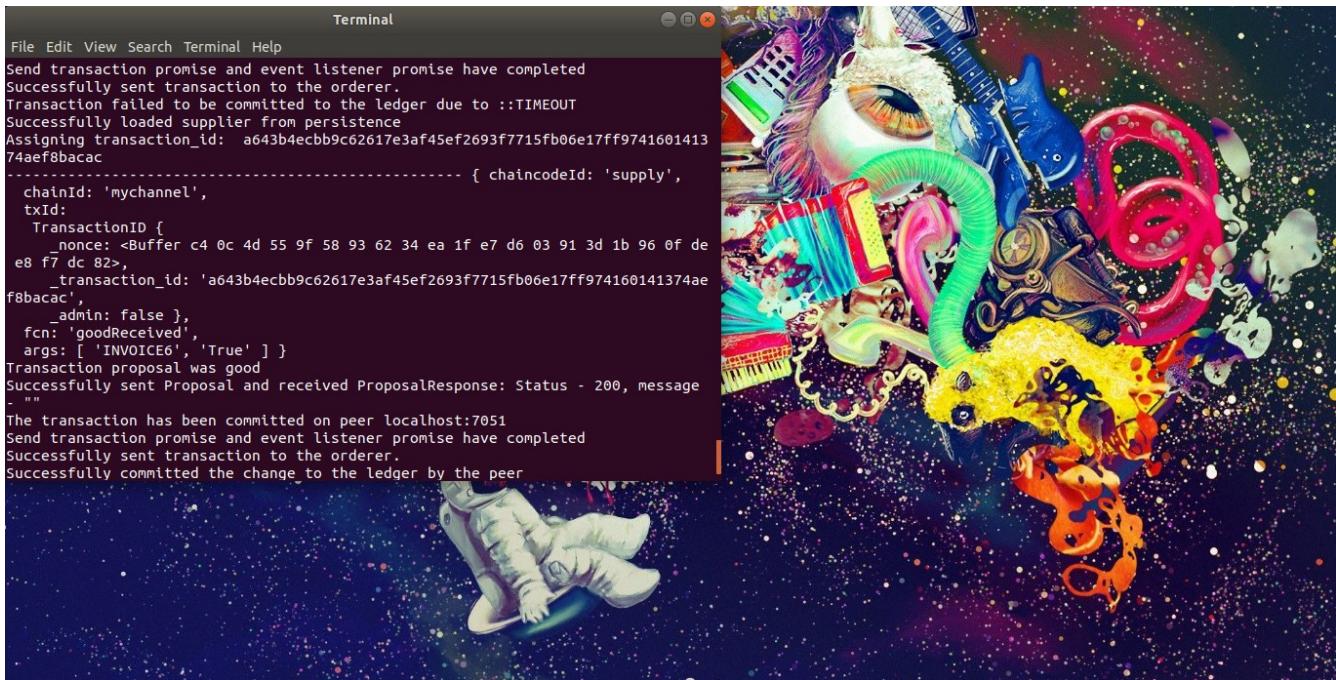
16. Now we will set goodreceived to true. Click on headers change the value of user to oem



The screenshot shows the Postman application interface. A PUT request is being made to `localhost:3000/invoice`. The Headers section is active, containing two entries: `Content-Type` set to `application/x-www-form-urlencoded` and `user` set to `oem`. Other tabs like Params, Authorization, Body, and Pre-request Script are visible.

should look like this.

