

**ĐẠI HỌC ĐÀ NẴNG  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CUỐI KÌ  
MẬT MÃ HỌC**

**ĐỀ TÀI:**

**Tìm hiểu Blockchain và ứng dụng xác thực tài sản  
trên nền tảng Hyperledger Fabric**

HỌ TÊN SINH VIÊN	MÃ SINH VIÊN	NHÓM
Đàm Quang Tiến	102180048	06
Phùng Văn Thắng	102180043	06
Đặng Xuân Minh Sơn	102180039	06

**Đà Nẵng, 06/2022**

# MỤC LỤC

1.1.	Tổng quan về đề tài .....	1
1.2.	Mục đích và ý nghĩa của đề tài.....	1
1.2.1.	Mục đích .....	1
1.2.2.	Ý nghĩa .....	2
1.3.	Phương pháp thực hiện.....	2
1.4.	Bố cục của báo cáo .....	2
<b>CHƯƠNG 2: CƠ SỞ LÝ THUYẾT .....</b>		<b>3</b>
2.1.	Hash Function.....	3
2.2.	Distributed Ledger.....	3
2.3.	Smart Contract.....	4
2.4.	Consensus .....	5
2.5.	Hyperledger fabric khác các mạng blockchain khác.....	5
<b>CHƯƠNG 3: PHÂN TÍCH THIẾT KẾ HỆ THỐNG .....</b>		<b>7</b>
3.1.	PHÁT BIỂU BÀI TOÁN .....	7
3.2.	PHÂN TÍCH HIỆN TRẠNG .....	7
3.3.	PHÂN TÍCH CHỨC NĂNG.....	7
3.3.1.	Đối tượng sử dụng .....	7
3.3.2.	Khai báo sở hữu.....	8
3.3.3.	Thay đổi chủ sở hữu .....	8
3.3.4.	Truy vấn chủ sở hữu của tài sản .....	9
3.3.5.	Hủy sở hữu tài sản .....	10
3.4.	THIẾT KẾ NETWORK .....	11
3.4.1.	Khởi tạo file crypto-config, configtx.yaml và tạo channel artifact .....	12
3.4.2.	Khởi tạo file docker-compose.yaml và tạo nên các peer của network .....	15
3.4.3.	Tạo channel và join các peer vào channel .....	16
3.4.4.	Cài đặt chaincode lên channel .....	17
3.5.	CÁC CHAINCODE .....	17
3.5.1.	Cấu trúc thư mục chaincode .....	17
3.5.2.	Định nghĩa .....	18
3.5.3.	Các logic transaction .....	18
3.5.4.	Các query transaction .....	19
3.6.	XÂY DỰNG API .....	20
3.6.1.	Identity.....	21
3.6.2.	Invoke/query một chaincode .....	22
3.7.	XÂY DỰNG ỨNG DỤNG .....	22

<b>CHƯƠNG 4: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ .....</b>	<b>23</b>
4.1. KẾT QUẢ THỰC NGHIỆM .....	23
4.1.1. Khởi động Network .....	23
4.1.2. Khởi động API server .....	24
4.1.3. Khởi động Server Ứng dụng .....	24
4.1.4. Kịch bản 1 – Đăng nhập hệ thống .....	25
4.1.5. Kịch bản 2 – Chức năng liệt kê mọi tài sản .....	25
4.1.6. Kịch bản 3 – Chức năng tạo một tài sản mới .....	25
4.1.7. Kịch bản 4 – Chức năng thay đổi chủ sở hữu .....	26
4.1.8. Kịch bản 5 – Chức năng xem lịch sử của tài sản: .....	27
4.1.9. Kịch bản 6 – Chức năng xóa tài sản .....	28
4.2. NHẬN XÉT ĐÁNH GIÁ KẾT QUẢ .....	28

# MỞ ĐẦU

## 1.1. Tổng quan về đề tài

Blockchain là một trong những công nghệ mang tính đột phá trong thế hệ các công nghệ mới của thế giới. Trong đó blockchain đã giải quyết được bài toán về độ tin cậy của các thông tin mà các hệ thống yêu cầu niềm tin hiện tại dường như khó mà có sự đảm bảo lớn.

Hyperledger Fabric là một framework hỗ trợ cho việc phát triển các giải pháp và ứng dụng blockchain, theo hướng phù hợp với kinh doanh với ý tưởng sổ cái phân tán với giới hạn cấp phép truy cập. Framework được thiết kế mang tính module và đa năng thỏa mãn nhiều usecase của người dùng trong doanh nghiệp.

Kết hợp với nhau Hyperledger cho phép thiết lập một network blockchain với khả năng quản lý truy cập riêng tư (permissionised). Nhờ vậy khi triển khai ứng dụng xác thực tài sản sẽ đảm bảo tính vẹn toàn, đầy đủ của dữ liệu mà vẫn đảm bảo tính riêng tư về tài sản sở hữu của các bên liên quan.

Trong đề tài này, nhóm triển khai một blockchain network có hai organizations (nhà quản lý và các công ty sản xuất) tham gia trên các channel tương tác chung giữ các sổ cái về tài sản của các bên liên quan. Ứng dụng blockchain này cho phép tra cứu, chuyển đổi chủ sở hữu, chấm dứt sở hữu một cách minh bạch khả kiến.

## 1.2. Mục đích và ý nghĩa của đề tài

### 1.2.1. Mục đích

Tìm hiểu được các khái niệm của mạng blockchain, xây dựng một mạng blockchain sử dụng hyperledger fabric, triển khai được ứng dụng cơ bản tương tác với cơ sở dữ liệu blockchain, hiểu được cách quản lý các package trên hệ điều hành linux và cách sử dụng các container (docker).

### **1.2.2. Ý nghĩa**

Tạo nên một sản phẩm được triển khai thực tiễn, thúc đẩy tư duy tự tìm hiểu, cài đặt và tự nghiên cứu của sinh viên. Nâng cao năng lực tìm hiểu, phân tích và giải quyết vấn đề.

### **1.3. Phương pháp thực hiện**

Phân tích bài toán và thiết kế hệ thống để giải quyết vấn đề của việc xác thực tài sản trên blockchain, thực hiện các thực nghiệm kiểm tra hiệu quả của giải pháp.

### **1.4. Bố cục của báo cáo**

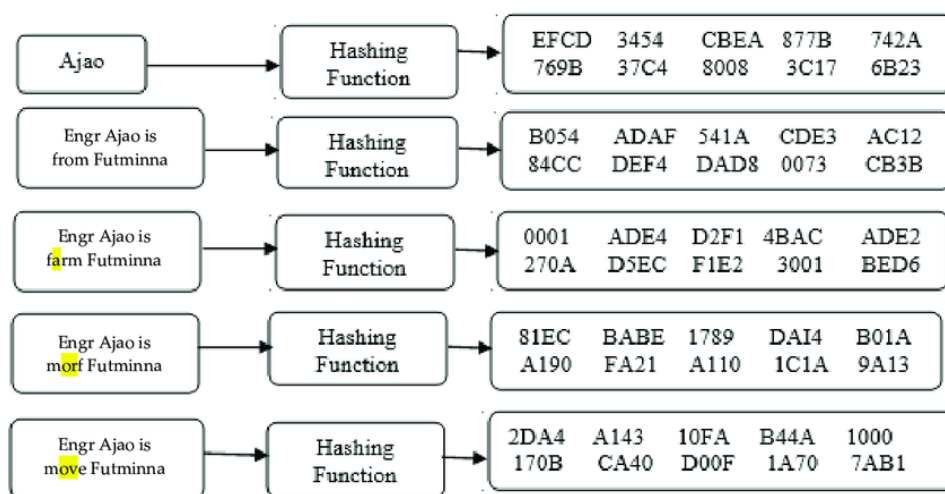
Báo cáo này bao gồm 4 phần

1. Cơ sở lý thuyết
2. Phân tích thiết kế
3. Triển khai và đánh giá kết quả
4. Kết luận

## CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

### 2.1. Hash Function

Hash function là một function được sử dụng để ánh xạ một dữ liệu có kích cỡ tùy ý đến một giá trị có kích cỡ cố định. Giá trị được trả về của hash function được gọi là hash values, hash codes, digests hoặc đơn giản là hashes. Với mỗi dữ liệu sẽ cho ra một mã hash, nếu chỉ cần thay đổi một bit hay một kí tự trong dữ liệu đó thì ta sẽ thu được một mã hash hoàn toàn mới toanh. Và cũng gần như là không thể để suy ra được dữ liệu từ một mã hash. Việc này đảm bảo tính toàn vẹn cho blockchain. Vì vậy, hash function được coi như là xương sống của mạng blockchain.

