

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA AN TOÀN THÔNG TIN**



**BÁO CÁO BÀI TẬP LỚN
HỌC PHẦN: CƠ SỞ AN TOÀN THÔNG TIN
MÃ HỌC PHẦN: INT1472**

**ĐỀ TÀI:
TÌM HIỂU VỀ CÔNG NGHỆ CHUỖI KHỐI
(BLOCKCHAIN), CÁC ỨNG DỤNG VÀ KHẢO
SÁT MỘT SỐ NỀN TẢNG CHẠY CÔNG NGHỆ
CHUỖI KHỐI**

Các sinh viên thực hiện

B22DCAT063 - Lê Tiến Dương

B22DCAT091 - Đỗ Anh Đức

B22DCAT271 - Vũ Hoàng Tuấn

B22DCAT183 - Cao Đức Mạnh

B22DCAT177 - Đoàn Thiên Long

Tên nhóm: 09

Tên lớp: CSATTT-INT1472-02

Giảng viên hướng dẫn: TS. Đinh Trường Duy

HÀ NỘI 2024

PHÂN CÔNG NHIỆM VỤ NHÓM THỰC HIỆN

TT	Công việc / Nhiệm vụ	SV thực hiện	Thời hạn hoàn thành
1	Tìm hiểu thành phần, nguyên lý hoạt động, phân loại Blockchain	Vũ Hoàng Tuấn	18/11/2024
2	Tìm hiểu cơ chế đồng thuận, đặc điểm nổi bật và demo cài đặt	Lê Tiến Dương	18/11/2024
3	Tìm hiểu ứng dụng của Blockchain	Đoàn Thiên Long	18/11/2024
4	Tìm hiểu về Bitcoin và khảo sát nền tảng	Cao Đức Mạnh	18/11/2024
5	Viết báo cáo, giới thiệu chung về Blockchain	Đỗ Anh Đức	20/11/2024

NHÓM THỰC HIỆN TỰ ĐÁNH GIÁ

TT	SV thực hiện	Thái độ tham gia	Mức hoàn thành CV	Kỹ năng giao tiếp	Kỹ năng hợp tác	Kỹ năng lãnh đạo
1	Vũ Hoàng Tuấn	5	5	4	5	3
2	Lê Tiến Dương	5	5	4	5	5
3	Đoàn Thiên Long	4	4	2	3	2
4	Cao Đức Mạnh	4	5	3	4	3
5	Đỗ Anh Đức	5	5	4	5	3

Ghi chú:

- Thái độ tham gia: Đánh giá điểm thái độ tham gia công việc chung của nhóm (từ 0: không tham gia, đến 5: chủ động, tích cực).
- Mức hoàn thành CV: Đánh giá điểm mức độ hoàn thành công việc được giao (từ 0: không hoàn thành, đến 5: hoàn thành xuất sắc).
- Kỹ năng giao tiếp: Đánh giá điểm khả năng tương tác, giao tiếp trong nhóm (từ 0: không hoặc giao tiếp rất yếu, đến 5: giao tiếp xuất sắc).
- Kỹ năng hợp tác: Đánh giá điểm khả năng hợp tác, hỗ trợ lẫn nhau, giải quyết mâu thuẫn, xung đột
- Kỹ năng lãnh đạo: Đánh giá điểm khả năng lãnh đạo (từ 0: không có khả năng lãnh đạo, đến 5: có khả năng lãnh đạo tốt, tổ chức và điều phối công việc trong nhóm hiệu quả).

MỤC LỤC

DANH MỤC CÁC HÌNH VẼ	5
DANH MỤC CÁC TỪ VIẾT TẮT.....	7
MỞ ĐẦU	8
CHƯƠNG 1. TỔNG QUAN VỀ BLOCKCHAIN	9
1.1. Giới thiệu chung về Blockchain	9
1.1.1. Giới thiệu về Blockchain	9
1.1.2. Lịch sử ra đời	10
1.1.3. Các phiên bản	10
1.2. Phân loại Blockchain	11
1.2.1. Blockchain công khai	11
1.2.2. Blockchain riêng tư.....	12
1.2.3. Blockchain liên hợp	12
1.3. Các thành phần của Blockchain	13
1.3.1. Khối	13
1.3.2. Nút	15
1.3.3. Chuỗi.....	16
1.3.4. Thợ đào	16
1.3.5. Giao dịch.....	17
1.3.6. Cơ chế đồng thuận	17
1.4. Một số mô hình đồng thuận phổ biến.....	17
1.4.1. Mô hình đồng thuận Bằng chứng công việc.....	17
1.4.2. Mô hình đồng thuận Bằng chứng cổ phần.....	19
1.4.3. Mô hình đồng thuận Round Robin	21
1.4.4. Mô hình đồng thuận Proof of Authority	21
1.5. Nguyên lý hoạt động của Blockchain.....	22
1.5.1. Nguyên lý mã hoá.....	22
1.5.2. Quy tắc cuốn sổ cái.....	24
1.5.3. Nguyên lý tạo khối.....	24
1.5.4. Thuật toán bảo mật Blockchain	25
1.6. Đặc điểm nổi bật của Blockchain.....	26
1.6.1. Tính bất biến	26