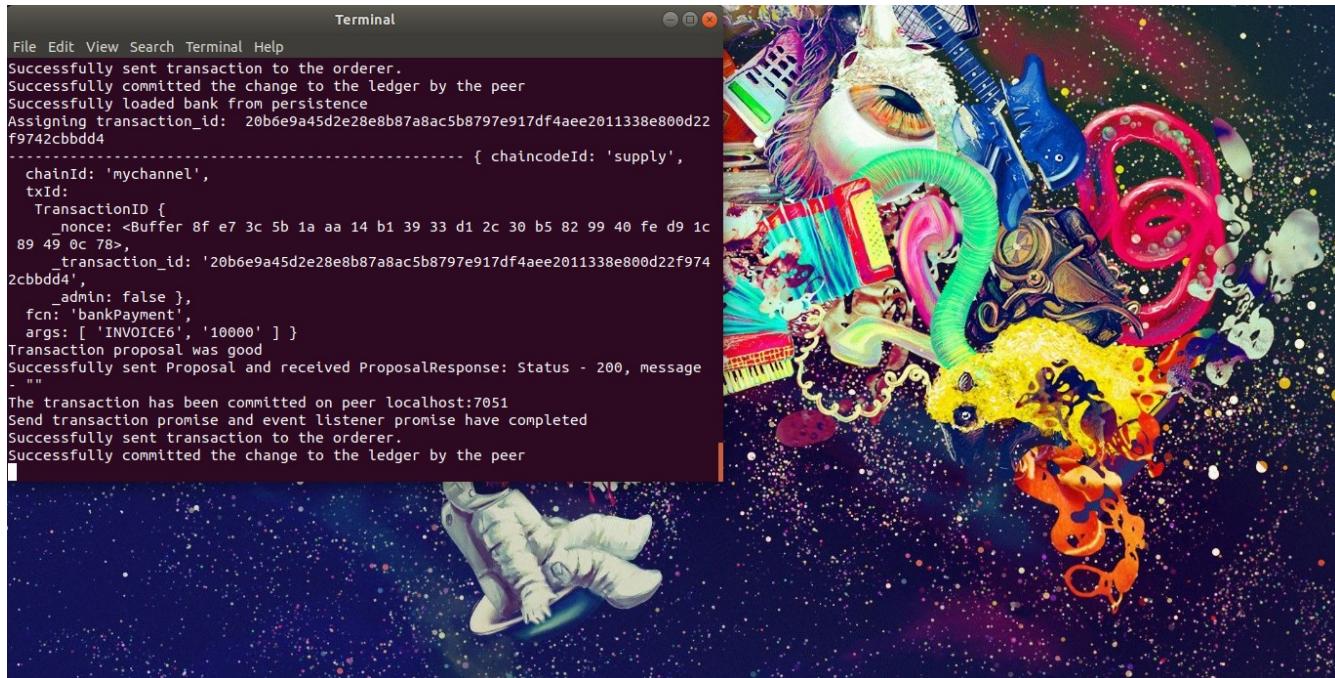
17. Go to body and uncheck all keys except “invoicenumber” and “goodreceived” and change the value of goodreceived to true



