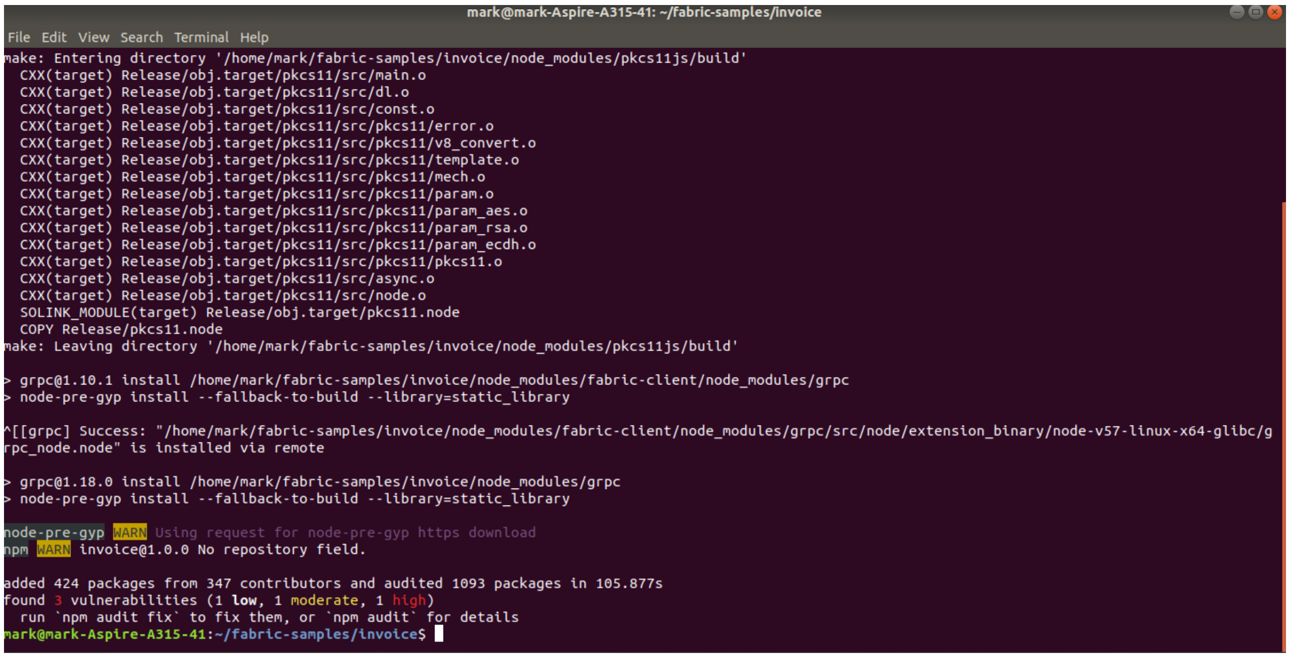
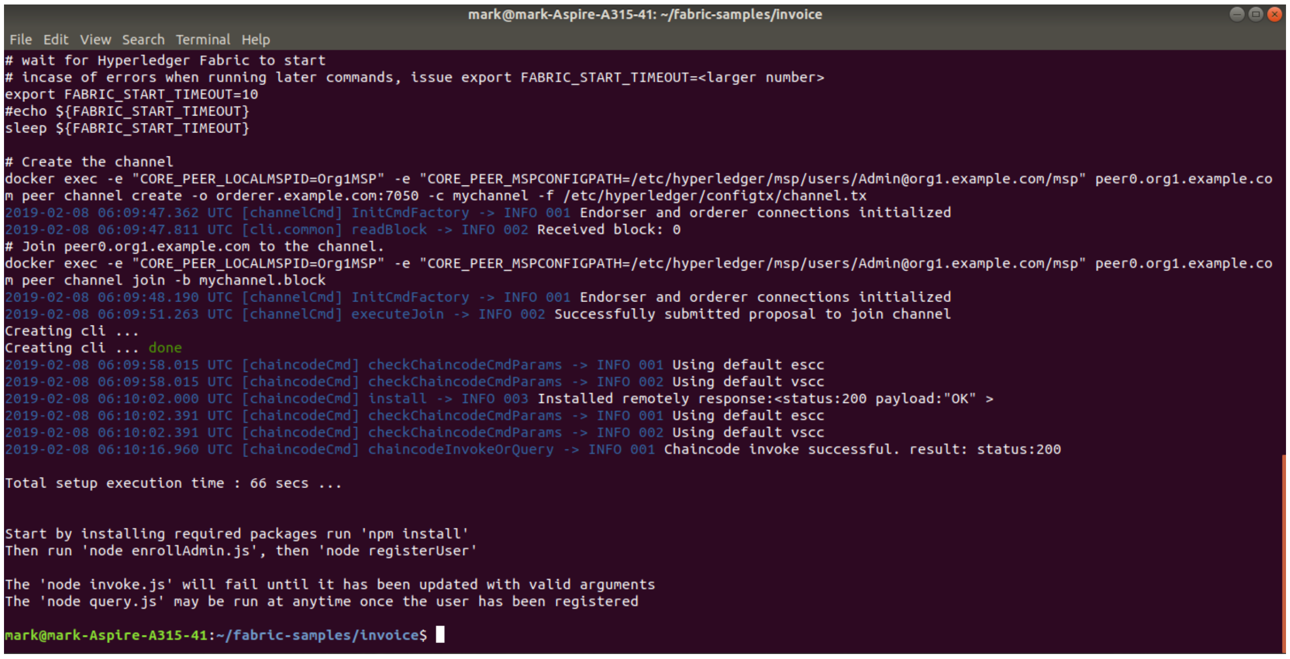
1. To clone this repository open terminal and type :
   “ **mkdir hyperledger && cd hyperledger** “
   “ curl -sSL http://bit.ly/2ysbOFE | bash -s – 1.4.0 “
   “ git clone https://github.com/jenriellegaon/blockchain-training-labs.git “
2. Open the **blockchain-training-labs** folder then go to **node** folder and copy all files except the **node-modules folder.**
3. Go to **fabric-sample** folder then create **invoice** folder and paste the file you copied.
4. Back to **blockchain training labs** directory then copy “**go**” folder
5. Back again to **fabric-sample** directory and open **chaincode** folder. Then create **invoice** folder and paste the **go** directory
6. Go back to the **fabric sample** directory, right-click then **Open in Terminal** and type **“npm install”**



```
mark@mark-Aspire-A315-41: ~/fabric-samples/invoice
File  Edit  View  Search  Terminal  Help
make: Entering directory '/home/mark/fabric-samples/invoice/node_modules/pkcs11js/build'
  CXX(target) Release/obj.target/pkcs11/src/main.o
  CXX(target) Release/obj.target/pkcs11/src/dl.o
