Alitagtag, Collin R.

Blockchain Cadet

Group 2

Getting Started with Hyperledger Fabric on Windows 10

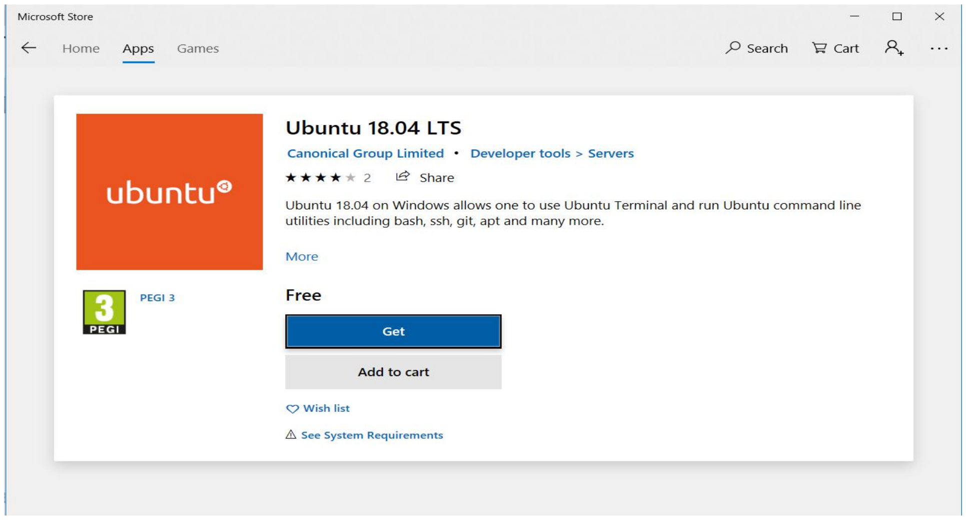
Laptop Specification

Manufacturer	ASUStek Computer Inc.
Unit	VivoBook S510UVivoBook S510U
Processor	Intel® Core™ i7-855OU
RAM	8 GB DDR4
OS	Windows 10 PRO
OS TYPE	64 Bit

Installation Method

1. You'll need the Windows Subsystem for Linux to run Ubuntu on Windows10.

   Go to Microsoft Store and search "Ubuntu" and Download it.



Ubuntu 18.04 on Windows allows one to use Ubuntu Terminal and run Ubuntu command line utilities including bash, ssh, git, apt and many more.

2. Install Node

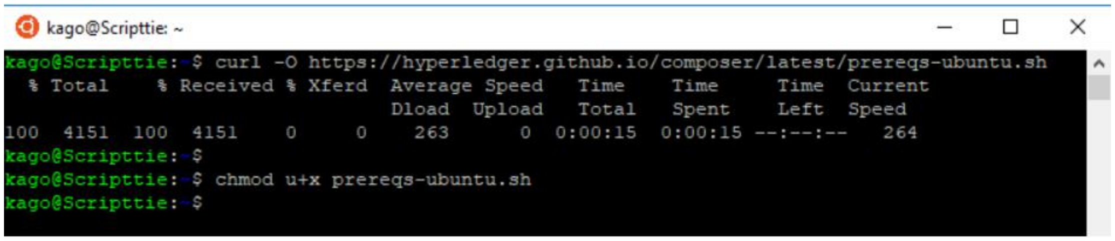
https://nodejs.org/en/download/package-manager/#debian-and-ubuntu-based-linux-distributions

3. Install Docker

https://medium.com/@sebagomez/installing-the-docker-client-on-ubuntus-windows-subsystem-for-linux-612b392a44c4

4. Open your terminal and start typing the following commands.

curl -O https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh



```
kago@Scripttie: ~
kago@Scripttie:~$ curl -O https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4151  100  4151    0     0    263      0  0:00:15  0:00:15 --:--:--   264
kago@Scripttie:~$
kago@Scripttie:~$ chmod u+x prereqs-ubuntu.sh
kago@Scripttie:~$
```

chmod u+x prereqs-ubuntu.sh

5. Install Go language
https://golang.org/dl/
download Go for Linux

Go to the directory where the file located.
Right click inside the foler and Open in terminal
Type "**tar -C /usr/local -xzf go**" then press tab to complete then enter it should be look like this Eg: "tar -C /usr/local -xzf go1.11.5.linux-amd64.tar.gz"

Add Go environment variable
Type this:
export PATH=$PATH:/usr/local/go/bin
export GOPATH=$HOME/go

6. Download Fabric samples
Open terminal
Type "**git clone https://github.com/hyperledger/fabric-samples.git**"
Type **cd fabric-sample**

7. Download Image and Binaries
Type "**curl -sSL http://bit.ly/2ysbOFE | bash -s -- 1.4.0**"

INVOICE TRACKING

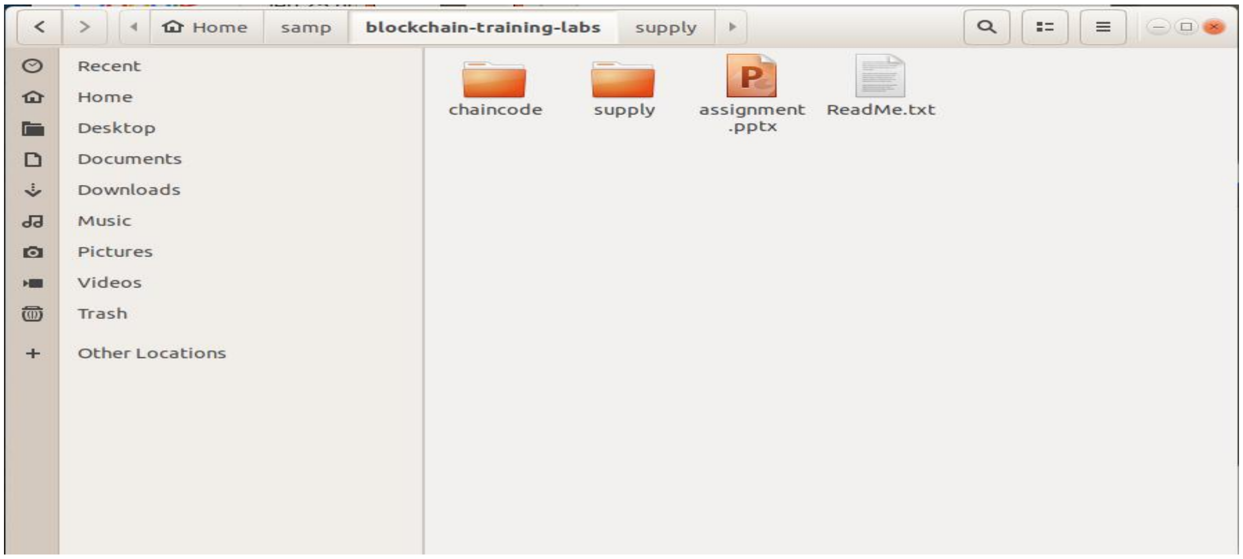
Step 1: Cloning repository

Open your CMD, then type the following commands:

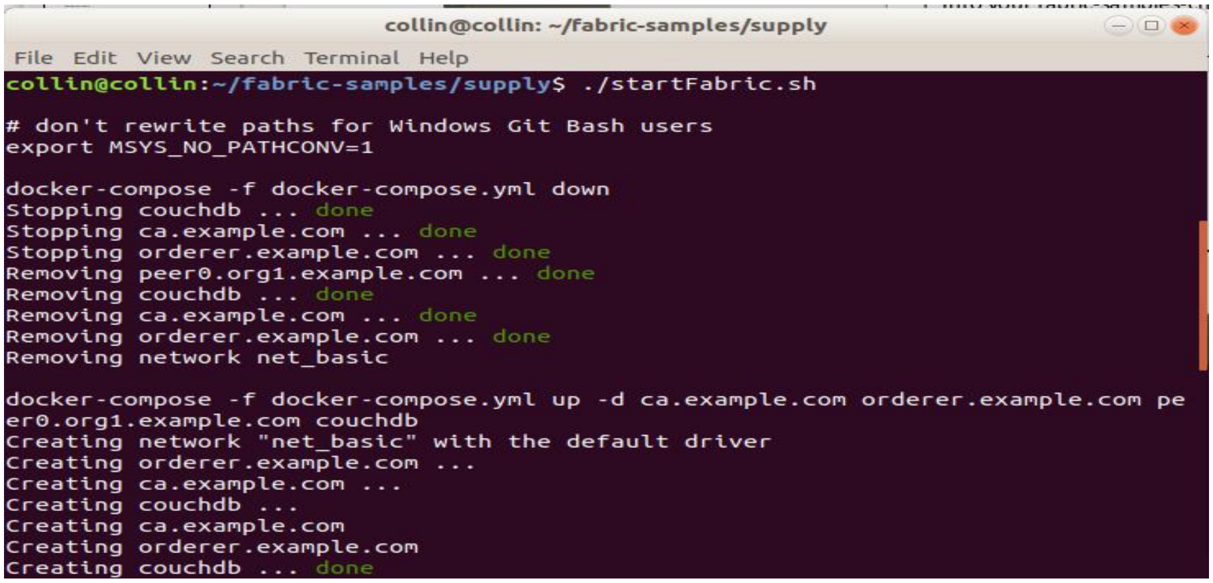
git clone https://github.com/hyperledger/fabric-samples

git clone https://github.com/collin1517/blockchain-training-labs

Step 2: Copy the chaincode and supply folder, paste it in the fabric-samples folder then merge it.



