

ELECTRONICS SUPPLY CHAIN MANAGEMENT

- **About The Project**

This Project is about maintaining the transparency between customer and E-com websites to ensure same product delivered to the end user or not by maintaining the ledger of Appliance or product data between Manufacturers and E-com websites and Delivery partners

This Project will be useful for tracking appliance or product life cycle between all participant to avoid gap between E-com websites and customers

- **Project Flow**

Manufacturer Will Release the product and add barcode to the product will contain below information of product

- * Product Name
- * Product Model Number
- *MRP
- * Date Of Manufacturing
- * Manufacturer Name

The information added by the manufacturer gets stored on the block chain, providing transparency to the supply chain to other stakeholders. Manufacturer will send product to E-commerce company with the help of Hash ID stored on the block chain, E-Commerce company can verify the origin of Product after collecting them from the logistics service Providers E-Commerce Company validate the received Product and sign the transaction digitally, which is then added to the block chain. E-Commerce Company can get the product details like product type, model number in block

When Customer orders the product through E-Commerce website based on product details the order has to handover to Product Order logistics company for delivery with order and product details

- * E-commerce can get the Product/Order details which can be traced back to know its origin, using the hash ID saved on the block chain

* Suppose any illegal product delivery happens and “product order” company to deliver counterfeit product with fake Order ID to Customer. In that case, the transaction is considered invalid because of the false information added to the Order.

* Also, unauthorized individuals cannot carry out transactions in the Electronics supply chain ecosystem without a valid private key

* Customer can validate the product received by scan the bar code received in product cover with E-commerce app

* Customer can ensure ordered product and delivered product both are matching or not

* The hash ID linked to the Barcode would fetch information from the blockchain for product quality to avoid false product E-commerce companies, in turn, send approved products to manufacturers.

* After that, the finished goods are delivered to a central warehouse, separated into packages, and distributed to local warehouses

* The repackaged Products will be handled by logistics to deliver to Ecommerce warehouses in different cities, based on the Customer order E commerce company will hand over product to “Product orders”

* Product order company will deliver the product to customer, customer/Ecommerce and track product based on deliver details and product details in website

- **Project Contents**

This Project will contain the 3 peers and Orderer will connect through a secured network layer with Fabric-ca certificate authentication

Step1: Up the network and enroll the peers and orderer with secured communication by generating the ca TLS signer certificates to exchange secure connection and data exchange

Step2: Deploy the chain code by invoking the business data

Step3 : insert product details and query from other peers

Step4: invoke the different operations of data

```
1 #!/bin/bash
2 echo "-----Register the ca admin for each organization-----"
3 docker-compose -f docker/docker-compose-ca.yaml up -d
4 sleep 3
5 echo "-----Register and enroll the users for each organization-----"
6 sudo chmod -R 777 organizations/
7 echo "-----"
8 cd /opt
9 cd /opt
10 cd /opt
11 cd /opt
12 cd /opt
13 cd /opt
14 cd /opt
15 cd /opt
16 cd /opt
17 cd /opt
18 cd /opt
19 cd /opt
20 cd /opt
21 cd /opt
22 cd /opt
23 cd /opt
24 cd /opt
25 cd /opt
26 cd /opt
27 cd /opt
28 cd /opt
29 cd /opt
30 cd /opt
31 cd /opt
32 cd /opt
33 cd /opt
34 cd /opt
35 cd /opt
36 cd /opt
37 cd /opt
38 cd /opt
39 cd /opt
40 cd /opt
41 cd /opt
42 cd /opt
43 cd /opt
44 cd /opt
45 cd /opt
46 cd /opt
47 cd /opt
48 cd /opt
49 cd /opt
50 cd /opt
51 cd /opt
52 cd /opt
53 cd /opt
54 cd /opt
55 cd /opt
56 cd /opt
57 cd /opt
58 cd /opt
59 cd /opt
60 cd /opt
61 cd /opt
62 cd /opt
63 cd /opt
64 cd /opt
65 cd /opt
66 cd /opt
67 cd /opt
68 cd /opt
69 cd /opt
70 cd /opt
71 cd /opt
72 cd /opt
73 cd /opt
74 cd /opt
75 cd /opt
76 cd /opt
77 cd /opt
78 cd /opt
79 cd /opt
80 cd /opt
81 cd /opt
82 cd /opt
83 cd /opt
84 cd /opt
85 cd /opt
86 cd /opt
87 cd /opt
88 cd /opt
89 cd /opt
90 cd /opt
91 cd /opt
92 cd /opt
93 cd /opt
94 cd /opt
95 cd /opt
96 cd /opt
97 cd /opt
98 cd /opt
99 cd /opt
100 cd /opt
```