  CXX(target) Release/obj.target/pkcs11/src/const.o
  CXX(target) Release/obj.target/pkcs11/src/pkcs11/error.o
  CXX(target) Release/obj.target/pkcs11/src/pkcs11/v8_convert.o
  CXX(target) Release/obj.target/pkcs11/src/pkcs11/template.o
  CXX(target) Release/obj.target/pkcs11/src/pkcs11/mech.o
  CXX(target) Release/obj.target/pkcs11/src/pkcs11/param.o
  CXX(target) Release/obj.target/pkcs11/src/pkcs11/param_aes.o
  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/fabric-samples/invoice/node_modules/pkcs11js/build'

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

^[[grpc] Success: "/home/mark/fabric-samples/invoice/node_modules/fabric-client/node_modules/grpc/src/node/extension_binary/node-v57-linux-x64-glibc/g
rpc_node.node" is installed via remote

> grpc@1.18.0 install /home/mark/fabric-samples/invoice/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
npm WARN invoice@1.0.0 No repository field.

added 424 packages from 347 contributors and audited 1093 packages in 105.877s
found 3 vulnerabilities (1 low, 1 moderate, 1 high)
  run `npm audit fix` to fix them, or `npm audit` for details
mark@mark-Aspire-A315-41:~/fabric-samples/invoice$
```