Step 3: Open terminal In your fabric-samples folder.

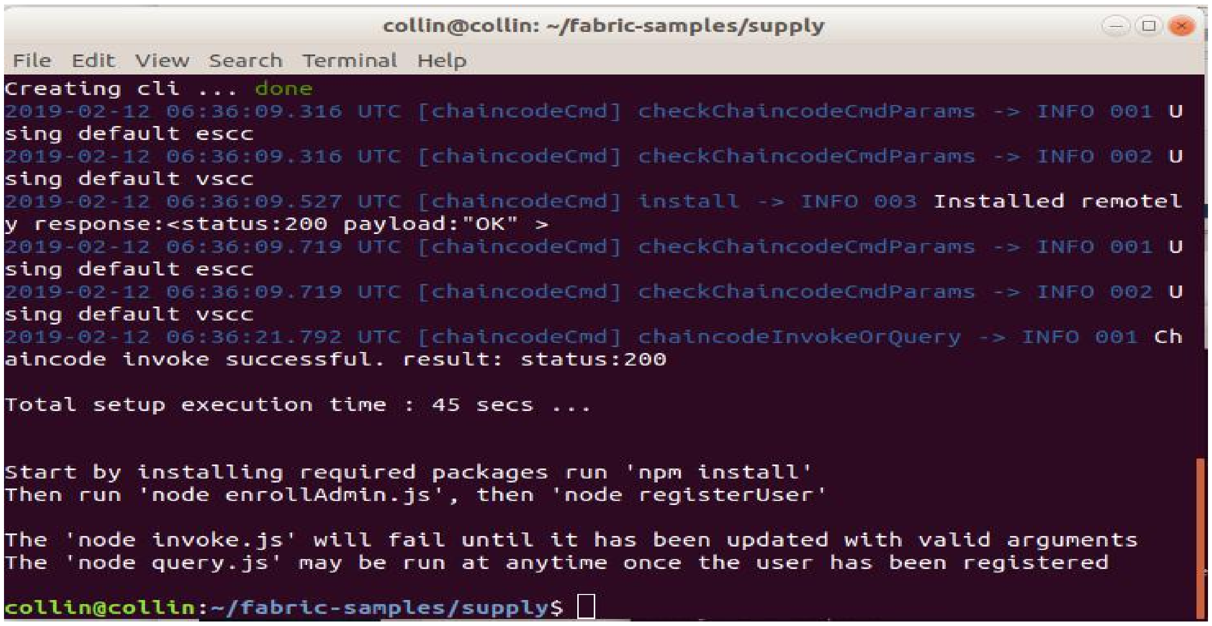

collin@collin: ~/fabric-samples/supply
File Edit View Search Terminal Help

```
collin@collin:~/fabric-samples/supply$ ./startFabric.sh

# don't rewrite paths for Windows Git Bash users
export MSYS_NO_PATHCONV=1

docker-compose -f docker-compose.yml down
Stopping couchdb ... done
Stopping ca.example.com ... done
Stopping orderer.example.com ... done
Removing peer0.org1.example.com ... done
Removing couchdb ... done
Removing ca.example.com ... done
Removing orderer.example.com ... done
Removing network net_basic

docker-compose -f docker-compose.yml up -d ca.example.com orderer.example.com pe
er0.org1.example.com couchdb
Creating network "net_basic" with the default driver
Creating orderer.example.com ...
Creating ca.example.com ...
Creating couchdb ...
Creating ca.example.com
Creating orderer.example.com
Creating couchdb ... done
```


collin@collin: ~/fabric-samples/supply
File Edit View Search Terminal Help

```
Creating cli ... done
2019-02-12 06:36:09.316 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 U
sing default escc
2019-02-12 06:36:09.316 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default vscc
2019-02-12 06:36:09.527 UTC [chaincodeCmd] install -> INFO 003 Installed remotel
y response:<status:200 payload:"OK" >
2019-02-12 06:36:09.719 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 U
sing default escc
2019-02-12 06:36:09.719 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default vscc
2019-02-12 06:36:21.792 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Ch
aincode invoke successful. result: status:200

Total setup execution time : 45 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

collin@collin:~/fabric-samples/supply$ 
```

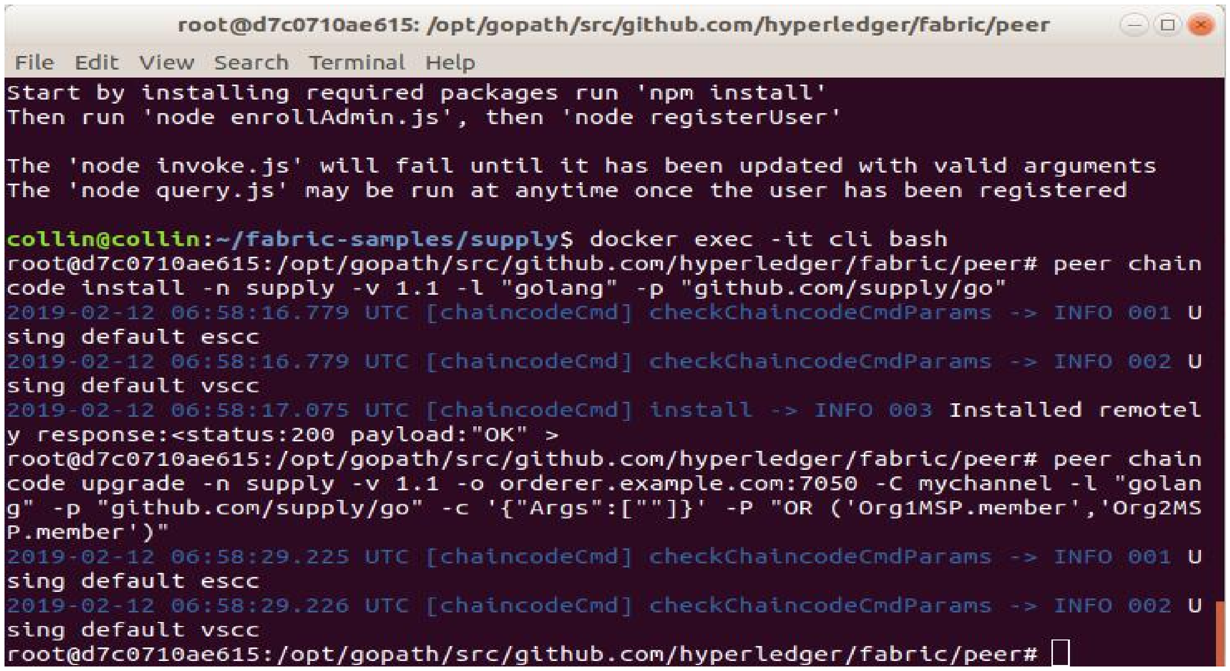
Step 4: Creating the peers we need to type following commands:

docker exec –it cli bash Press Enter.

peer chaincode install -n supply -v 1.1 -l "golang" -p "github.com/supply/go" (then press enter)

peer chaincode upgrade -n supply -v 1.1 -o orderer.example.com:7050 -C mychannel -l "golang" -p "github.com/supply/go" -c '{"Args":[""]}' -P "OR ('Org1MSP.member','Org2MSP.member')"

Then exit.


root@d7c0710ae615: /opt/gopath/src/github.com/hyperledger/fabric/peer
File Edit View Search Terminal Help