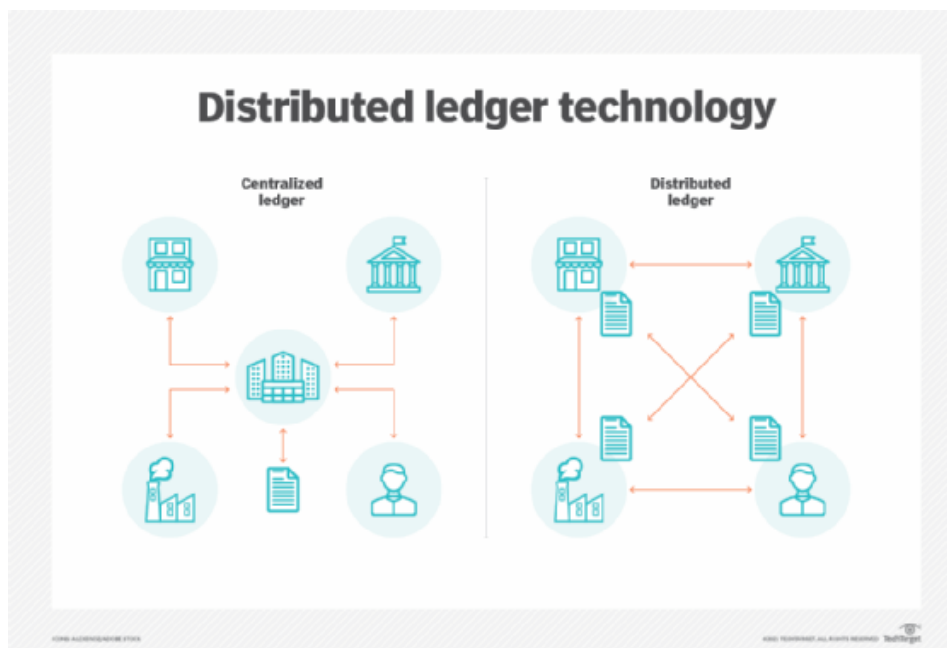


Hình 1. Mô phỏng hoạt động của hash function.

Trong Blockchain, giá trị hash được tạo ra dựa trên thông tin của block header. Mỗi block sẽ chứa giá trị hash của block-header của block phía trước nó, việc đó đảm bảo rằng không có một block giả mạo nào được chèn vào blockchain.

### 2.2. Distributed Ledger

Distributed Ledger là một cơ sở dữ liệu mà dữ liệu được chia sẻ, trùng lặp và đồng bộ giữa các bên tham gia trong một mạng blockchain. Không giống như các giao dịch tài chính và ngân hàng truyền thống – được điều khiển bởi một cơ quan trung tâm, blockchain được dựa vào các node được phân phối và phi tập trung. Tất cả các node thành viên của mạng đều có một bản copy của transaction và yêu cầu sự đồng thuận (consensus – sẽ nói ở phần sau) của đa số các thành viên để thêm một transaction mới vào mà không cần bất kì bên thứ ba nào, điều này giúp nâng cao sự tin tưởng và tính toàn vẹn của blockchain. Mỗi record trong ledger đều có timestamp và chữ kí mã hóa duy nhất, do đó khiến cho sổ cái có tính bất biến và giúp hỗ trợ kiểm tra tất cả giao dịch trong mạng.



Hình 2. Sổ cái phân tán

### 2.3. Smart Contract

Để đảm bảo cho việc cập nhật thông tin một cách nhất quán – và cho phép thực hiện những chức năng của sổ cái (giao dịch, truy vấn...) – một mạng blockchain sử dụng các smart contract để cung cấp quyền truy cập đến sổ cái.



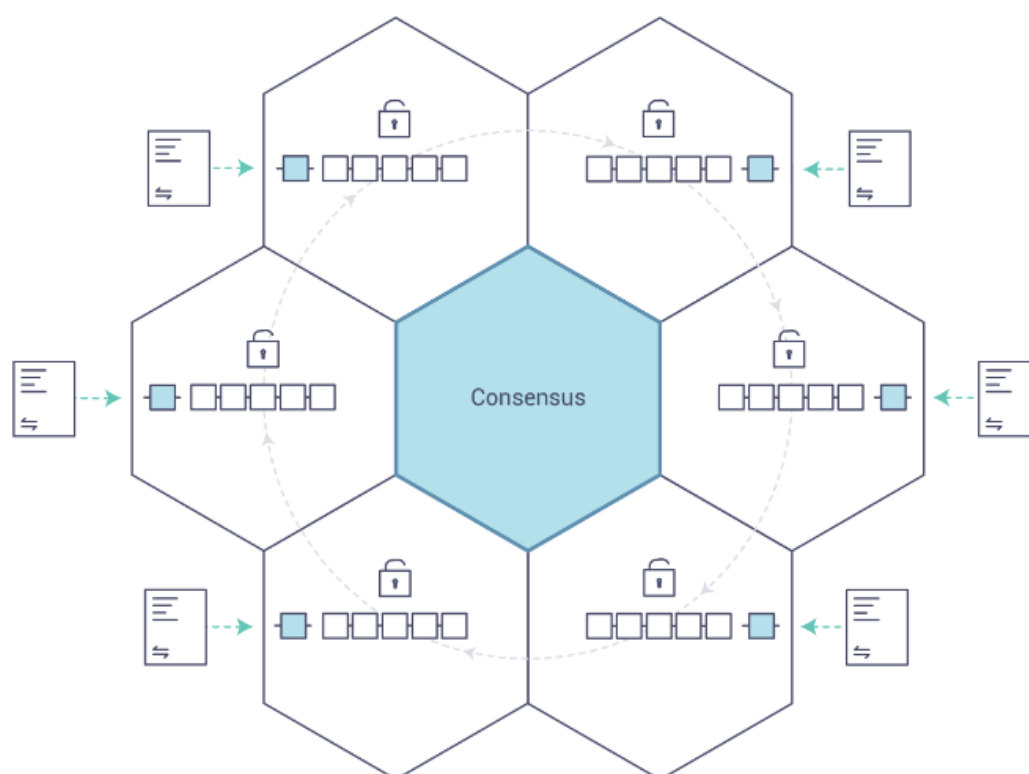
Hình 3. Smart contract cho phép truy cập đến sổ cái.

Smart contract không chỉ là cơ chế chính để đóng gói thông tin và giữ cho mọi thứ đơn giản trên toàn bộ network, mà còn có thể được viết để cho phép các bên tham gia thực thi một vài phần của giao dịch một cách tự động.

Một smart contract có thể, ví dụ, được viết để tính giá để vận chuyển một món hàng khi mà giá vận chuyển có thể thay đổi tùy thuộc vào món hàng đó đến nhanh như thế nào. Với khái niệm được xác nhận bởi hai thực thể tham gia và được viết vào sổ cái, một lượng tiền sẽ được cập nhật trừ một cách tự động khi món hàng được giao thành công.

## 2.4. Consensus

Là quá trình giữ cho những giao dịch trong sổ cái được đồng nhất trên toàn mạng – để đảm bảo sổ cái chỉ cập nhật khi giao dịch được xác nhận bởi các bên tham gia thích hợp, và sau đó sổ cái sẽ được cập nhật, và tất cả sẽ cập nhật với thứ tự giống hệt nhau gọi là quá trình đồng thuận – consensus.



Hình 4. Quá trình đồng thuận đảm bảo mọi sổ cái trên mạng blockchain giống hệt nhau.

## 2.5. Hyperledger fabric khác các mạng blockchain khác

Trong một mạng blockchain, bất kỳ bên liên quan nào tham gia vào cũng có một bản copy của sổ cái. Bên cạnh sổ cái được chia sẻ, quá trình cập nhật sổ cái cũng được chia sẻ cho các bên đó. Bằng việc có thể xem được thông tin thông suốt trên mạng, các mạng blockchain có thể giảm thời gian, chi phí và rủi ro với các thông tin nhạy cảm trong khi có thể cung cấp được sự tin tưởng và khả kiến.

Hyperledger fabric khác các mạng blockchain khác ở tính riêng tư và cấp phép. Một hệ thống blockchain mở không cấp phép sẽ cho phép các thực thể với danh tính không xác định tham gia vào mạng (yêu cầu phải sử dụng các giao thức như “proof of work” để xác thực các giao dịch và đảm bảo ổn định cho mạng), trong khi các thành viên của mạng



hyperledger fabric tham gia vào mạng thông qua một thực thể cấp phép tin cậy Membership Service Provider (MSP).

Mạng Hyperledger Fabric cũng cho phép tạo channels, cho phép một nhóm các thành viên tham gia tạo những sổ cái lưu giao dịch khác nhau. Đây là tính năng quan trọng cho phép những thành viên của network có thể là những bên cạnh tranh nhau không nhìn thấy những thông tin quan trọng trong kinh doanh.

## **CHƯƠNG 3: PHÂN TÍCH THIẾT KẾ HỆ THỐNG**

### **3.1. PHÁT BIỂU BÀI TOÁN**

Có nhiều công ty sản xuất tài sản với các mã định danh, nhà cầm quyền muốn cung cấp một hệ thống có chức năng xác thực tài sản để hỗ trợ nhanh chóng cho các dịch vụ yêu cầu xác thực tài sản của mình như ngân hàng, đảm bảo điều kiện nhập cư.

Thiết kế một hệ thống blockchain có các tổ chức là nhà cầm quyền và các công ty sản xuất tài sản, trong đó mỗi công ty khi bán sản phẩm ra thị trường phải khai báo lên hệ thống về sự tồn tại của sản phẩm và sở hữu tài sản cho người mua. Nhà cầm quyền thực hiện kiểm tra chủ sở hữu và thực hiện các dịch vụ về chuyển nhượng quyền sở hữu tài sản cũng như tuyên bố tiêu hủy tài sản và sở hữu tài sản.