NOTE: If get stuck in this node-pre-gyp WARN Using request for node-pre-gyp https download
just “Ctrl + C”.

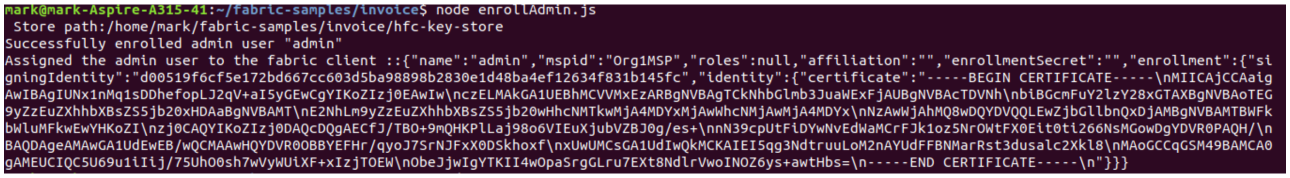
7. Type in the terminal
   "./startFabric.sh"
   you should see this:



```
mark@mark-Aspire-A315-41: ~/fabric-samples/invoice
File Edit View Search Terminal Help
# wait for Hyperledger Fabric to start
# incase of errors when running later commands, issue export FABRIC_START_TIMEOUT=<larger number>
export FABRIC_START_TIMEOUT=10
#echo ${FABRIC_START_TIMEOUT}
sleep ${FABRIC_START_TIMEOUT}

# Create the channel
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example.com/msp" peer0.org1.example.co
m peer channel create -o orderer.example.com:7050 -c mychannel -f /etc/hyperledger/configtx/channel.tx
2019-02-08 06:09:47.362 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2019-02-08 06:09:47.811 UTC [cli.common] readBlock -> INFO 002 Received block: 0
# Join peer0.org1.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example.com/msp" peer0.org1.example.co
m peer channel join -b mychannel.block
2019-02-08 06:09:48.190 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2019-02-08 06:09:51.263 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
Creating cli ...
Creating cli ... done
2019-02-08 06:09:58.015 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc
2019-02-08 06:09:58.015 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
2019-02-08 06:10:02.000 UTC [chaincodeCmd] install -> INFO 003 Installed remotely response:<status:200 payload:"OK" >
2019-02-08 06:10:02.391 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc
2019-02-08 06:10:02.391 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
2019-02-08 06:10:16.960 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:~/fabric-samples/invoice$ 
```

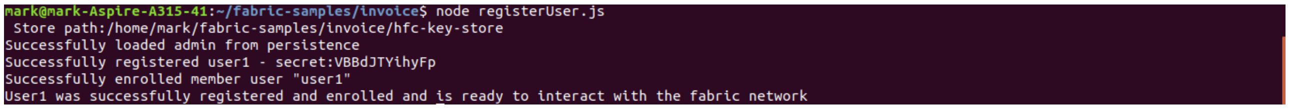
8. You should update your chaincode by typing this:
   - docker exec -it cli bash
   - peer chaincode install -n invoice -v 3.5 -l "golang" -p "github.com/invoice/go"
   - peer chaincode upgrade -n invoice -v 3.5 -o orderer.example.com:7050 -C mychannel -l "golang" -p "github.com/invoice/go" -c '{"Args":[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"