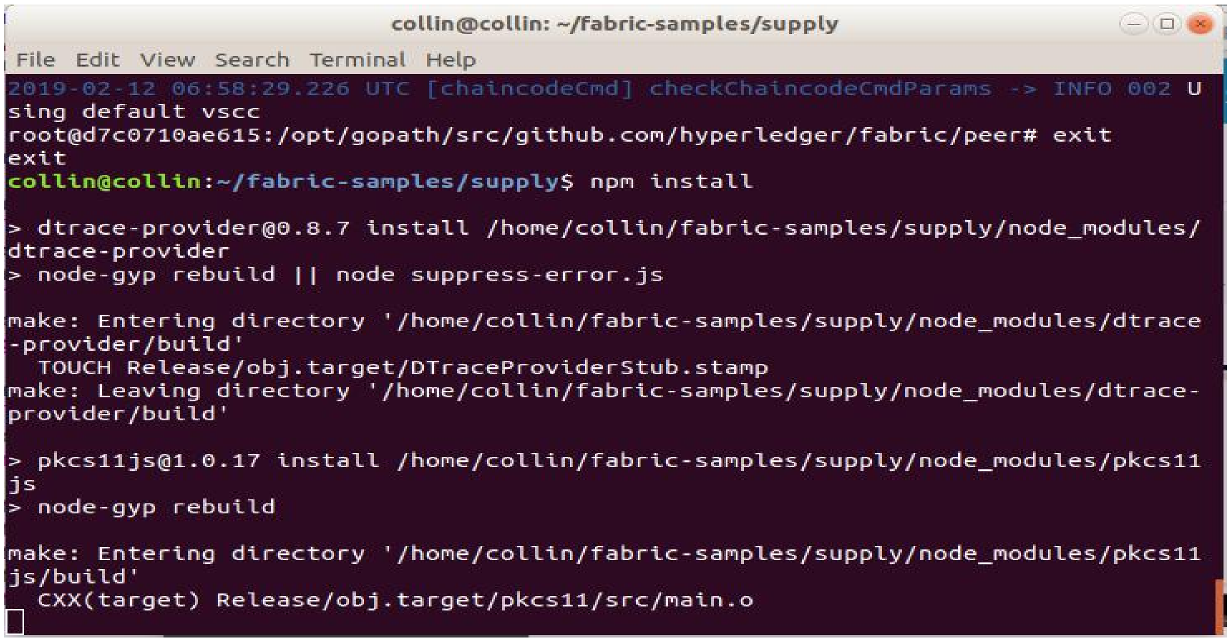
```
Start by installing required packages run 'npm install'
Then run 'node enrollAdmin.js', then 'node registerUser'

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

collin@collin:~/fabric-samples/supply$ docker exec -it cli bash
root@d7c0710ae615:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chain
code install -n supply -v 1.1 -l "golang" -p "github.com/supply/go"
2019-02-12 06:58:16.779 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 U
sing default escc
2019-02-12 06:58:16.779 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default vscc
2019-02-12 06:58:17.075 UTC [chaincodeCmd] install -> INFO 003 Installed remotel
y response:<status:200 payload:"OK" >
root@d7c0710ae615:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chain
code upgrade -n supply -v 1.1 -o orderer.example.com:7050 -C mychannel -l "golan
g" -p "github.com/supply/go" -c '{"Args":[""]}' -P "OR ('Org1MSP.member','Org2MS
P.member')"
2019-02-12 06:58:29.225 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 U
sing default escc
2019-02-12 06:58:29.226 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default vscc
root@d7c0710ae615:/opt/gopath/src/github.com/hyperledger/fabric/peer# 
```

Step 5: Type ***npm instal*l**

The purpose of npm install is to install the list of dependencies


collin@collin: ~/fabric-samples/supply
File Edit View Search Terminal Help

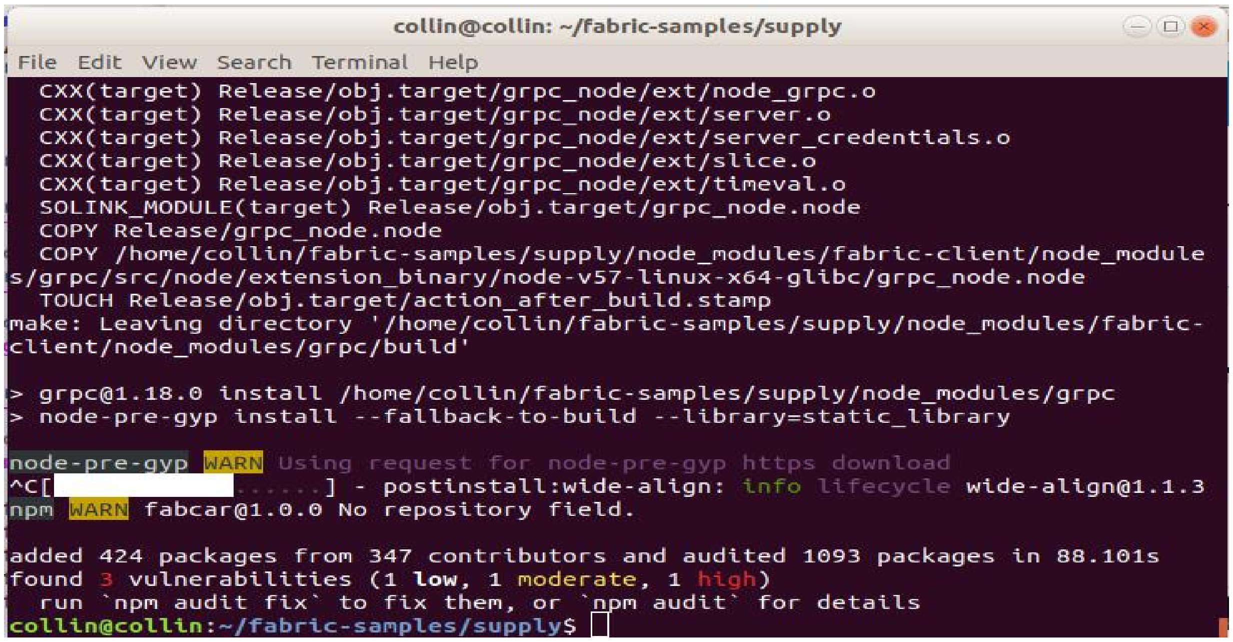
```
2019-02-12 06:58:29.226 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default vscc
root@d7c0710ae615:/opt/gopath/src/github.com/hyperledger/fabric/peer# exit
exit
collin@collin:~/fabric-samples/supply$ npm install

> dtrace-provider@0.8.7 install /home/collin/fabric-samples/supply/node_modules/
dtrace-provider
> node-gyp rebuild || node suppress-error.js

make: Entering directory '/home/collin/fabric-samples/supply/node_modules/dtrace
-provider/build'
  TOUCH Release/obj.target/DTraceProviderStub.stamp
make: Leaving directory '/home/collin/fabric-samples/supply/node_modules/dtrace-
provider/build'

> pkcs11js@1.0.17 install /home/collin/fabric-samples/supply/node_modules/pkcs11
js
> node-gyp rebuild

make: Entering directory '/home/collin/fabric-samples/supply/node_modules/pkcs11
js/build'
  CXX(target) Release/obj.target/pkcs11/src/main.o
```


collin@collin: ~/fabric-samples/supply
File Edit View Search Terminal Help

```
  CXX(target) Release/obj.target/grpc_node/ext/node_grpc.o
  CXX(target) Release/obj.target/grpc_node/ext/server.o
  CXX(target) Release/obj.target/grpc_node/ext/server_credentials.o
  CXX(target) Release/obj.target/grpc_node/ext/slice.o
  CXX(target) Release/obj.target/grpc_node/ext/timeval.o
  SOLINK_MODULE(target) Release/obj.target/grpc_node.node
  COPY Release/grpc_node.node
  COPY /home/collin/fabric-samples/supply/node_modules/fabric-client/node_module
s/grpc/src/node/extension_binary/node-v57-linux-x64-glibc/grpc_node.node
  TOUCH Release/obj.target/action_after_build.stamp
make: Leaving directory '/home/collin/fabric-samples/supply/node_modules/fabric-
client/node_modules/grpc/build'

> grpc@1.18.0 install /home/collin/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
^C[            ......] - postinstall:wide-align: info lifecycle wide-align@1.1.3
npm WARN fabcar@1.0.0 No repository field.

added 424 packages from 347 contributors and audited 1093 packages in 88.101s
found 3 vulnerabilities (1 low, 1 moderate, 1 high)
  run `npm audit fix` to fix them, or `npm audit` for details
collin@collin:~/fabric-samples/supply$
```

Step 6: Type the following commands;

node enrollAdmin.js (creates an admin for the network)