### **3.2. PHÂN TÍCH HIỆN TRẠNG**

Hiện nay chưa có hệ thống nào phục vụ mục tiêu tương tự cho Việt Nam, việc xác nhận quyền sở hữu tài sản – ví dụ như – xe hơi thường được xác nhận sở hữu qua cà vẹt xe, quyền sử dụng đất được xác nhận qua các giấy tờ liên quan sổ đỏ. Các chứng từ gốc của các giấy tờ đó được văn thư nhà nước lưu trữ và người liên quan cũng có các bản chứng nhận quyền sở hữu/sử dụng. Các ngân hàng khi cần xác thực tài sản phải đi qua nhiều quy trình xác nhận phức tạp thì mới giải ngân vốn cho người cần. Với hệ thống hiện tại như vậy, người Việt phải tốn rất nhiều thời gian thực hiện các quy trình hành chính với rất nhiều chi phí, rủi ro và hiệu quả thấp.

Các chức năng trên đều có thể thực hiện được sử dụng một hệ thống blockchain bằng hyperledger fabric mà vẫn đảm bảo tính nhất quán, tin cậy, an toàn, khả kiến của thông tin trên toàn bộ hệ thống.

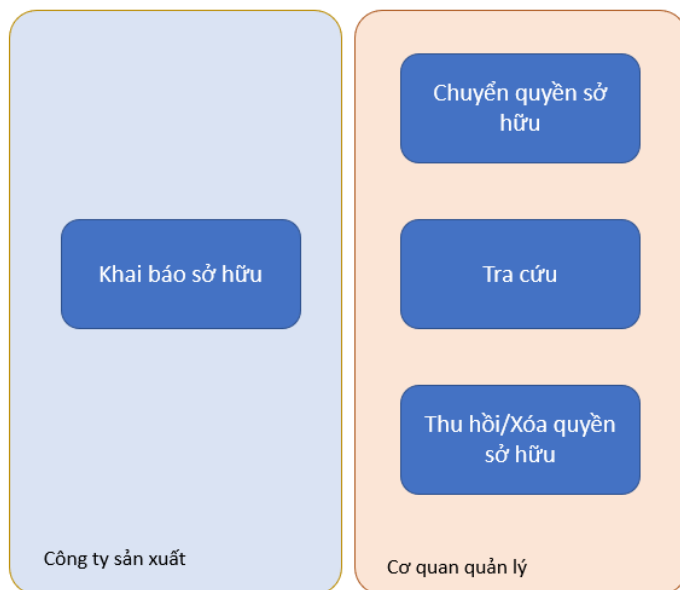
### **3.3. PHÂN TÍCH CHỨC NĂNG**

#### **3.3.1. Đối tượng sử dụng**

Cơ quan nhà nước, các công ty và các bên liên quan trong quan hệ sở hữu.

Tiến trình xác thực tài sản được thực hiện theo các bước như sau:

1. Nhà sản xuất/bên nhập khẩu hàng hóa khai báo sở hữu sản phẩm lên hệ thống khi họ bán cho bên mua.
2. Bên sở hữu sản phẩm đến cơ quan chức năng khai báo chuyển quyền sở hữu sản phẩm.
3. Các bên khác yêu cầu cơ quan chức năng truy xuất tài sản liên quan của người sở hữu xác định.
4. Bên sở hữu sản phẩm hoặc cơ quan chức năng thu hồi quyền sở hữu tài sản.



Hình 5. Các chức năng của các tổ chức

### 3.3.2. Khai báo sở hữu

Khởi tạo một tài sản mới trên hệ thống, bao gồm khai báo các thông tin của nhà sản xuất, tên tài sản, số hiệu quản lý tài sản, số hiệu chủ sở hữu tài sản và tên chủ sở hữu tài sản. Ngoài ra các thông tin như ngày đăng ký, ngày chỉnh sửa sẽ được hệ thống tự động thêm vào.

```

{
  "Key": "ASSET5",
  "Record": {
    "manufacturer": "Vinfast",
    "name": "Fadil",
    "serialnumber": "F00192182",
    "ownerid": "201799069",
    "ownername": "Đàm Quang Tiến",
    "registerdate": "2022-06-11 08:07:58.995 +0000 UTC",
    "lastupdatedate": "2022-06-11 08:07:58.995 +0000 UTC",
    "isdelete": false
  }
}

```

Hình 6. Các thông tin của tài sản khi mới được khai báo.

### 3.3.3. Thay đổi chủ sở hữu

Chủ sở hữu sẽ yêu cầu thay đổi chủ sở hữu với các thông tin của chủ sở hữu mới. Khi đó trạng thái bản ghi trên sổ cái như **Hình 7**.

```
"manufacturer": "Vinfast",  
"name": "Fadil",  
"serialnumber": "F00192182",  
"ownerid": "102180048",  
"ownername": "Phùng Văn Thắng",  
"registerdate": "2022-06-11 08:07:58.995 +0000 UTC",  
"lastupdatedate": "2022-06-11 08:13:08.065 +0000 UTC",  
"isdelete": false
```

Hình 7. Trạng thái của bản ghi khi thay đổi chủ sở hữu bao gồm thay đổi id và tên chủ sở hữu.

#### 3.3.4. Truy vấn chủ sở hữu của tài sản

Có thể thực hiện truy vấn sở hữu của một tài sản xác định, truy vấn tài sản của toàn mạng (tính năng có thể giới hạn trong tương lai).

Các phương thức truy vấn tài sản không yêu cầu endorsement của cả hai peer của các tổ chức vì transaction này không dẫn đến thay đổi trạng thái của sổ cái.

- a) Truy vấn sở hữu một tài sản
- b) Truy vấn sở hữu của toàn bộ tài sản trên mạng
- c) Truy vấn lịch sử chuyển đổi sở hữu của một tài sản xác định

Tác vụ truy vấn sở hữu của toàn bộ tài sản trên mạng chưa được cài đặt tính năng pagination nên có thể gây lỗi tràn body của một request, trong các bản cập nhật tương lai có thể điều chỉnh khả năng này.

Bên cạnh đó trong chaincode của hệ thống này chưa hỗ trợ các ad hoc query (rich query). Các query này thường cho phép thực hiện các truy vấn phức tạp hơn phục vụ các nghiệp vụ liên quan xác thực tài sản sau này, ví dụ truy vấn tài sản của một khu vực đô thị, truy vấn tài sản của tất cả các thành viên của một tổ chức.

```

{
  "Key": "ASSET0",
  "Record": {
    "manufacturer": "Toyota",
    "name": "Toyota Prius",
    "serialnumber": "TE_0000",
    "ownerid": "DamQuangTien002",
    "ownername": "003",
    "registerdate": "2022-01-02",
    "lastupdatedate": "2022-06-11 08:05:33.385198477 +0000 UTC",
    "isdelete": false
  }
},
{
  "Key": "ASSET1",
  "Record": {
    "manufacturer": "Toyota",
    "name": "Toyota Madza",
    "serialnumber": "TE_1000",
    "ownerid": "01",
    "ownername": "Thang",
    "registerdate": "2022-01-02",
    "lastupdatedate": "2022-03-06",
    "isdelete": false
  }
},
{
  "Key": "ASSET3",
  "Record": {
    "manufacturer": "Than",
    "name": "xe hoi",
    "serialnumber": "hha",
    "ownerid": "111",
    "ownername": "DQTT",
    "registerdate": "2022-06-11 08:05:29.182070342 +0000 UTC",
  }
}

```

Hình 8. Kết quả truy vấn tất cả tài sản có trong mạng.