9. Then "**node enrollUser.js**"
   you should see this:



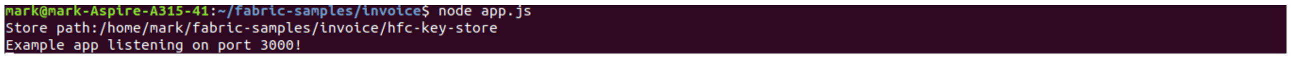
```
mark@mark-Aspire-A315-41:~/fabric-samples/invoice$ node enrollAdmin.js
 Store path:/home/mark/fabric-samples/invoice/hfc-key-store
Successfully enrolled admin user "admin"
Assigned the admin user to the fabric client ::{"name":"admin","mspid":"Org1MSP","roles":null,"affiliation":"","enrollmentSecret":"","enrollment":{"si
gningIdentity":"d00519f6cf5e172bd667cc603d5ba98898b2830e1d48ba4ef12634f831b145fc","identity":{"certificate":"-----BEGIN CERTIFICATE-----\nMIICAjCCAaig
AwIBAgIUNx1nMq1sDDhefopLJ2qV+aI5yGEwCgYIKoZIzj0EAwIw\nczELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNh\nbiBGcmFuY2lzY28xGTAXBgNVBAoTEG
9yZzEuZXhhbXBsZS5jb20xHDAaBgNVBAMT\nE2NhLm9yZzEuZXhhbXBsZS5jb20wHhcNMTkwMjA4MDYxMjAwWhcNMjAwMjA4MDYx\nNzAwWjAhMQ8wDQYDVQQLEwZjbGllbnQxDjAMBgNVBAMTBWFk
bWluMFkwEwYHKoZI\nzj0CAQYIKoZIzj0DAQcDQgAECfJ/TBO+9mQHKPlLaj98o6VIEuXjubVZBJ0g/es+\nnN39cpUtFiDYwNvEdWaMCrFJk1oz5NrOWtFX0Eit0ti266NsMGowDgYDVR0PAQH/\n
BAQDAgeAMAwGA1UdEwEB/wQCMAAwHQYDVR0OBBYEFHr/qyoJ7SrNJFxX0DSkhoxf\nxUwUMCsGA1UdIwQkMCKAIEI5qg3NdtruuLoM2nAYUdFFBNMarRst3dusalc2Xkl8\nMAoGCCqGSM49BAMCA0
gAMEUCIQC5U69u1iIij/75UhO0sh7wVyWUiXF+xIzjTOEW\nObeJjwIgYTKII4wOpaSrgGLru7EXt8NdlrVwoINOZ6ys+awtHbs=\n-----END CERTIFICATE-----\n"}}}
```

10. Next "**node registerUser.js**"
    you should see this:



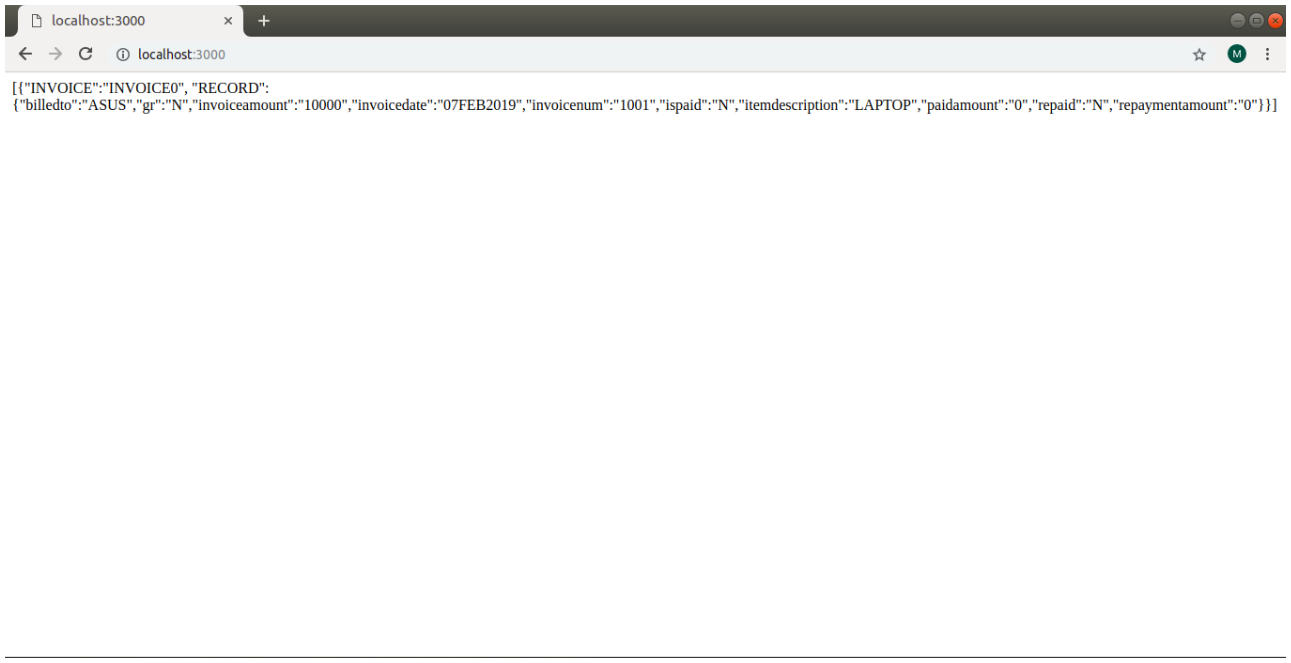
```
mark@mark-Aspire-A315-41:~/fabric-samples/invoice$ node registerUser.js
 Store path:/home/mark/fabric-samples/invoice/hfc-key-store
Successfully loaded admin from persistence
Successfully registered user1 - secret:VBBdJTYihyFp
Successfully enrolled member user "user1"
User1 was successfully registered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

11. Then type “**node app.js**”
    you should see this:



```
mark@mark-Aspire-A315-41:~/fabric-samples/invoice$ node app.js
Store path:/home/mark/fabric-samples/invoice/hfc-key-store
Example app listening on port 3000!
```

12. To test type “http://localhost:3000/” on your browser.
    you should see this:



```
[{"INVOICE":"INVOICE0", "RECORD":
{"billedto":"ASUS","gr":"N","invoiceamount":"10000","invoicedate":"07FEB2019","invoicenum":"1001","ispaid":"N","itemdescription":"LAPTOP","paidamount":"0","repaid":"N","repaymentamount":"0"}}]
```

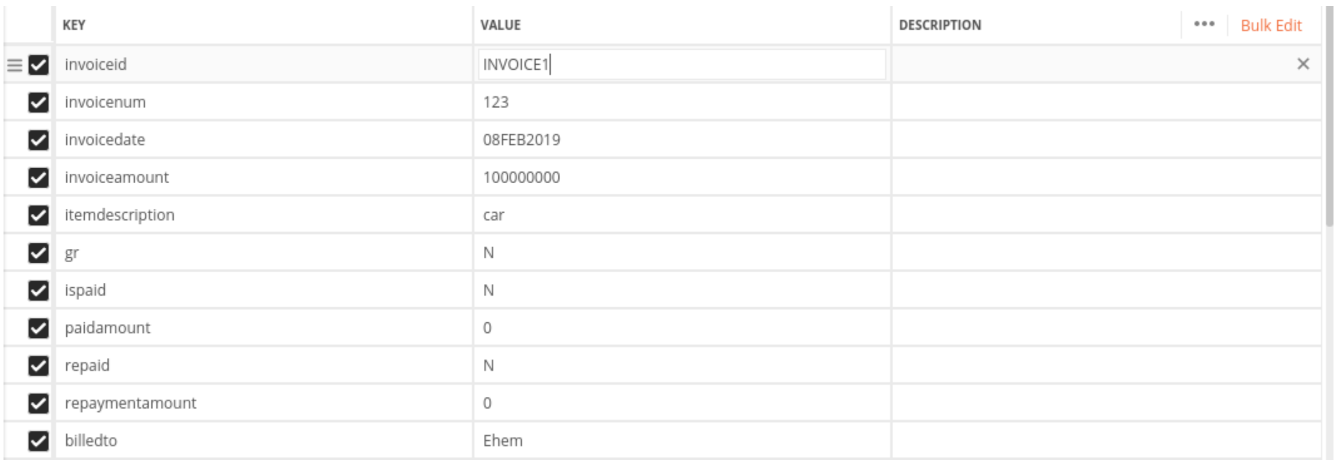
13. Open **postman** to push or add data.
    NOTE: If you don’t have postman, to download type in the terminal “**snap install postman**”
14. Select “**POST**” then type “**localhost:3000/invoice**”
15. Switch to the Body Tab and select x-www-form-urlencoded and add this parameters as a key:

- invoiceid
- invoicenum
- billedto
- invoicedate
- invoiceamount
- itemsdescription
- gr
- ispaid

- paidamount
- repaid
- repaymentamount

NOTE:
This key has default value of:
gr = N
ispaid = N
paidamount = 0
repaid = N
repaymentamount = 0

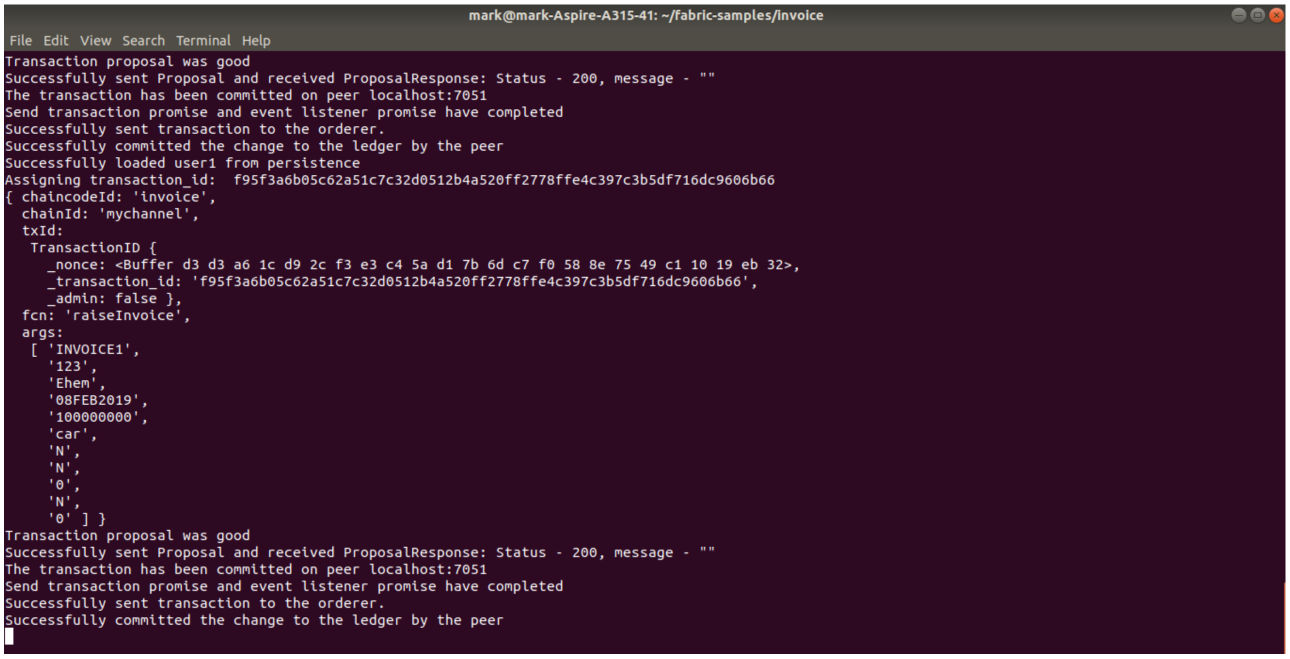
16. Then enter the this value:
NOTE: you can add your preffered value.



	KEY	VALUE	DESCRIPTION
✓	invoiceid	INVOICE1	
✓	invoicenum	123	
✓	invoicedate	08FEB2019	
✓	invoiceamount	100000000	
✓	itemdescription	car	
✓	gr	N	
✓	ispaid	N	
✓	paidamount	0	
✓	repaid	N	