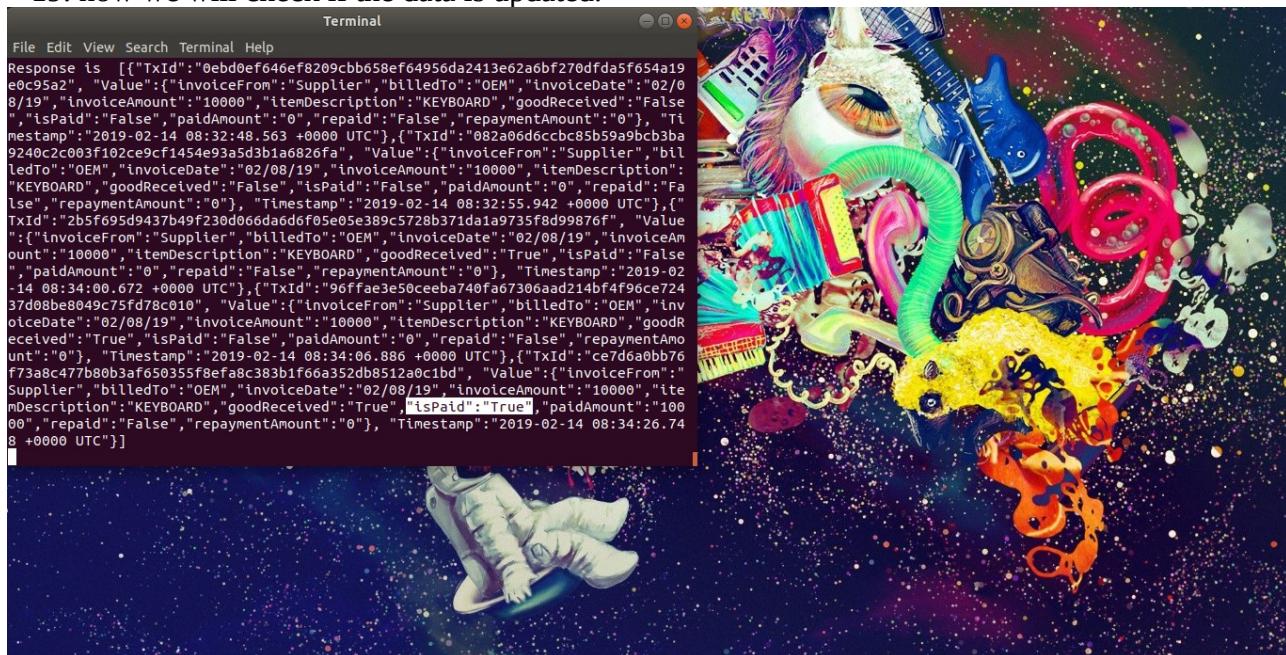
your terminal should look like this

Mark John R. Frias

18. now we will pay the supplier. Click on header and change the user value to “bank” and go to body and uncheck all except “invoicenumber” and “paidamount” then change the value of paidamount to “10000” and click send



19. now we will check if the data is updated.

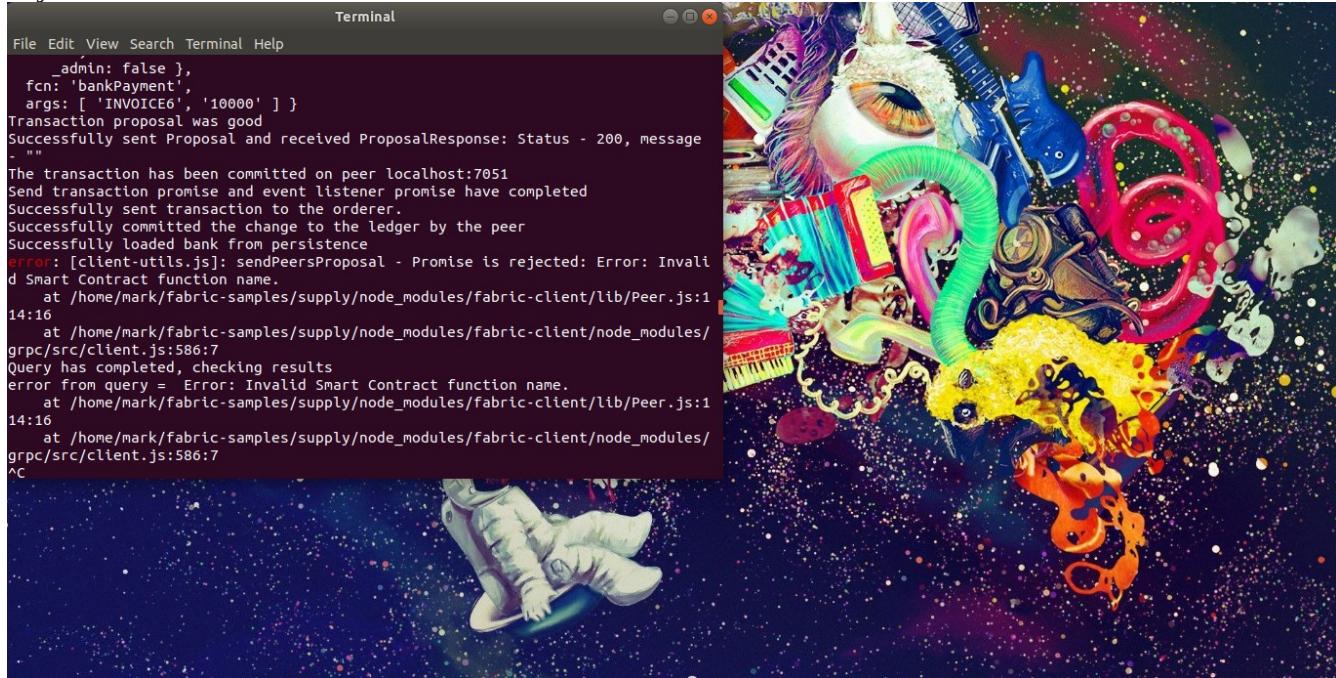


Now you can see the the isPaid key value is updated to “True” when we enter the paidamount.

Mark John R. Frias

NOTE:

If you encounter this kind of error:



you need to kill your docker by typing this into your terminal.

First right click on your fabric-samples/supply open terminal then type:

```
docker kill $(docker ps -q)
docker rm $(docker ps -aq)
docker rmi $(docker images dev-* -q)
```

then repeat from step 6 to 19.

Mark John R. Frias

REFERENCE:

<https://hyperledger.github.io/composer/latest/installing/installing-prereqs.html>

<https://golang.org/>

<https://github.com/khrandm/blockchain-training-labs>