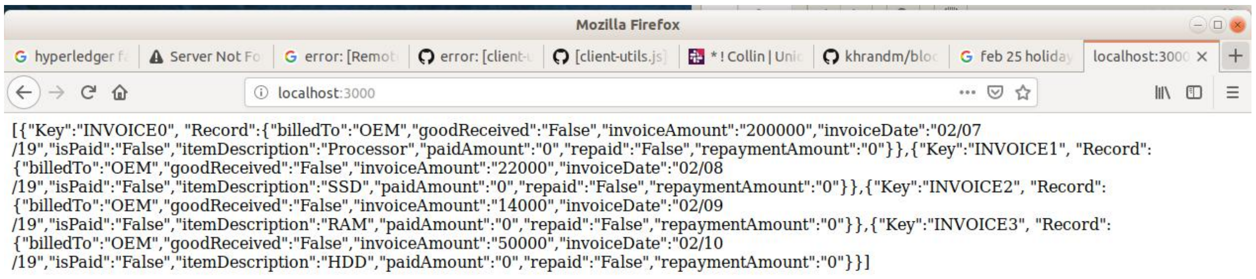
node registerSupplier.js (creates supplier for the network)

node registerOEM.js (creates OEM for the network)

node registerBank.js (creates bank for the network

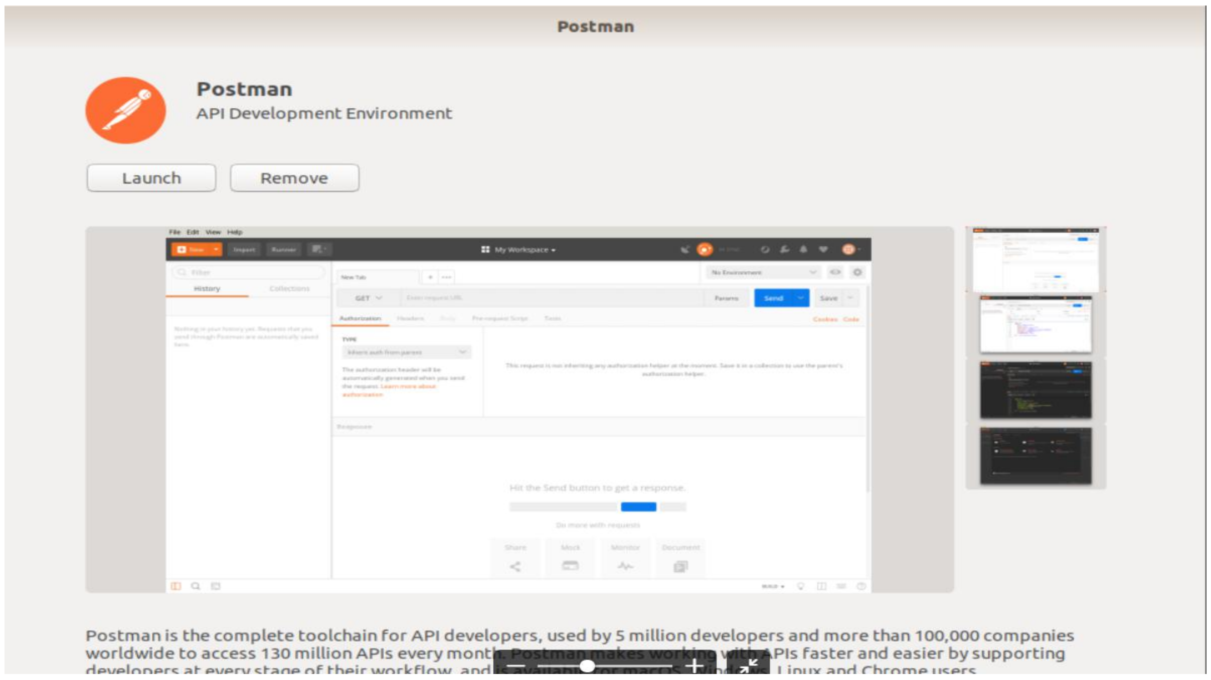
node app.js (run the application)

Step 7: Testing Endpoints
Type "**http://localhost:3000/**" in your browser it will display something like this.



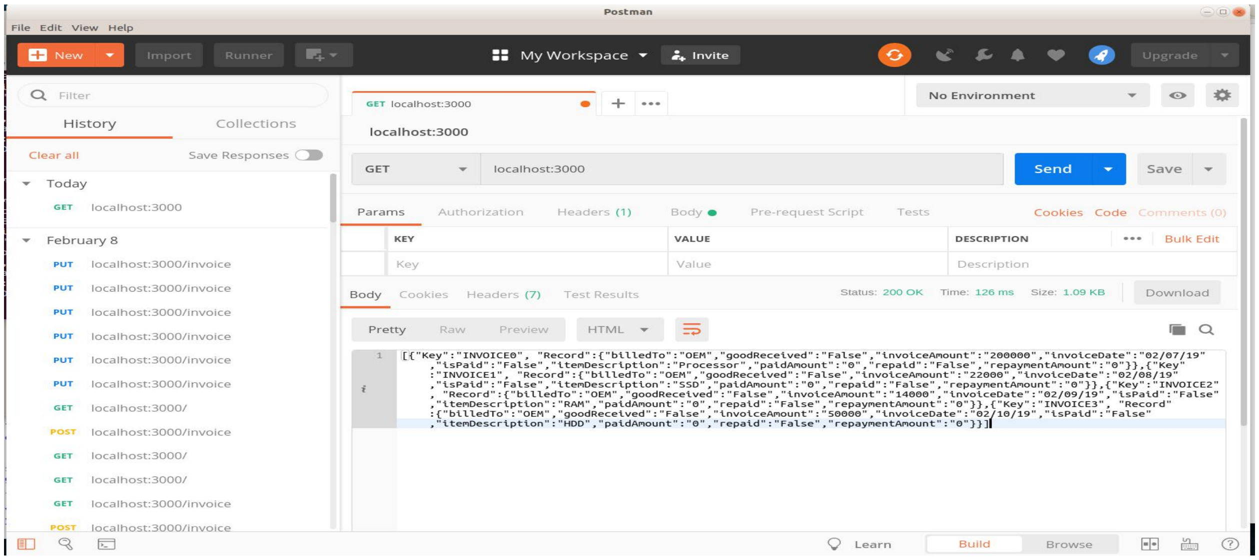
```
[{"Key":"INVOICE0", "Record":{"billedTo":"OEM","goodReceived":"False","invoiceAmount":"200000","invoiceDate":"02/07
/19","isPaid":"False","itemDescription":"Processor","paidAmount":"0","repaid":"False","repaymentAmount":"0"}},{"Key":"INVOICE1", "Record":
{"billedTo":"OEM","goodReceived":"False","invoiceAmount":"22000","invoiceDate":"02/08
/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"}}]
```

Step 8: In the Ubuntu software search and download **Postman**


Postman
Postman
API Development Environment
Launch
Remove
Hit the Send button to get a response.
Share
Mock
Monitor
Document

Postman is the complete toolchain for API developers, used by 5 million developers and more than 100,000 companies worldwide to access 130 million APIs every month. Postman makes working with APIs faster and easier by supporting developers at every stage of their workflow, and is available for macOS, Windows, Linux and Chrome users

Step 9: In your Postman
Use the GET HTTP Method in this function as we are retrieving the data. Click Send.



GET localhost:3000

Send

Save

Status: 200 OK Time: 126 ms Size: 1.09 KB

HTML

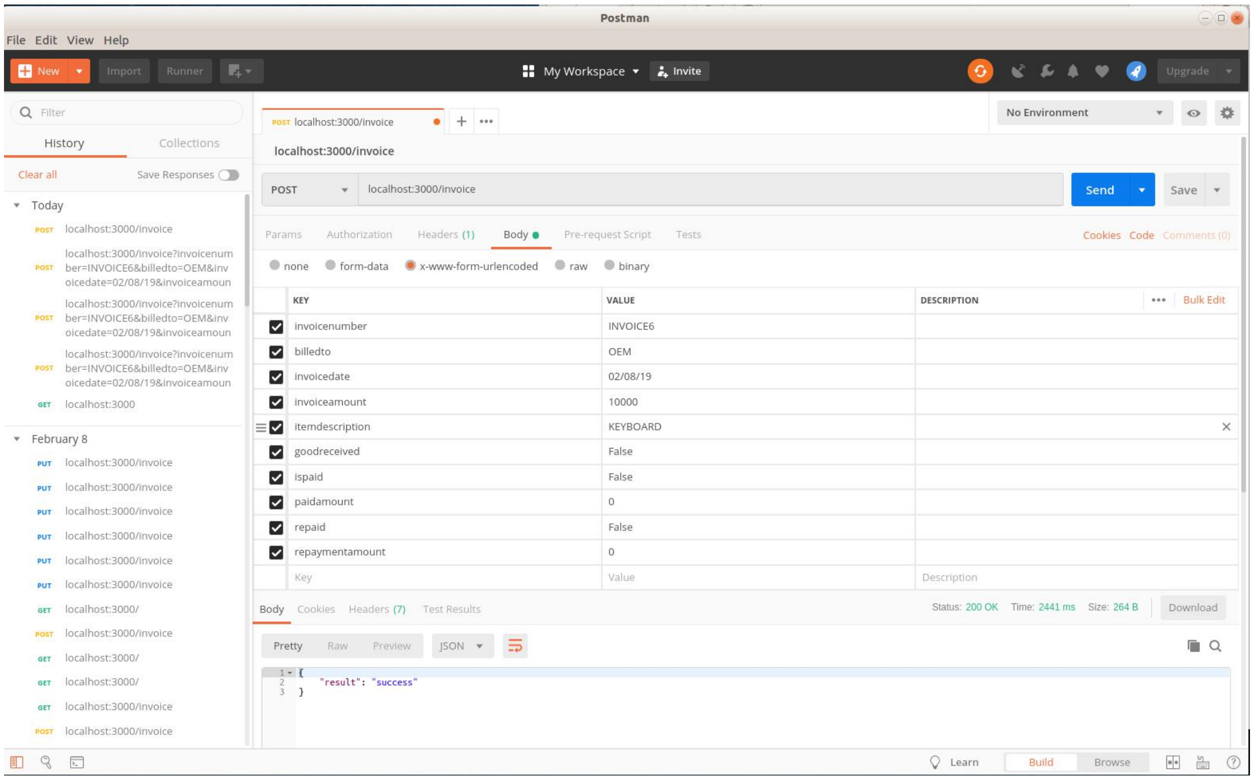
```
[{"Key":"INVOICE0", "Record":{"billedTo":"OEM","goodReceived":"False","invoiceAmount":"200000","invoiceDate":"02/07/19"
    ,"isPaid":"False","itemDescription":"Processor","paidAmount":"0","repaid":"False","repaymentAmount":"0"}},{"Key"
    :"INVOICE1", "Record":{"billedTo":"OEM","goodReceived":"False","invoiceAmount":"22000","invoiceDate":"02/08/19"
    ,"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"}}]
```