1.6.2.	Tính phân tán	26
1.6.3.	Tính phi tập trung	27
1.6.4.	Bảo mật	27
1.6.5.	Thanh toán nhanh chóng.....	27
CHƯƠNG 2.	ỨNG DỤNG CỦA BLOCKCHAIN.....	28
2.1.	BITCOIN – Ứng dụng đầu tiên của Blockchain	28
2.1.1.	Khái niệm Bitcoin.....	28
2.1.2.	Mối liên hệ giữa Blockchain và Bitcoin	28
2.1.3.	Tài khoản và Ví Bitcoin.....	28
2.1.4.	Tính bảo mật	29
2.1.5.	Tính riêng tư	29
2.1.6.	Tìm hiểu về đào Bitcoin	29
2.1.7.	Sự khác biệt giữa bitcoin và blockchain.....	30
2.2.	Ứng dụng của Blockchain trong đời sống.....	31
2.2.1.	Sản xuất	31
2.2.2.	Y tế.....	31
2.2.3.	Giáo dục.....	32
2.2.4.	Dịch vụ tài chính & ngân hàng	32
2.2.5.	Thương mại điện tử	32
2.2.6.	Truyền thông và viễn thông.....	33
CHƯƠNG 3.	MỘT SỐ NỀN TẢNG BLOCKCHAIN VÀ THỬ NGHIỆM	
BLOCKCHAIN TRÊN HYPERLEDGER FABRIC		34
3.1.	Một số nền tảng Blockchain	34
3.1.1.	Ethereum.....	34
3.1.2.	Hyperledger Fabric	34
3.1.3.	Binance Smart Chain (BSC).....	34
3.1.4.	Solana	35
3.2.	Thử nghiệm Blockchain trên môi trường Hyperledger Fabric.....	35
KẾT LUẬN.....		49
TÀI LIỆU THAM KHẢO		50

DANH MỤC CÁC HÌNH VẼ

Hình 1.1. Giới thiệu về Blockchain	9
Hình 1.2. Các phiên bản của Blockchain	10
Hình 1.3. Hợp đồng thông minh (Smart Contract).	11
Hình 1.4. Các thành phần chính của Blockchain	13
Hình 1.5. Cấu trúc của 1 block.....	14
Hình 1.6. Cây Merkle.....	15
Hình 1.7. Phân loại Node	15
Hình 1.8. Quá trình truy vấn dữ liệu từ full node	16
Hình 1.9. Mô hình Proof of Work (PoW).....	18
Hình 1.10. Cơ chế đồng thuận Proof of Stake (PoS).	20
Hình 1.11. Nguyên lí hoạt động hàm băm	22
Hình 1.12. Cơ chế liên kết giữa các khối.....	23
Hình 1.13. Nguyên lí hoạt động mã hoá RSA	23
Hình 1.14. Mô tả hoạt động của RSA	24
Hình 1.15. Cách thức thực hiện giao dịch của Blockchain.....	25
Hình 2.1. Logo Bitcoin	28
Hình 2.2. Dữ liệu trong khối Bitcoin	28
Hình 2.3. Độ khó trong việc đào Bitcoin qua các năm	30
Hình 2.4. Ứng dụng blockchain trong sản xuất	31
Hình 3.1. Logo Ethereum.....	34
Hình 3.2. Logo Hyperledger	34
Hình 3.5. Cài đặt các phần mềm	35
Hình 3.6. Thêm người dùng vào group docker và kiểm tra.....	36
Hình 3.7. Kiểm tra phiên bản Node Js với npm.....	36
Hình 3.8. Copy đoạn lệnh	37
Hình 3.9. Kiểm tra cài đặt Go và phiên bản cài đặt	37
Hình 3.10. Phiên bản của JavaScript mới cài đặt.....	37
Hình 3.11. Tạo 1 root project cho người lập trình Go	38
Hình 3.12. Tắt các container và artifact.....	38
Hình 3.13. Khởi tạo hệ thống mạng	38
Hình 3.14. Các container đang chạy	39
Hình 3.15. Câu lệnh tạo kênh.....	39
Hình 3.16. Tạo genesis block của kênh	39
Hình 3.17. Join peer của Org1 và Org2 vào kênh.....	40
Hình 3.18. Kiểm tra lại danh sách kênh mà các peer đã join.....	40
Hình 3.19. Anchor peer cho Org1 được thêm vào kênh	40
Hình 3.20. Anchor peer cho Org2 được thêm vào kênh	40
Hình 3.21. Câu lệnh khi tạo kênh thành công.....	40
Hình 3.22. Chaincode đang được triển khai lên kênh.....	41