Step5: Create the Client app to enroll the data to register the product

← → ↺

0.0.0.0:3000

☆

🔍 ⬇️ 🏠 ☰

Auto App

Create Appliance

ApplianceId	<input type="text" value="LG1234"/>
Make	<input type="text" value="LG"/>
Model	<input type="text" value="LG TV X545 LM"/>
Color	<input type="text" value="BLACK"/>
Date of Manufacture	<input type="text" value="01 / 11 / 2023"/>
Manufacturer Name	<input type="text" value="LG INDIA PVT LTD"/>

Query Appliance

Enter ApplianceId

Step6: we can perform and verify the data exchange between client and server

```

FWXRXRX-X 2 npcl-admin npcl-admin 4096 Nov 21 15:34 Client/
FW-RW-R-- 1 npcl-admin npcl-admin 18330 Nov 21 10:38 commandsElectronicsCommands.txt
FW-RW-R-- 1 npcl-admin npcl-admin 431 Nov 10 11:56 Commands.txt
FW-RW-R-- 1 npcl-admin npcl-admin 4966 Nov 23 09:11 CreateOrderDetails
FWXRXRX-X 7 npcl-admin npcl-admin 4096 Nov 22 09:40 Electronics-network/
FWXRXRX-X 2 npcl-admin npcl-admin 4096 Nov 10 11:56 Events/
FWXRXRX-X 3 npcl-admin npcl-admin 4096 Nov 10 11:56 Gin-App/
FW-RW-R-- 1 npcl-admin npcl-admin 256464 Nov 10 11:56 Project Document_Template.pdf
FW-RW-R-- 1 npcl-admin npcl-admin 3290 Nov 10 11:56 ProjectGuidelines.md
FWXRXRX-X 4 npcl-admin npcl-admin 4096 Nov 10 11:56 Simple-Dapp/
pcl-admin@test121: /KBA-CHF/KBA-Electronics$ cd Simple-Dapp/
pcl-admin@test121: /KBA-CHF/KBA-Electronics/Simple-Dapp$ go run .
GIN-debug [WARNING] Creating an Engine instance with the Logger and Recovery middleware already attached.

GIN-debug [WARNING] Running in "debug" mode. Switch to "release" mode in production.
- using env: export GIN_MODE=release
- using code: gin.SetMode(gin.ReleaseMode)

GIN-debug GET /public/*filepath --> github.com/gin-gonic/gin.(*RouterGroup).createStaticHandler.func1 (3 handlers)
GIN-debug HEAD /public/*filepath --> github.com/gin-gonic/gin.(*RouterGroup).createStaticHandler.func1 (3 handlers)
GIN-debug Loaded HTML Templates (2):
- index.html

GIN-debug GET / --> main.main.func1 (3 handlers)
GIN-debug POST /api/appliance --> main.main.func2 (3 handlers)
GIN-debug GET /api/appliance/:id --> main.main.func3 (3 handlers)
GIN-debug GET /get-event --> main.main.func4 (3 handlers)
GIN-debug [WARNING] You trusted all proxies, this is NOT safe. We recommend you to set a value.
lease check https://pkg.go.dev/github.com/gin-gonic/gin#readme-don-t-trust-all-proxies for details.
GIN-debug Listening and serving HTTP on 0.0.0.0:3000

** Start Chaincode event listening
GIN] 2023/11/23 - 09:52:51 | 200 | 249.201µs | 127.0.0.1 | GET | "/"
GIN] 2023/11/23 - 09:52:51 | 400 | 708.161µs | 127.0.0.1 | GET | "/public/styles/style.css"
GIN] 2023/11/23 - 09:52:51 | 304 | 124.085µs | 127.0.0.1 | GET | "/public/scripts/index.js"
GIN] 2023/11/23 - 09:52:51 | 400 | 623ns | 127.0.0.1 | GET | "/favicon.ico"