Step 10: Use the Post HTTP Method. In this function it request that a web server accepts the data enclosed in the body of the request message, most likely for storing it.Add url **localhost:3000/invoice**
Below url you should see Params Authroization Headers Body, click Headers
Add another key below Content-Type
Type **user** and the value would be **"supplier"**
Next open the body tab, click the **x-www-form-url-encoded**
Click Send.

For example we will use these following fields and their specific value.

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

The result below will look like these:



Today
- POST localhost:3000/invoice
- POST localhost:3000/invoice?invoicenumber=INVOICE6&billedto=OEM&invoicedate=02/08/19&invoiceamoun
- POST localhost:3000/invoice?invoicenumber=INVOICE6&billedto=OEM&invoicedate=02/08/19&invoiceamoun
- POST localhost:3000/invoice?invoicenumber=INVOICE6&billedto=OEM&invoicedate=02/08/19&invoiceamoun
- GET localhost:3000

February 8
- PUT localhost:3000/invoice
- PUT localhost:3000/invoice
- PUT localhost:3000/invoice
- PUT localhost:3000/invoice
- PUT localhost:3000/invoice
- PUT localhost:3000/invoice
- GET localhost:3000/
- POST localhost:3000/invoice
- GET localhost:3000/
- GET localhost:3000/
- GET localhost:3000/invoice
- POST localhost:3000/invoice

localhost:3000/invoice

POST localhost:3000/invoice

Body: ( ) none ( ) form-data (●) x-www-form-urlencoded ( ) raw ( ) binary

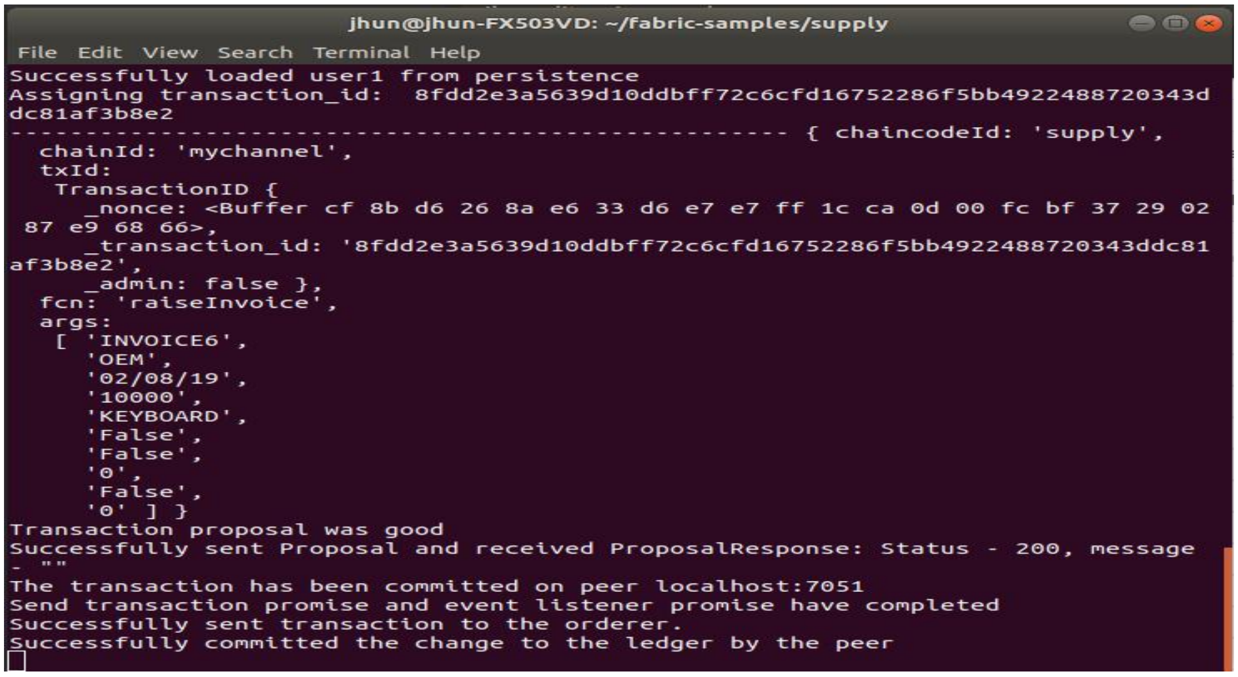
	KEY	VALUE	DESCRIPTION
✓	invoicenumber	INVOICE6	
✓	billedto	OEM	
✓	invoicedate	02/08/19	
✓	invoiceamount	10000	
✓	itemdescription	KEYBOARD	
✓	goodreceived	False	
✓	ispaid	False	
✓	paidamount	0	
✓	repaid	False	
✓	repaymentamount	0	
	Key	Value	Description

Body Cookies Headers (7) Test Results — Status: 200 OK Time: 2441 ms Size: 264 B Download

Pretty Raw Preview JSON

```
{
    "result": "success"
}
```

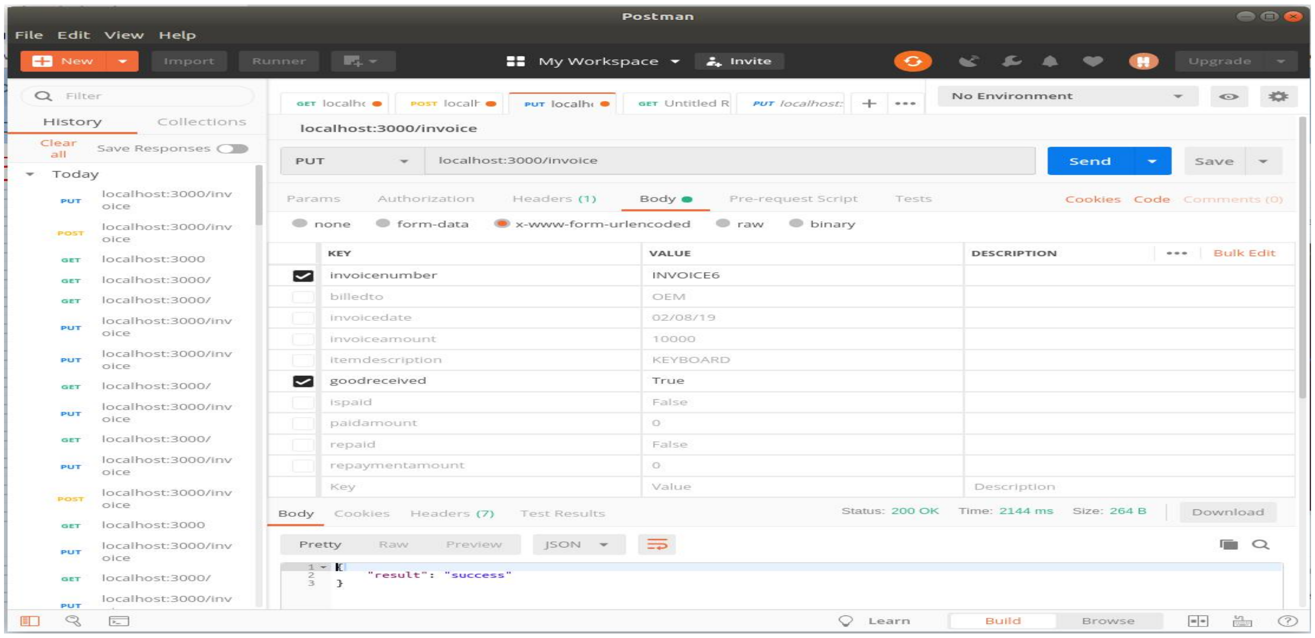
And you will see an ouput in your terminal like these:


jhun@jhun-FX503VD: ~/fabric-samples/supply
File Edit View Search Terminal Help

```
Successfully loaded user1 from persistence
Assigning transaction_id:  8fdd2e3a5639d10ddbff72c6cfd16752286f5bb4922488720343d
dc81af3b8e2
--------------------------------------------------- { chaincodeId: 'supply',
  chainId: 'mychannel',
  txId:
   TransactionID {
     _nonce: <Buffer cf 8b d6 26 8a e6 33 d6 e7 e7 ff 1c ca 0d 00 fc bf 37 29 02
 87 e9 68 66>,
     _transaction_id: '8fdd2e3a5639d10ddbff72c6cfd16752286f5bb4922488720343ddc81
af3b8e2',
     _admin: false },
  fcn: 'raiseInvoice',
  args:
   [ 'INVOICE6',
     'OEM',
     '02/08/19',
     '10000',
     'KEYBOARD',
     'False',
     'False',
     '0',
     'False',
     '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
```

The transaction has been sent in your peer. A new invoice has been created!