### 3.3.5. Hủy sở hữu tài sản

Hủy sở hữu tài sản tức là tuyên bố dừng sự tồn tại của tài sản trên hệ thống, bởi vì hyperledger fabric nói riêng và blockchain nói chung không cho phép thực hiện xóa hoàn toàn dấu vết của một tài sản trên hệ thống. Nên tính năng này sẽ thực hiện thao tác xóa mềm, tức là đổi cờ trạng thái của một tài sản thành đã xóa tồn tại.

```

{
  "Key": "32cbe7970d093a3dbb9008771c9f5aa1004612fb841c976391bd8a7bc14386fd",
  "Record": {
    "manufacturer": "Vinfast",
    "name": "Fadil",
    "serialnumber": "F00192182",
    "ownerid": "102180048",
    "ownername": "Phùng Văn Thắng",
    "registerdate": "2022-06-11 08:07:58.995 +0000 UTC",
    "lastupdatedate": "2022-06-11 08:35:47.081 +0000 UTC",
    "isdelete": true
  }
},
{
  "Key": "8adfc6b69cc6baa247a467c15834eac2069fbb47589fd5bc2554182c6e81180ea",
  "Record": {
    "manufacturer": "Vinfast",
    "name": "Fadil",
    "serialnumber": "F00192182",
    "ownerid": "102180048",
    "ownername": "Phùng Văn Thắng",
    "registerdate": "2022-06-11 08:07:58.995 +0000 UTC",
    "lastupdatedate": "2022-06-11 08:13:08.065 +0000 UTC",
    "isdelete": false
  }
},

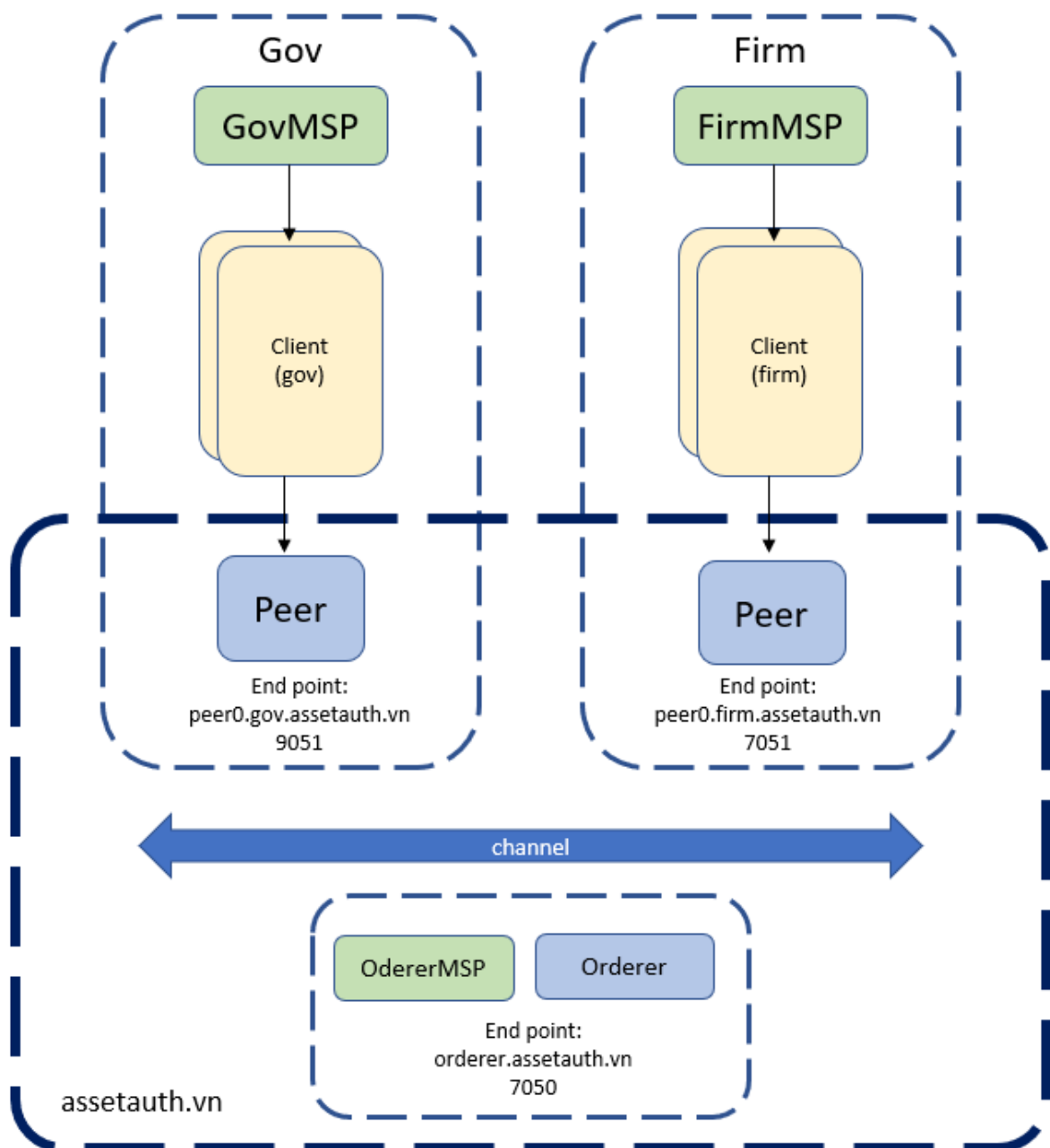
```

Hình 9. Lịch sử thay đổi của một tài sản bị tuyên bố hủy sở hữu.

### 3.4. THIẾT KẾ NETWORK

Trong hệ thống xác thực mạng Assetauth.vn bao gồm hai tổ chức, mỗi tổ chức có các ứng dụng phía người dùng riêng, có MSP (máy chủ cấp định danh), có các peer để lưu sổ cái và tiến hành các chính sách xác thực tài sản của riêng mình.

Tất cả các peer giao tiếp với nhau qua channel chung với các orderer đặt tại cơ quan chính phủ.



Hình 10. Sơ đồ topology cho các tổ chức thuộc mạng xác thực tài sản.

Với các tính năng và thiết kế như trên ta đã có thể triển khai được network và cài đặt chaincode.

### 3.4.1. Khởi tạo file crypto-config, configtx.yaml và tạo channel artifact

#### 1. Crypto-config.yaml

Khởi tạo trong thư mục artifacts/channel

File này chủ yếu để tạo các crypto material cho các thành phần thuộc network sau này, do đó quan trọng là khai báo số lượng các đối tượng này để chạy câu lệnh cryptogen để sinh ra các artifacts cần thiết.

```

# -----
# "PeerOrgs" - Definition of organizations managing peer
# -----
PeerOrgs:
# -----
# Org1
# -----
- Name: gov
  Domain: gov.assetauth.vn
  EnableNodeOUs: true

  Template:
    Count: 2
    # Start: 5
    # Hostname: {[.Prefix]}{[.Index]} # default
    SANS:
      - "localhost"

  Users:
    Count: 1

- Name: firm
  Domain: firm.assetauth.vn
  EnableNodeOUs: true

  Template:
    Count: 2
    # Start: 5
    # Hostname: {[.Prefix]}{[.Index]} # default
    SANS:
      - "localhost"

  Users:
    Count: 1

```

Hình 11. Khai báo peer trong cryptoconfig để chuẩn bị crypto materials.

## 2. Configtx.yaml

File này cũng chứa ở thư mục như crypto-config

Được sử dụng để config các cài đặt của channel như quy định các chính sách endorsement cho các tác vụ như read, write hay admin đối với mỗi tổ chức, quy định đường dẫn đến các crypto material đã được sinh ra bởi crypto-config, hỗ trợ sinh ra các anchor peer, system genesis block (các bước config hết sức cần thiết để khởi tạo một channel).



```

        Type: ImplicitMeta
        Rule: "ANY Writers"
    Admins:
        Type: ImplicitMeta
        Rule: "MAJORITY Admins"
    LifecycleEndorsement:
        Type: ImplicitMeta
        Rule: "MAJORITY Endorsement"
    Endorsement:
        Type: ImplicitMeta
        Rule: "MAJORITY Endorsement"

    Capabilities:
      <<: *ApplicationCapabilities
#####
#
#   SECTION: Orderer
#
#   - This section defines the values to encode into a config transaction or
#   genesis block for orderer related parameters
#
#####
Orderer: &OrdererDefaults

    # Orderer Type: The orderer implementation to start
    OrdererType: etcdraft

    EtcdRaft:
        Consenters:
            - Host: orderer.assetauth.vn
              Port: 7050
              ClientTLS-cert: crypto-config/ordererOrganizations/assetauth.vn/orderers/
orderer.assetauth.vn/tls/server.crt
              ServerTLS-cert: crypto-config/ordererOrganizations/assetauth.vn/orderers/
orderer.assetauth.vn/tls/server.crt

        Addresses:
            - orderer.assetauth.vn:7050

    # Batch Timeout: The amount of time to wait before creating a batch
    BatchTimeout: 2s

    # Batch Size: Controls the number of messages batched into a block
    BatchSize:

```

YAML ▾ Tab Width: 8 ▾ Ln 28, Co

Hình 12. Configtx.yaml quy định các config của channel.

### 3. Khởi tạo các artifacts

```

chmod -R 0755 ./crypto-config
# Delete existing artifacts
rm -rf ./crypto-config
rm genesis.block mychannel.tx
rm -rf ../../channel-artifacts/*

#Generate Crypto artifacts for organizations
cryptogen generate --config=./crypto-config.yaml --output=./crypto-config/

# System channel
SYS_CHANNEL="sys-channel"

# channel name defaults to "mychannel"
CHANNEL_NAME="mychannel"

echo $CHANNEL_NAME

# Generate System Genesis block
configtxgen -profile OrdererGenesis -configPath . -channelID $SYS_CHANNEL -outputBlock ./genesis.block

# Generate channel configuration block
configtxgen -profile BasicChannel -configPath . -outputCreateChannelTx ./mychannel.tx -channelID $CHANNEL_NAME

echo "##### Generating anchor peer update for FirmMSP #####"
configtxgen -profile BasicChannel -configPath . -outputAnchorPeersUpdate ./FirmMSPanchors.tx -channelID $CHANNEL_NAME -asOrg FirmMSP

echo "##### Generating anchor peer update for GovMSP #####"
configtxgen -profile BasicChannel -configPath . -outputAnchorPeersUpdate ./GovMSPanchors.tx -channelID $CHANNEL_NAME -asOrg GovMSP