```

```

** Start Chaincode event listening
received Chaincode Event: CreateAppliance
Data: {"Type": "Appliance creation", "Model": "Refrigrator"}

received Chaincode Event: CreateAppliance
Data: {"Type": "Appliance creation", "Model": "SS121"}

Csignal: Interrupt
pcl-admin@test121: /KBA-CHF/KBA-Electronics/Events$ go run .

** Start Chaincode event listening
received Chaincode Event: CreateAppliance
Data: {"Type": "Appliance creation", "Model": "ss1"}

received Chaincode Event: CreateAppliance
Data: {"Type": "Appliance creation", "Model": "LG1"}

received Chaincode Event: CreateAppliance
Data: {"Type": "Appliance creation", "Model": "HR1"}

received Chaincode Event: CreateAppliance
Data: {"Type": "Appliance creation", "Model": "One1"}

Csignal: Interrupt
pcl-admin@test121: /KBA-CHF/KBA-Electronics/Events$ go run .

** Start Chaincode event listening
Csignal: Interrupt
pcl-admin@test121: /KBA-CHF/KBA-Electronics/Events$ go run .

** Start Chaincode event listening
received Chaincode Event: CreateAppliance
Data: {"Type": "Appliance creation", "Model": "JBL1"}

received Chaincode Event: CreateAppliance
Data: {"Type": "Appliance creation", "Model": "Refrigrator"}

```

We can invoke multiple operation to track the product data history and sort of data, pagination data and history and all order and product details

```

pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer
electronics.com --tls --cafile $ORDERER_CA -c $CHANNEL_NAME -n KBA-Electronics --peerAddresses localhost:7051 --tlsRootCertFiles $ORG1_PEER_T
LSROOTCERT --peerAddresses localhost:9051 --tlsRootCertFiles $ORG2_PEER_TLSROOTCERT --peerAddresses localhost:11051 --tlsRootCertFiles $ORG3_P
ER_TLSROOTCERT -c '{"function":"CreateAppliance","Args":["Electronic1","LG11","Refrigrator","Blue","HYDF-1","20/10/2019"]}'
2023-11-02 11:02:02.096 [INFO] peer http [chaincodeCmd] chaincodeInvokeorQuery -> Chaincode invoke successful. result: status:200 payload:"succes
Fully added appliance Electronic1"
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$ peer chaincode query -C autochannel -n KBA-Automobile -c '{"Args":["Applianc
Contract:GetApplianceHistory","LG11"]}'
error: endorsement failure during query. response: status:500 message:"channel 'autochannel' not found"
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$ peer chaincode query -C electronicschannel -n KBA-Electronics -c '{"Args":["
pplianceContract:GetApplianceHistory","LG11"]}'
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$ peer chaincode query -C electronicschannel -n KBA-Electronics -c '{"Args":["
pplianceContract:GetApplianceHistory","Electronic1"]}'
{"record":{"AssetType":"appliance","ApplianceId":"Electronic1","Color":"Blue","DateOfManufacture":"20/10/2019","OwnedBy":"HYDF-1","Make":"LG1
","Model":"Refrigrator","Status":"In Factory"},"txId":"5aabd18cbf3b971b29719240240f4a1cc99b19ed0af844572737eb300ab7e5e4","timestamp":"Wed, 2
Nov 2023 11:02:02 UTC","isDelete":false}}
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$ peer chaincode query -C electronicschannel -n KBA-Electronics -c '{"Args":["
pplianceContract:GetApplianceHistory","Electronic1"]}'
{"record":{"AssetType":"appliance","ApplianceId":"Electronic1","Color":"Blue","DateOfManufacture":"20/10/2019","OwnedBy":"HYDF-1","Make":"LG1
","Model":"Refrigrator","Status":"In Factory"},"txId":"5aabd18cbf3b971b29719240240f4a1cc99b19ed0af844572737eb300ab7e5e4","timestamp":"Wed, 2
Nov 2023 11:02:02 UTC","isDelete":false}}
pci-admin@test121:~/KBA-CHF/KBA-Electronics/Electronics-network$

```

- **Functions Used:**

Various functions used to insert and retrieve data based on organization requirement (we can find business contract data in contracts folder)

CreateOrder()

ReadOrder()

GetPrivateData()

DeleteOrder()

Range

GetStateByRange(startkey,endkey)

History

GetHistoryForKey(ID)

Pagination

GetQueryResultWithPagination(QueryString,pageSize,bookmark)

Matching Orders

GetPrivateDataQueryResult(collectionName,querystring)

Match order and appliances

GetPrivateData(collectionName,ID)