Step 12: Use the PUT HTTP Method.In this function as we are modifying a data.
Go to header tab and add **user** with value of "**oem"**
Next go to body **x-www-form-urlencoded,** click Send.



PUT localhost:3000/invoice

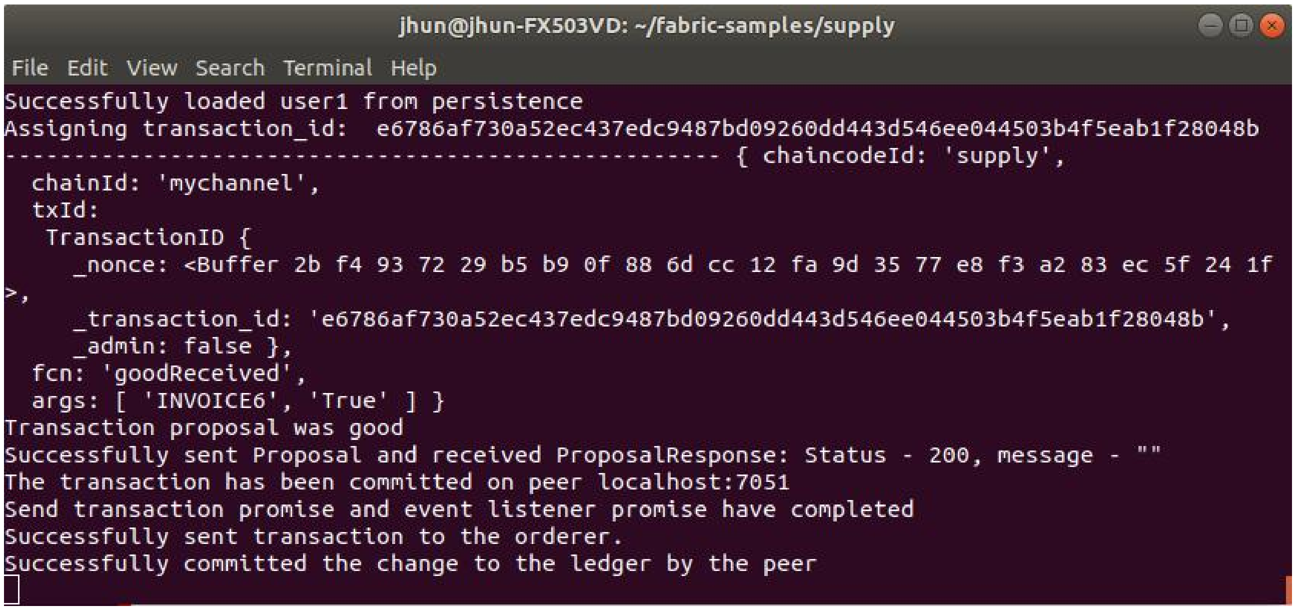
Body: x-www-form-urlencoded

	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	

Status: 200 OK Time: 2144 ms Size: 264 B

```
{
    "result": "success"
}
```

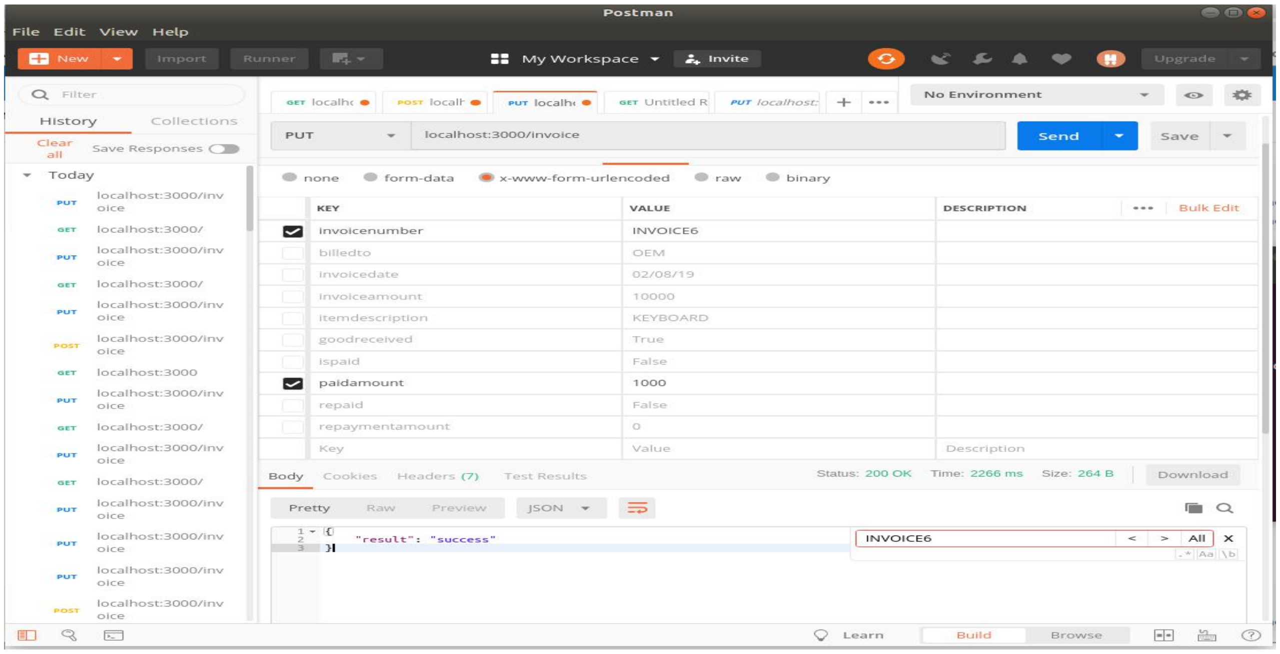
And you will see an ouput in your terminal like these:


jhun@jhun-FX503VD: ~/fabric-samples/supply
File Edit View Search Terminal Help

```
Successfully loaded user1 from persistence
Assigning transaction_id:  e6786af730a52ec437edc9487bd09260dd443d546ee044503b4f5eab1f28048b
--------------------------------------------------- { chaincodeId: 'supply',
  chainId: 'mychannel',
  txId:
   TransactionID {
     _nonce: <Buffer 2b f4 93 72 29 b5 b9 0f 88 6d cc 12 fa 9d 35 77 e8 f3 a2 83 ec 5f 24 1f
>,
     _transaction_id: 'e6786af730a52ec437edc9487bd09260dd443d546ee044503b4f5eab1f28048b',
     _admin: false },
  fcn: 'goodReceived',
  args: [ 'INVOICE6', 'True' ] }
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
```

Step 13: Use the PUT HTTP Method.In this function as we are modifying a data.
Go to header tab and add **user** with value of **“bank”**
Next go to body **x-www-form-urlencoded,** click Send.

Unchecked all the fields except to **invoicenumber** and **paidamount.** Then click send and it should like these.