```

Hình 13. Script khởi tạo các artifact cần thiết cho channel

### 3.4.2. Khởi tạo file docker-compose.yaml và tạo nên các peer của network

File docker-compose.yaml được sử dụng khi khởi tạo các peer của network, do đó nó chứa các thông tin như quy định môi trường, port, đường dẫn đến các chứng chỉ, các crypto-material để thực hiện xác thực. Quy định khởi tạo các services CA cho các tổ chức, khởi tạo các orderer, các peer, cài đặt mật khẩu truy cập cơ sở dữ liệu...

```

peer1.gov.assetauth.vn:
  container_name: peer1.gov.assetauth.vn
  extends:
    file: base.yaml
    service: peer-base
  environment:
    - FABRIC_LOGGING_SPEC=info
    - ORDERER_GENERAL_LOGLEVEL=info
    - CORE_PEER_LOCALMSPID=GovMSP

    - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=artifacts_test

    - CORE_PEER_ID=peer1.gov.assetauth.vn
    - CORE_PEER_ADDRESS=peer1.gov.assetauth.vn:10051
    - CORE_PEER_LISTENADDRESS=0.0.0.0:10051
    - CORE_PEER_CHAINCODEADDRESS=peer1.gov.assetauth.vn:10052
    - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:10052
    # Exposed for discovery Service
    - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer1.gov.assetauth.vn:10051
    - CORE_PEER_GOSSIP_BOOTSTRAP=peer0.gov.assetauth.vn:9051

    - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
    - CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=couchdb3:5984
    - CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=
    - CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=
    - CORE_METRICS_PROVIDER=prometheus
    # - CORE_OPERATIONS_LISTENADDRESS=0.0.0.0:9440
    - CORE_PEER_TLS_ENABLED=true
    - CORE_PEER_TLS_CERT_FILE=/etc/hyperledger/crypto/peer/tls/server.crt
    - CORE_PEER_TLS_KEY_FILE=/etc/hyperledger/crypto/peer/tls/server.key
    - CORE_PEER_TLS_ROOTCERT_FILE=/etc/hyperledger/crypto/peer/tls/ca.crt
    - CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/crypto/peer/msp
  ports:
    - 10051:10051
  volumes:
    - ./channel/crypto-config/peerOrganizations/gov.assetauth.vn/peers/peer1.gov.assetauth.vn/
sp:/etc/hyperledger/crypto/peer/msp
    - ./channel/crypto-config/peerOrganizations/gov.assetauth.vn/peers/peer1.gov.assetauth.vn/
ls:/etc/hyperledger/crypto/peer/tls
    - /var/run:/host/var/run/
    - ./channel:/etc/hyperledger/channel/

```

Hình 14. Các thông số để khởi tạo peer 1 thuộc tổ chức gov của network.

### 3.4.3. Tạo channel và join các peer vào channel

```

createChannel(){
  rm -rf ./channel-artifacts/*
  setGlobalsForPeer0Org1

  peer channel create -o localhost:7050 -c $CHANNEL_NAME \
  --ordererTLSHostnameOverride orderer.assetauth.vn \
  -f ./artifacts/channel/${CHANNEL_NAME}.tx --outputBlock ./channel-artifacts/$
{CHANNEL_NAME}.block \
  --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA
}

```

Hình 15. Khởi tạo channel bằng câu lệnh peer channel create trên peer đóng vai trò orderer.

```

joinChannel(){
    setGlobalsForPeer0Org1
    peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block

    setGlobalsForPeer1Org1
    peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block

    setGlobalsForPeer0Org2
    peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block

    setGlobalsForPeer1Org2
    peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
}

updateAnchorPeers(){
    setGlobalsForPeer0Org1
    peer channel update -o localhost:7050 --ordererTLSHostnameOverride orderer.assetauth.vn -c
    $CHANNEL_NAME -f ./artifacts/channel/${CORE_PEER_LOCALMSPID}anchors.tx --tls
    $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA

    setGlobalsForPeer0Org2
    peer channel update -o localhost:7050 --ordererTLSHostnameOverride orderer.assetauth.vn -c
    $CHANNEL_NAME -f ./artifacts/channel/${CORE_PEER_LOCALMSPID}anchors.tx --tls
    $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA
}

```

Hình 16. Joinchannel và cập nhật anchor peer cho các tổ chức thuộc channel

Bởi vì orderer phải biết được mỗi tổ chức phải có ít nhất một peer đang vận hành để nó ping để nhằm xác thực các transaction. Anchor peer đóng vai trò là peer đầu tiên/công khai mà orderer hoặc mọi thành phần của channel có thể liên hệ được.

#### 3.4.4. Cài đặt chaincode lên channel

Cài đặt chaincode lên channel bao gồm các bước:

1. Đóng gói chaincode
2. Cài đặt chaincode lên các peer thuộc channel
3. Approve chaincode cho mỗi tổ chức thuộc channel
4. Kiểm tra chaincode đã được kí
5. Commit chaincode
6. Kiểm tra commit
7. Invoke hoặc query chaincode

Để update chaincode cũng thực hiện các bước tương tự.

### 3.5. CÁC CHAINCODE

Chaincode được viết bằng ngôn ngữ lập trình Golang, lưu tại thư mục `/BasicNetwork-2.0/artifacts/src/github.com/asset_contract_api/go`

#### 3.5.1. Cấu trúc thư mục chaincode

```

├─ assetauth.go
├─ go.mod
└─ go.sum

```

```
└─ vendor
    ├── github.com
    ├── golang.org
    ├── google.golang.org
    ├── gopkg.in
    └─ modules.txt
```

Trong đó file `assetauth.go` chứa `chaincode`, `go.mod` chứa các requirement để chạy `chaincode`.

Trước khi đóng gói `chaincode` và cài đặt lên các peer, phải thực hiện câu lệnh `go mod vendor` để cài đặt các package cần thiết, trong đó quan trọng là thư viện `github.com/hyperledger/fabric-contract-api-go v1.0.0`

### 3.5.2. Định nghĩa

Trong `chaincode` này các function gọi là `SmartContract`, các `smartcontract` này kế thừa từ lớp `contract` thuộc thư viện `contract api` mà mình đã require ở trên.

```
type SmartContract struct {
    contractapi.Contract
}
```

### 3.5.3. Các logic transaction

Như đã đề cập trước đó, `assetauth` sẽ cần 5 transaction, trong đó các transaction liên quan đến thay đổi trạng thái dữ liệu là quan trọng nhất, nên ở phần này chỉ đề cập đến các logic transaction bao gồm:

1. `CreateAsset`
2. `DeleteAsset`
3. `ChangeAssetOwner`

Khi hàm `createAsset` được gọi, nó sẽ lấy các tham số được truyền đến từ client và gán nó vào các attribute của tài sản mới:

```
func (s *SmartContract) CreateAsset(ctx contractapi.TransactionContextInterface,
    assetNumber string, manufacturer string, name string, serialNumber string, ownerID string,
    ownerName string) error {
    asset := Asset{
        Manufacturer: manufacturer,
        Name:         name,
        SerialNumber: serialNumber,
        OwnerID:      ownerID,
        OwnerName:    ownerName,
    }

    txntmsp, errN := ctx.GetStub().GetTxTimestamp()
    _ = errN
    time1 := time.Unix(txntmsp.Seconds, int64(txntmsp.Nanos)).String()

    asset.LastUpdateDate = time1
```

```

    asset.RegisterDate = time1
    asset.IsDelete = false
    assetAsBytes, _ := json.Marshal(asset)
    return ctx.GetStub().PutState(assetNumber, assetAsBytes)
}

```

Ở cuối hàm tạo tài sản, `ctx.GetStub().PutState()` được gọi để cập nhật trạng thái mới vào sổ cái. Từ dạng dữ liệu kiểu `Json` phải chuyển kiểu nó về thành kiểu mảng `byte`. Trong hàm này có một điểm đáng chú ý khi làm việc với những timestamps trong khi viết smartcontract, đó là vì các smartcontract sẽ được tất cả các peers thực thi rồi thực hiện chính sách endorsement do đó nếu gọi hàm `time.time()` thông thường thì dẫn đến trường hợp timestamp ở mỗi peer là khác nhau và sẽ không thể endorse được transaction mới vào sổ cái. Do đó phải sử dụng `ctx.GetStub().GetTxTimestamp()` để lấy thời gian trong context để đồng bộ với tất cả các peer.

Tương tự với hàm tạo tài sản, hai hàm cập nhật còn lại đều sử dụng hai hàm `QueryAsset` và `PutState` để lấy và cập nhật các tài sản ở trong network.

```

func (s *SmartContract) ChangeAssetOwner(ctx contractapi.TransactionContextInterface,
assetNumber string, newOwnerId string, newOwnerName string) error {
    asset, err := s.QueryAsset(ctx, assetNumber)
    if err != nil {
        return err
    }
    if asset.IsDelete == true {
        return fmt.Errorf("The designated asset has been deleted since %s",
asset.LastUpdateDate)
    }
    txntmsp, errN := ctx.GetStub().GetTxTimestamp()
    _ = errN
    time1 := time.Unix(txntmsp.Seconds, int64(txntmsp.Nanos)).String()
    asset.OwnerID = newOwnerId
    asset.OwnerName = newOwnerName
    asset.LastUpdateDate = time1
    assetAsBytes, _ := json.Marshal(asset)
    return ctx.GetStub().PutState(assetNumber, assetAsBytes)
}

```

### 3.5.4. Các query transaction

Các transaction query sẽ không thay đổi các nội dung của sổ cái nên chính sách endorsement sẽ không nghiêm ngặt bằng, như đã quy định ở lúc thiết lập channel, các transaction này chỉ yêu cầu một trong các peer đọc nội dung của sổ cái mà nó đang giữ là được.

```

func (s *SmartContract) QueryAsset(ctx contractapi.TransactionContextInterface,
assetNumber string) (*Asset, error) {
    assetAsBytes, err := ctx.GetStub().GetState(assetNumber)
    if err != nil {
        return nil, fmt.Errorf("Failed to read from world state. %s", err.Error())
    }
}

```

```

    }

    if assetAsBytes == nil {
        return nil, fmt.Errorf("%s does not exist", assetNumber)
    }

    asset := new(Asset)
    _ = json.Unmarshal(assetAsBytes, asset)
    return asset, nil
}

```

Thông thường các transaction này sử dụng các hàm được cung cấp bởi thư viện `contractapi` như `GetState()`, `GetStateByRange()`, `GetHistoryForKey()`.