✓	repaymentamount	0	
✓	billedto	Ehem	

17. Click the **Send** button.

    You should see this:


mark@mark-Aspire-A315-41: ~/fabric-samples/invoice
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
Assigning transaction_id:  f95f3a6b05c62a51c7c32d0512b4a520ff2778ffe4c397c3b5df716dc9606b66
{ chaincodeId: 'invoice',
  chainId: 'mychannel',
  txId:
   TransactionID {
     _nonce: <Buffer d3 d3 a6 1c d9 2c f3 e3 c4 5a d1 7b 6d c7 f0 58 8e 75 49 c1 10 19 eb 32>,
     _transaction_id: 'f95f3a6b05c62a51c7c32d0512b4a520ff2778ffe4c397c3b5df716dc9606b66',
     _admin: false },
  fcn: 'raiseInvoice',
  args:
   [ 'INVOICE1',
     '123',
     'Ehem',
     '08FEB2019',
     '100000000',
     'car',
     'N',
     'N',
     '0',
     'N',
     '0' ] }
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
```

18. To update the data inside:

    Select “**PUT**” then type “**localhost:3000/invoice**”

    Switch to the Body Tab and select x-www-form-urlencoded and uncheck all keys except the invoiceid and the keys you want to update.

19. Then click “**Send**” button.

20. To view the data inside: Select “**GET**” then type “**localhost:3000/**” and it will return all data.

REFERENCE:

URL: https://github.com/jenriellegaon/blockchain-training-labs/
AUTHOR: Jenrielle Gaon
John Carlo Cuya
Joshua Caramancion
Ron Vincent Conde
ACCESS: February 9, 2019