PUT localhost:3000/invoice

none · form-data · [✓ x-www-form-urlencoded] · raw · binary

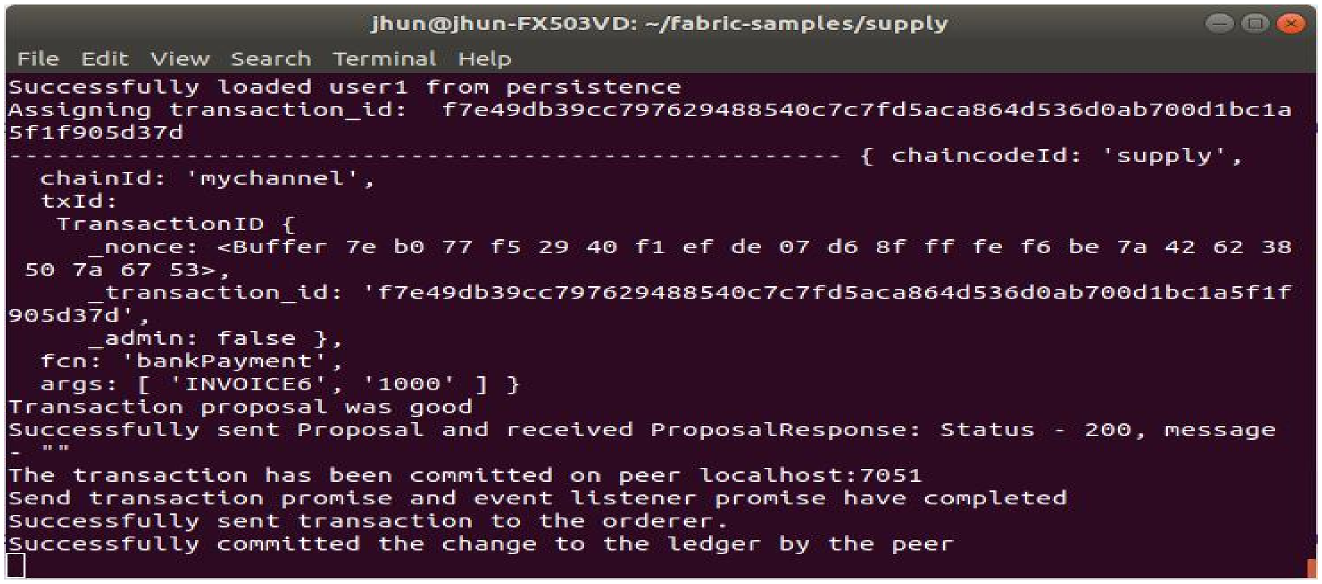
	KEY	VALUE	DESCRIPTION
✓	invoicenumber	INVOICE6	
	billedto	OEM	
	invoicedate	02/08/19	
	invoiceamount	10000	
	itemdescription	KEYBOARD	
	goodreceived	True	
	ispaid	False	
✓	paidamount	1000	
	repaid	False	
	repaymentamount	0	
	Key	Value	Description

Body Cookies Headers (7) Test Results — Status: 200 OK Time: 2266 ms Size: 264 B

Pretty Raw Preview JSON

```
{
    "result": "success"
}
```

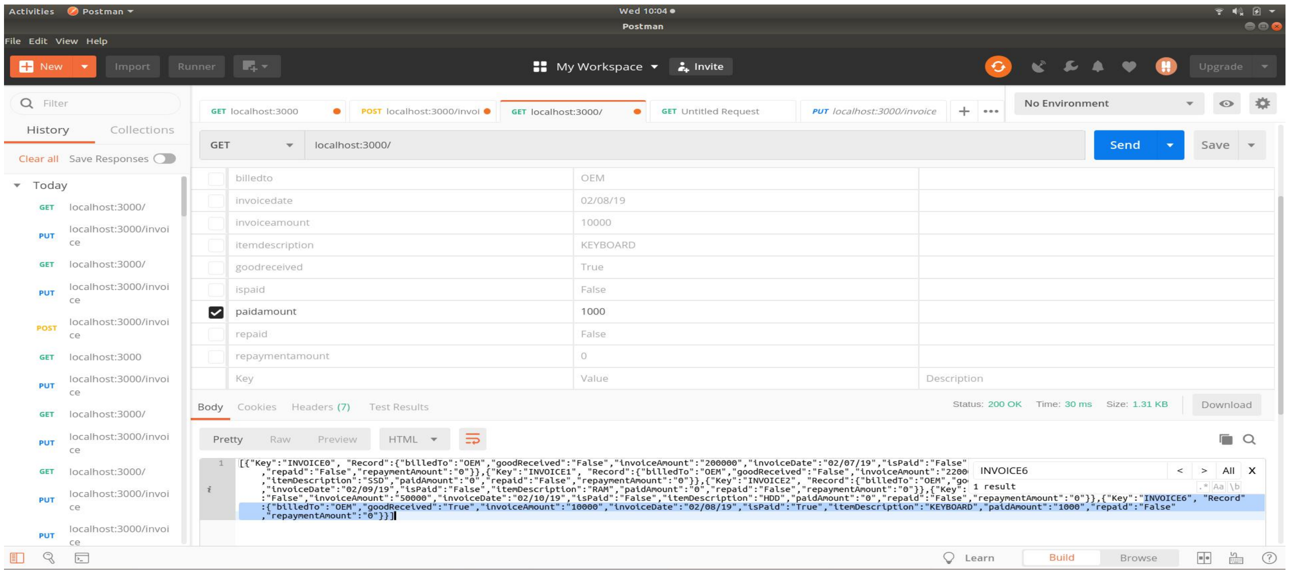
And you will see an ouput in your terminal like these:


jhun@jhun-FX503VD: ~/fabric-samples/supply
File Edit View Search Terminal Help

```
Successfully loaded user1 from persistence
Assigning transaction_id:  f7e49db39cc797629488540c7c7fd5aca864d536d0ab700d1bc1a
5f1f905d37d
---------------------------------------------------- { chaincodeId: 'supply',
  chainId: 'mychannel',
  txId:
   TransactionID {
     _nonce: <Buffer 7e b0 77 f5 29 40 f1 ef de 07 d6 8f ff fe f6 be 7a 42 62 38
 50 7a 67 53>,
     _transaction_id: 'f7e49db39cc797629488540c7c7fd5aca864d536d0ab700d1bc1a5f1f
905d37d',
     _admin: false },
  fcn: 'bankPayment',
  args: [ 'INVOICE6', '1000' ] }
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
```

In your postman the result will be something like these:

Use the GET HTTP Method then type "localhost:3000" and add value to the **paidamount** key. Click Send. Here is the example.



GET localhost:3000/

☐	billedto	OEM
☐	invoicedate	02/08/19
☐	invoiceamount	10000
☐	itemdescription	KEYBOARD
☐	goodreceived	True
☐	ispaid	False
✓	paidamount	1000
☐	repaid	False
☐	repaymentamount	0

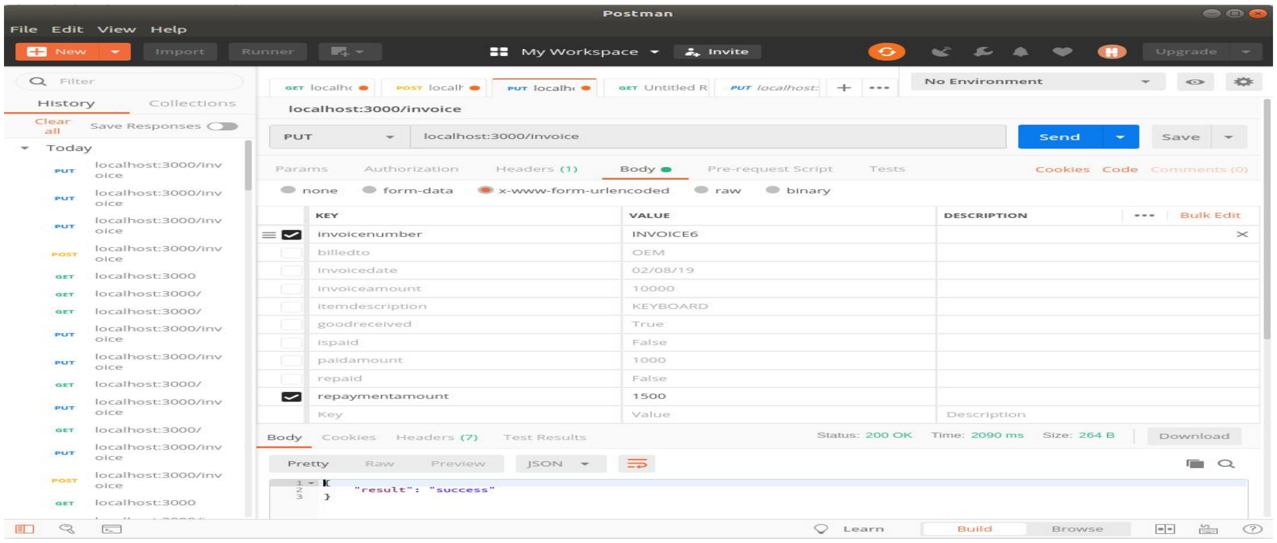
Status: 200 OK Time: 30 ms Size: 1.31 KB

```
[{"Key":"INVOICE0", "Record":{"billedTo":"OEM","goodReceived":"False","invoiceAmount":"200000","invoiceDate":"02/07/19","isPaid":"False"
    ,"repaid":"False","repaymentAmount":"0"}},{"Key":"INVOICE1", "Record":{"billedTo":"OEM","goodReceived":"False","invoiceAmount":"2200
    ,"itemDescription":"SSD","paidAmount":"0","repaid":"False","repaymentAmount":"0"}},{"Key":"INVOICE2", "Record":{"billedTo":"OEM","go
    ,"invoiceDate":"02/09/19","isPaid":"False","itemDescription":"RAM","paidAmount":"0","repaid":"False","repaymentAmount":"0"}},{"Key":
    :"False","invoiceAmount":"50000","invoiceDate":"02/10/19","isPaid":"False","itemDescription":"HDD","paidAmount":"0","repaid":"False","repaymentAmount":"0"}},{"Key":"INVOICE6", "Record"
    :{"billedTo":"OEM","goodReceived":"True","invoiceAmount":"10000","invoiceDate":"02/08/19","isPaid":"True","itemDescription":"KEYBOARD","paidAmount":"1000","repaid":"False"
    ,"repaymentAmount":"0"}}]
```

As you can see in the results below the value of the field **isPaid** have changed from "False" to "True" it means that the product has been paid.