### 3.6. XÂY DỰNG API

Đối với mỗi tổ chức sẽ phải viết các api riêng để giao tiếp với từng peer của từng tổ chức đó, tuy vậy ở bài này nhóm thực hiện viết api giao tiếp chung với các api khác nhau. Mỗi khi ứng dụng client thuộc tổ chức nào thì api với chữ kí thuộc tổ chức đó thực hiện lời gọi đến các peer xác định cho từng tác vụ.

Sử dụng thư viện `fabric-client` trên `nodejs` để viết api này. Thư viện này hỗ trợ các tính năng kết nối đến các peer mà không cần phải viết các câu lệnh dài dòng phức tạp. Thư viện này yêu cầu phải cấu hình các `connection-profile-path` đến các tổ chức để hướng dẫn nó cách liên hệ với peer cấp định danh, địa chỉ lấy các chứng chỉ, địa chỉ liên hệ các peer của `network`.

```

network
~/fabric-samples/asset-auth-netw
crypto-config.yaml x helper.js x config.json x
orderers:
  - orderer.assetauth.vn

peers:
  peer0.firm.assetauth.vn:
    endorsingPeer: true
    chaincodeQuery: true
    ledgerQuery: true
    eventSource: true

  peer1.firm.assetauth.vn:
    endorsingPeer: true
    chaincodeQuery: true
    ledgerQuery: true
    eventSource: true

  peer0.gov.assetauth.vn:
    endorsingPeer: true
    chaincodeQuery: true
    ledgerQuery: true
    eventSource: true

  peer1.gov.assetauth.vn:
    endorsingPeer: true
    chaincodeQuery: true
    ledgerQuery: true
    eventSource: true

chaincodes:
  - mycc:v0

organizations:
  Firm:
    mspid: FirmMSP

    peers:
      - peer0.firm.assetauth.vn
      - peer1.firm.assetauth.vn

  certificateAuthorities:
    - ca.firm.assetauth.vn

  adminPrivateKey:
    path: ../artifacts/channel/crypto-config/p

```

Hình 17. Network config hướng dẫn thư viện fabric-client để liên hệ đến network.

### 3.6.1. Identity

Mỗi request API đến đều phải có token access và định danh rõ ràng. Để hệ thống xác định được ai đang truy cập đến. Ở phần này có các route `registerUser()` và `login()` có nhiệm vụ cấp token hoặc đăng ký user mới vào hệ thống.

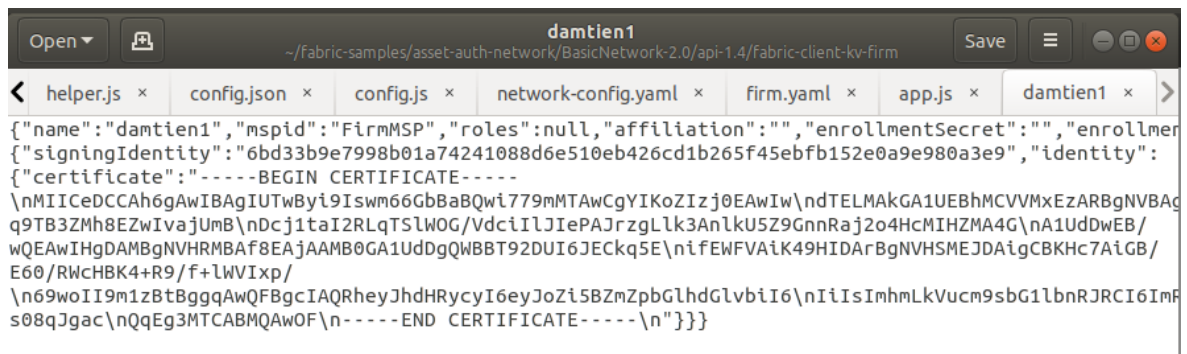
```

Pretty Raw Preview Visualize JSON
1
2  "success": true,
3  "secret": "",
4  "message": "damtien1 enrolled Successfully",
5  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE2NTQ5NzA3NTYsInVzZXJuYV11IjoizGFtdGllbGJlIiLCJvcmd0YV11IjoiaRmlybSIsImhhdCI6MTY1NDkzNDc1Nm8uZqa9aUhXdkkjhfvqSiawoNShnL_cHMTz-kj7ytN6DDM"
6

```

Hình 18. Kết quả đăng ký một user thuộc một tổ chức thành công, và được cấp 1 token truy cập





Hình 19. User đăng ký thành công sẽ được hệ thống kí một chứng chỉ từ peer định danh của tổ chức tương ứng. Khi nào người dùng này đăng nhập lại sẽ sử dụng chữ kí này để tương tác với network.

Do đó nếu khi hệ thống thực hiện một fresh reboot, phải xóa mọi identity của người dùng. Do các identity này không còn được ký bởi peer CA của network mới khởi tạo.

### 3.6.2. Invoke/query một chaincode

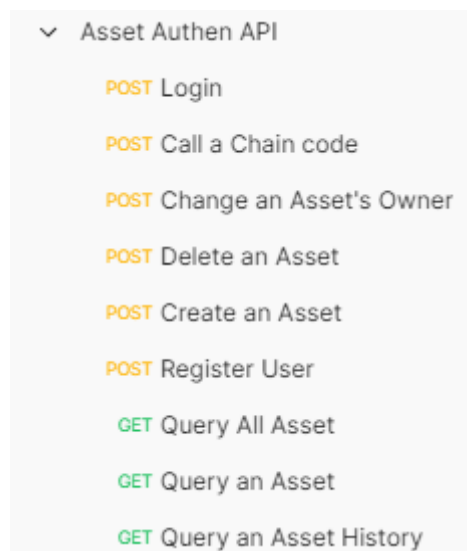
Thư viện Fabric-client đã hỗ trợ lớp invoke để gọi thực thi transaction bằng hàm `invoke.invokeChaincode`.

```
let message = await invoke.invokeChaincode(peers, channelName, chaincodeName, fcn,
args, req.username, req.orgname);
```

Việc query một chaincode cũng diễn ra tương tự khi sử dụng hàm `query.queryChaincode`

```
let message = await query.queryChaincode(peer, channelName, chaincodeName, args, fcn,
req.username, req.orgname);
```

Danh sách các api được cung cấp bởi hệ thống này được thể hiện như hình phía dưới.



Hình 20. Danh sách các route API đã triển khai.

## 3.7. XÂY DỰNG ỨNG DỤNG

Trong phần này, do không có đủ nguồn lực nhóm quyết định chỉ xây dựng một ứng dụng người dùng cho cả hai tổ chức. Công nghệ sử dụng là framework react trên nodejs. Ứng dụng demo này có các tính năng cơ bản phục vụ cho việc xác thực tài sản.

## CHƯƠNG 4: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

### 4.1. KẾT QUẢ THỰC NGHIỆM

#### 4.1.1. Khởi động Network

Tại thư mục gốc thực hiện chạy script start, đoạn script này sẽ dựng lên network như đã thiết kế tại các file config.