Step 14: Use the PUT HTTP Method.In this function as we are modifying a data.
Go to header tab and add **user** with value of "**oem"**
Next go to body **x-www-form-urlencoded,** click Send.

Unchecked all the fields except to **invoicenumber** and **repaymentamount.** Then click send and it should like these.



PUT localhost:3000/invoice

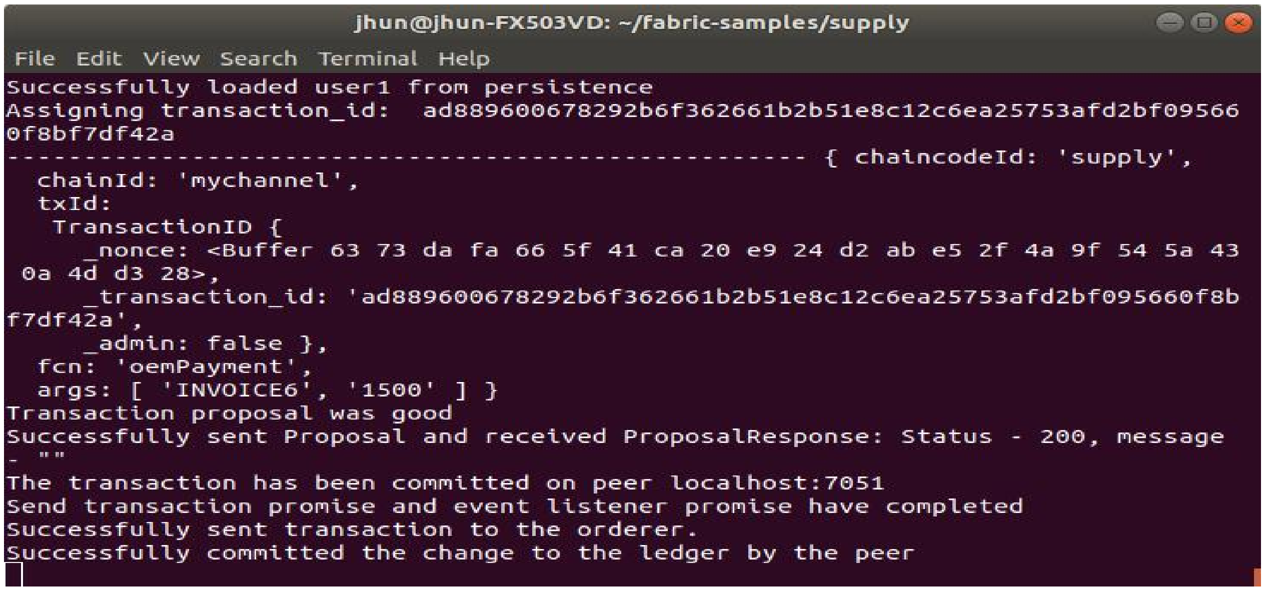
Body: x-www-form-urlencoded

	KEY	VALUE	DESCRIPTION
✓	invoicenumber	INVOICE6	
☐	billedto	OEM	
☐	invoicedate	02/08/19	
☐	invoiceamount	10000	
☐	itemdescription	KEYBOARD	
☐	goodreceived	True	
☐	ispaid	False	
☐	paidamount	1000	
☐	repaid	False	
✓	repaymentamount	1500	

Status: 200 OK Time: 2090 ms Size: 264 B

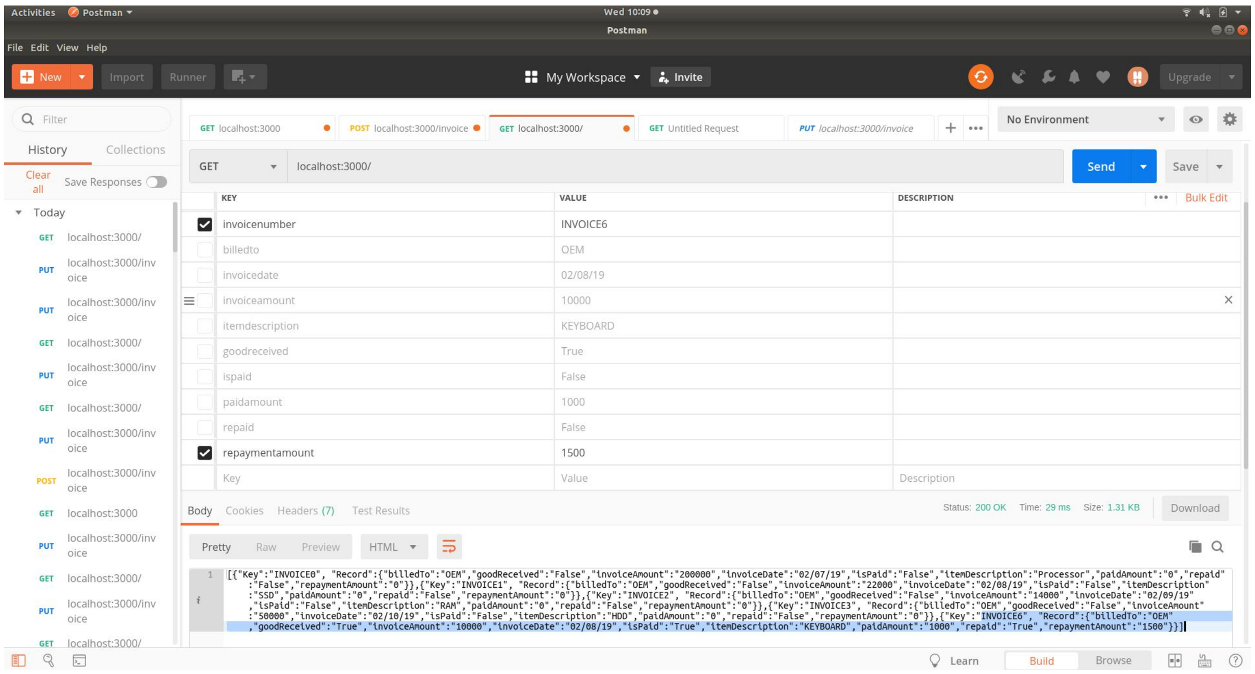
```
{
    "result": "success"
}
```

And you will see an ouput in your terminal like these:



```
Successfully loaded user1 from persistence
Assigning transaction_id:  ad889600678292b6f362661b2b51e8c12c6ea25753afd2bf09566
0f8bf7df42a
---------------------------------------------------- { chaincodeId: 'supply',
  chainId: 'mychannel',
  txId:
   TransactionID {
     _nonce: <Buffer 63 73 da fa 66 5f 41 ca 20 e9 24 d2 ab e5 2f 4a 9f 54 5a 43
 0a 4d d3 28>,
     _transaction_id: 'ad889600678292b6f362661b2b51e8c12c6ea25753afd2bf095660f8b
f7df42a',
     _admin: false },
  fcn: 'oemPayment',
  args: [ 'INVOICE6', '1500' ] }
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
```

In your postman the result will be something like these:
Use the GET HTTP Method then type "localhost:3000" and add value to the **repayment** key. Click Send.
Here is the example.



GET localhost:3000/

	KEY	VALUE	DESCRIPTION
✓	invoicenumber	INVOICE6	
	billedto	OEM	
	invoicedate	02/08/19	
	invoiceamount	10000	
	itemdescription	KEYBOARD	
	goodreceived	True	
	ispaid	False	
	paidamount	1000	
	repaid	False	
✓	repaymentamount	1500	

Status: 200 OK Time: 29 ms Size: 1.31 KB

```
[{"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":"INVOICE6", "Record":{"billedTo":"OEM"
    ,"goodReceived":"True","invoiceAmount":"10000","invoiceDate":"02/08/19","isPaid":"True","itemDescription":"KEYBOARD","paidAmount":"1000","repaid":"True","repaymentAmount":"1500"}}]
```

You can see in the results below the value of the field **repaid** have changed from "False" to "True".