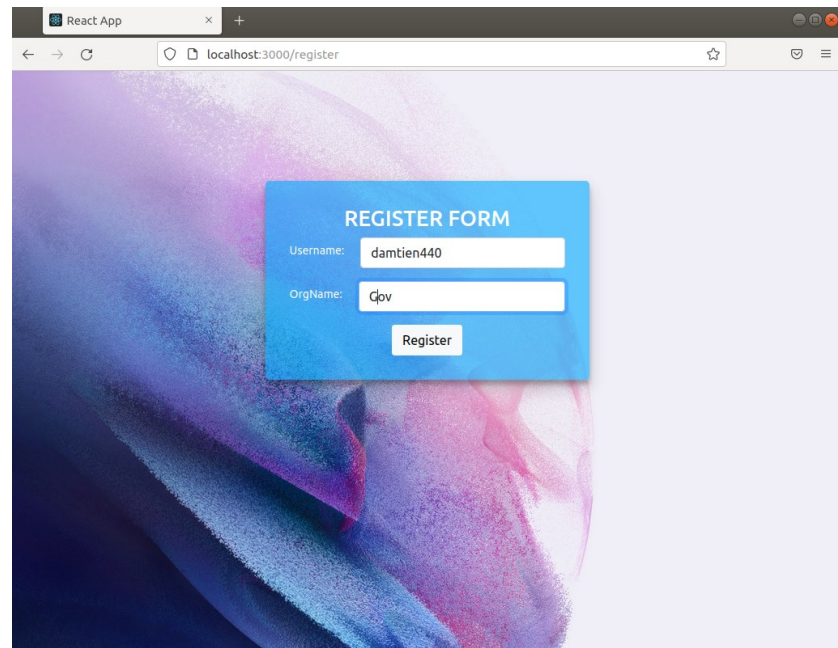
```
./start.sh

tien@ubuntu:~/fabric-samples/asset-auth-network/BasicNetwork-2.0$ ./start.sh
Stopping peer0.firm.assetauth.vn ... done
Stopping peer0.gov.assetauth.vn ... done
Stopping peer1.firm.assetauth.vn ... done
Stopping couchdb2 ... done
Stopping ca.firm.assetauth.vn ... done
Stopping orderer.assetauth.vn ... done
Stopping peer1.gov.assetauth.vn ... done
Stopping couchdb3 ... done
Stopping ca.gov.assetauth.vn ... done
Stopping couchdb1 ... done
Stopping couchdb0 ... done
Removing peer0.firm.assetauth.vn ... done
Removing peer0.gov.assetauth.vn ... done
Removing peer1.firm.assetauth.vn ... done
Removing couchdb2 ... done
Removing ca.firm.assetauth.vn ... done
Removing orderer.assetauth.vn ... done
Removing peer1.gov.assetauth.vn ... done
Removing couchdb3 ... done
Removing ca.gov.assetauth.vn ... done
Removing couchdb1 ... done
Removing couchdb0 ... done
Removing network artifacts_test
Creating network "artifacts_test" with the default driver
Creating couchdb1 ... done
Creating couchdb2 ... done
Creating orderer.assetauth.vn ... done
Creating peer1.gov.assetauth.vn ... done
Creating peer0.gov.assetauth.vn ... done
Creating ca.firm.assetauth.vn ... done
Creating peer1.firm.assetauth.vn ... done
Creating couchdb3 ... done
Creating couchdb0 ... done
Creating ca.gov.assetauth.vn ... done
Creating peer0.firm.assetauth.vn ... done
2022-06-11 07:18:58.171 PDT 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2022-06-11 07:18:58.193 PDT 0002 INFO [cli.common] readBlock -> Expect block, but got status: &{NOT_FOUND}
2022-06-11 07:18:58.197 PDT 0003 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2022-06-11 07:18:58.399 PDT 0004 INFO [cli.common] readBlock -> Expect block, but got status: &{SERVICE_UNAVAILABLE}
2022-06-11 07:18:58.401 PDT 0005 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2022-06-11 07:18:58.603 PDT 0006 INFO [cli.common] readBlock -> Expect block, but got status: &{SERVICE_UNAVAILABLE}
2022-06-11 07:18:58.607 PDT 0007 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2022-06-11 07:18:58.809 PDT 0008 INFO [cli.common] readBlock -> Expect block, but got status: &{SERVICE_UNAVAILABLE}
2022-06-11 07:18:58.813 PDT 0009 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2022-06-11 07:18:59.015 PDT 000a INFO [cli.common] readBlock -> Expect block, but got status: &{SERVICE_UNAVAILABLE}
2022-06-11 07:18:59.019 PDT 000b INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
```

Hình 21. Khởi tạo network



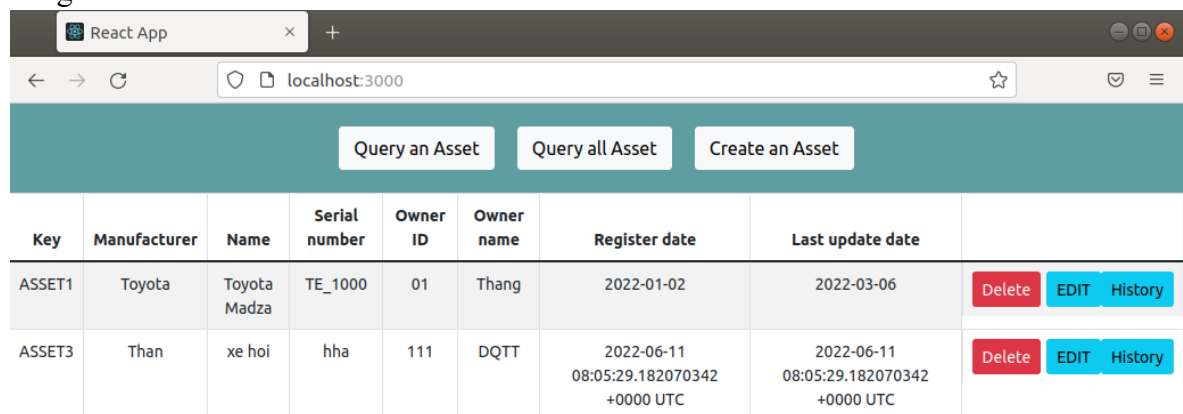
#### 4.1.4. Kịch bản 1 – Đăng nhập hệ thống



Hình 25. Giao diện đăng ký người dùng.

#### 4.1.5. Kịch bản 2 – Chức năng liệt kê mọi tài sản

Khi đăng nhập xong hoặc nhấn vào nút Query All Asset sẽ hiện ra tất cả các tài sản đang có.

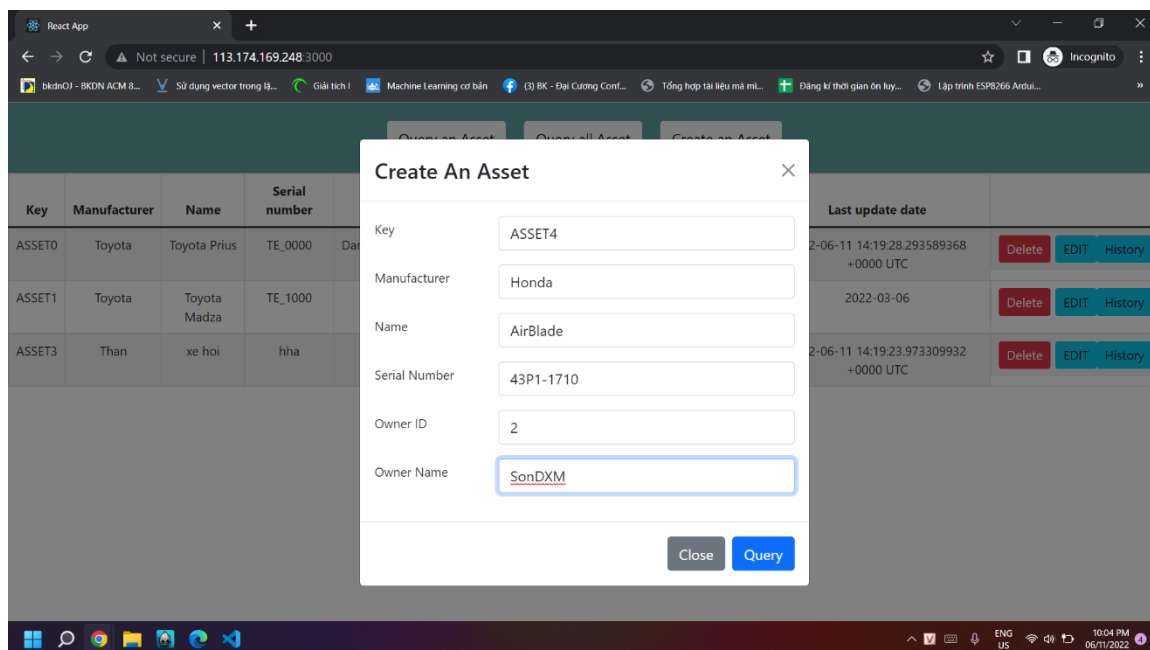


Query an Asset   Query all Asset   Create an Asset								
Key	Manufacturer	Name	Serial number	Owner ID	Owner name	Register date	Last update date	
ASSET1	Toyota	Toyota Madza	TE_1000	01	Thang	2022-01-02	2022-03-06	Delete EDIT History
ASSET3	Than	xe hoi	hha	111	DQT	2022-06-11 08:05:29.182070342 +0000 UTC	2022-06-11 08:05:29.182070342 +0000 UTC	Delete EDIT History

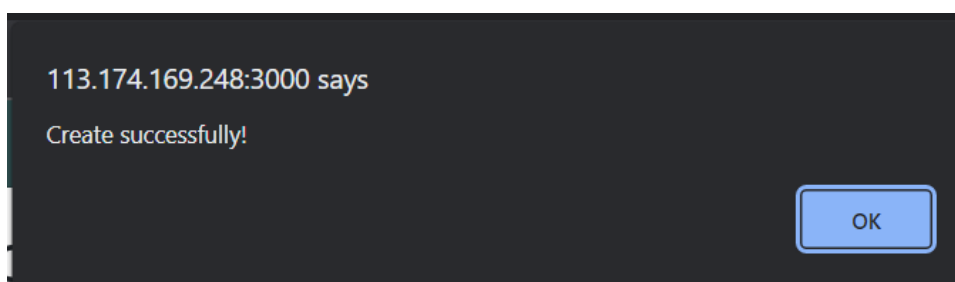
Hình 26. Đăng nhập thành công, trang web tự động vào giao diện liệt kê tất cả tài sản.

#### 4.1.6. Kịch bản 3 – Chức năng tạo một tài sản mới

Khi bấm vào nút Create an Asset sẽ hiện một cửa sổ để nhập thông tin của tài sản mới.



Bấm nút Query thì tài sản mới sẽ được lưu vào hệ thống. Sau khi lưu vào sẽ hiện thông báo tạo Asset thành công.



Hình 25. Thông báo tạo Asset thành công

Key	Manufacturer	Name	Serial number	Owner ID	Owner name	Register date	Last update date	
ASSET0	Toyota	Toyota Prius	TE_0000	DamQuangTien002	003	2022-01-02	2022-06-11 14:19:28.293589368 +0000 UTC	Delete EDIT History
ASSET1	Toyota	Toyota Madza	TE_1000	01	Thang	2022-01-02	2022-03-06	Delete EDIT History
ASSET3	Than	xe hoi	hha	111	DQTT	2022-06-11 14:19:23.973309932 +0000 UTC	2022-06-11 14:19:23.973309932 +0000 UTC	Delete EDIT History
ASSET4	Honda	AirBlade	43P1-1710	2	SonDXM	2022-06-11 15:05:39.565 +0000 UTC	2022-06-11 15:05:39.565 +0000 UTC	Delete EDIT History

Hình 26. Tài sản mới đã được thêm vào cuối danh sách các tài sản đã có

#### 4.1.7. Kịch bản 4 – Chức năng thay đổi chủ sở hữu

Bấm vào nút EDIT của tài sản vừa tạo (ASSET4).

Key	Manufacturer	Name	Serial number	Owner ID	Owner name	Register date	Last update date	
ASSET0	Toyota	Toyota Prius	TE_0000	Da			2-06-11 14:19:28.293589368 +0000 UTC	Delete EDIT History
ASSET1	Toyota	Toyota Madza	TE_1000				2022-03-06	Delete EDIT History
ASSET3	Than	xe hoi	hha				2-06-11 14:19:23.973309932 +0000 UTC	Delete EDIT History
ASSET4	Honda	AirBlade	43P1-1710				06-11 15:05:39.565 +0000 UTC	Delete EDIT History

### Change An Asset's Owner

Owner ID

Owner Name

Close Save

Hình 27. Form điền thông tin của chủ sở hữu mới cho tài sản  
Sau đó nhập thông tin của chủ sở hữu mới và nhấn nút Save.

## Change An Asset's Owner

Owner ID

Owner Name

Close Save

Hình 28. Điền thông tin của chủ sở hữu mới cho tài sản  
Sau đó chủ sở hữu mới đã được chuyển từ SonDXM thành Thang.

Key	Manufacturer	Name	Serial number	Owner ID	Owner name	Register date	Last update date	
ASSET0	Toyota	Toyota Prius	TE_0000	DamQuangTien002	003	2022-01-02	2022-06-11 14:19:28.293589368 +0000 UTC	Delete EDIT History
ASSET1	Toyota	Toyota Madza	TE_1000	01	Thang	2022-01-02	2022-03-06	Delete EDIT History
ASSET3	Than	xe hoi	hha	111	DQTT	2022-06-11 14:19:23.973309932 +0000 UTC	2022-06-11 14:19:23.973309932 +0000 UTC	Delete EDIT History
ASSET4	Honda	AirBlade	43P1-1710	01	Thang	2022-06-11 15:05:39.565 +0000 UTC	2022-06-11 15:10:35.273 +0000 UTC	Delete EDIT History

Hình 29. Chủ sở hữu mới đã được cập nhật cho tài sản

#### 4.1.8. Kịch bản 5 – Chức năng xem lịch sử của tài sản:

Bấm vào nút History bên cạnh Asset vừa thay đổi chủ sở hữu ở trên, từ đó ta thấy được sự thay đổi của thông tin tài sản.

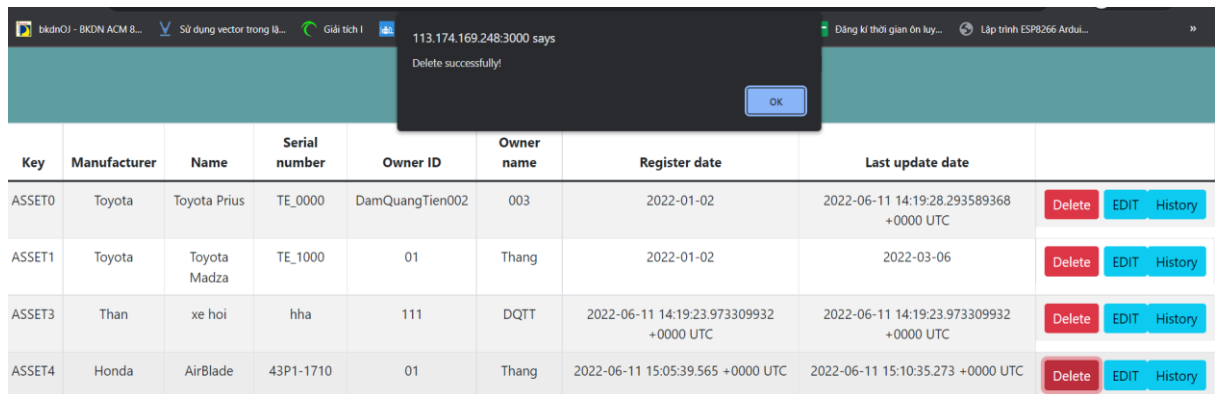
Key	Manufacturer	Name	Serial number	Owner ID	Owner name	Register date	Last update date
ASSET4	Honda	AirBlade	43P1-1710	01	Thang	2022-06-11 15:05:39.565 +0000 UTC	2022-06-11 15:10:35.273 +0000 UTC
ASSET4	Honda	AirBlade	43P1-1710	2	SonDXM	2022-06-11 15:05:39.565 +0000 UTC	2022-06-11 15:05:39.565 +0000 UTC
ASSET4	Honda	AirBlade	43P1-1710	2	SonDXM	2022-06-11 15:05:38.117 +0000 UTC	2022-06-11 15:05:38.117 +0000 UTC

Hình 30. Xem lịch sử của tài sản



#### 4.1.9. Kịch bản 6 – Chức năng xóa tài sản

Bấm vào nút delete của tài sản ASSET4, sẽ hiện ra thông báo Delete Successfully.



Hình 31. Bấm nút xóa tài sản

Query an Asset   Query all Asset   Create an Asset							
Key	Manufacturer	Name	Serial number	Owner ID	Owner name	Register date	Last update date
ASSET0	Toyota	Toyota Prius	TE_0000	DamQuangTien002	003	2022-01-02	2022-06-11 14:19:28.293589368 +0000 UTC
ASSET1	Toyota	Toyota Madza	TE_1000	01	Thang	2022-01-02	2022-03-06
ASSET3	Than	xe hoi	hha	111	DQTT	2022-06-11 14:19:23.973309932 +0000 UTC	2022-06-11 14:19:23.973309932 +0000 UTC

Hình 32. Tài sản đã được xóa khỏi danh sách các tài sản.

#### 4.2. NHẬN XÉT ĐÁNH GIÁ KẾT QUẢ

Qua kết quả thực hiện, nhóm đã triển khai được những tính năng cơ bản của ứng dụng blockchain sử dụng hyperledger fabric. Tuy nhiên, có thể nghiên cứu kỹ hơn về kỹ thuật xây dựng network và ứng dụng để có thể tạo ra một phiên bản hoàn chỉnh với nghiệp vụ cần.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 1. KẾT QUẢ ĐẠT ĐƯỢC

Về mặt lý thuyết, nhóm đã hiểu được cách một blockchain network sử dụng hyperledger fabric vận hành, hiểu được nền tảng mật mã học nằm phía dưới hỗ trợ cho sự tin cậy của hệ thống blockchain network.

Thực tế, nhóm đã cài đặt được ứng dụng sử dụng cơ sở dữ liệu blockchain và permissionised hyperledger fabric.

Hạn chế trong bài tập này:

- Chưa phát triển kĩ được theo nghiệp vụ thực tế.
- Chưa phát triển kĩ được tính năng một cách an toàn và hoàn chỉnh.

## 2. KIẾN NGHỊ VÀ HƯỚNG PHÁT TRIỂN

Một số hướng nghiên cứu và phát triển của đề tài như sau:

- Bổ sung và hoàn thiện một số chức năng của hệ thống ...
- Đánh giá hiệu năng trên các môi trường khác nhau ...
- Kiểm thử các chức năng của chương trình ...
- Bổ sung các giải pháp bảo mật và an toàn cho hệ thống ...



## TÀI LIỆU THAM KHẢO

- [1] Readthedocs.io. (2022). *A Blockchain Platform for the Enterprise — hyperledger-fabricdocs main documentation*. [online] Available at: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/> [Accessed 11 Jun. 2022].
- [2] Pavan Adhav (2020). *Introduction to Course :Create Basic Network with Hyperledger Fabric v2.0*. YouTube. Available at: <https://www.youtube.com/watch?v=SJTdJt6N6Ow&list=PLSBNVhWU6KjW4qo1RlmR7cvvV8XIIlUb6> [Accessed 11 Jun. 2022].
- [3] Zand, Wu, & Morris, 2021. *Hands-on smart contract development with Hyperledger Fabric V2 : building enterprise blockchain applications*. Cambridge : O'Reilly, 2021.

## PHỤ LỤC

Mã nguồn của dự án có thể xem tại:

[https://github.com/damtien444/Auth\\_Network](https://github.com/damtien444/Auth_Network)

Để cài đặt lại network này có thể làm theo các bước tại phần readme:

1. Yêu cầu hệ thống:

- Ubuntu 18
- Docker, Docker-compose
- npm, node
- Go
- Hyperledger Fabric Binaries and docker images

\* Các thành phần trên cần được thêm vào system path của linux.

2. Thực hiện các bước ở phần KẾT QUẢ THỰC NGHIỆM để khởi động các chức năng của hệ thống.