Hình 3.23. Chaincode đã được cài đặt trên peer.org1.....	41
Hình 3.24. Chaincode đã được cài đặt trên peer.org2.....	41
Hình 3.25. Kiểm tra định nghĩa chaincode trên Org1	41
Hình 3.26. Kiểm tra định nghĩa chaincode trên Org2.....	41
Hình 3.27. Xác thực định nghĩa Chaincode trên kênh	42
Hình 3.28. Định nghĩa truy vấn chaincode trên 2 Org	42
Hình 3.29. Thiết lập môi trường để sử dụng biến Fabric binary	42
Hình 3.30. Thiết lập biến môi trường để tương tác với vai trò peer của Org1	43
Hình 3.31. InitLedger.....	43
Hình 3.32. CreateAsset	43
Hình 3.33. ReadAsset.....	43
Hình 3.34. UpdateAsset	43
Hình 3.35. DeleteAsset	43
Hình 3.36. AssetExists	44
Hình 3.37. TransferAsset	44
Hình 3.38. GetAllAssets	44
Hình 3.39. Khởi tạo sổ cái có tài sản	45
Hình 3.40. Truy vấn trạng thái của tài sản.....	45
Hình 3.41. Đọc trạng thái của tài sản asset6.....	46
Hình 3.42. Tài sản 7 đã được tạo thành công.....	46
Hình 3.43. Trạng thái của tài sản 7 đã được cập nhật.....	47
Hình 3.44. Xóa tài sản 6.....	47
Hình 3.45. Thay đổi người sở hữu tài sản 7.....	47
Hình 3.46. Kiểm tra tình trạng của sổ cái.	47
Hình 3.47. Thiết lập môi trường và truy vấn với vai trò peer của Org2.....	48

DANH MỤC CÁC TỪ VIẾT TẮT

Từ viết tắt	Thuật ngữ tiếng Anh/Giải thích	Thuật ngữ tiếng Việt/Giải thích
PoW	Proof of Work	Bằng chứng công việc
PoS	Proof of Stake	Bằng chứng cổ phần
DLT	Distributed Ledger Technology	Công nghệ sổ cái phân tán
PoA	Proof of Authority	Bằng chứng về quyền hạn
PoH	Proof of History	Bằng chứng về lịch sử
DeFi	Decentralized Finance	Tài chính phi tập trung
NFT	Non-Fungible Token	Tài sản kỹ thuật số không thể thay thế
Org	Organization	Tổ chức
CLI	Command Line Interface	Giao diện dòng lệnh
EVM	Ethereum Virtual Machine	Máy ảo Ethereum

MỞ ĐẦU

Trong thời đại công nghệ 4.0, khi các công nghệ số và Internet đang phát triển mạnh mẽ, blockchain đã trở thành một trong những xu hướng công nghệ đột phá, thu hút sự chú ý của các nhà phát triển, nhà đầu tư, cũng như các tổ chức và chính phủ trên toàn thế giới. Được biết đến như một hệ thống lưu trữ và xác minh dữ liệu phân tán, blockchain không chỉ thay đổi cách thức giao dịch tài chính mà còn mở ra tiềm năng ứng dụng trong nhiều lĩnh vực khác như chăm sóc sức khỏe, logistics, quản lý chuỗi cung ứng, và bảo mật thông tin.

Khái niệm blockchain lần đầu tiên xuất hiện vào năm 2008 trong tài liệu nghiên cứu của Satoshi Nakamoto, người sáng tạo ra đồng tiền điện tử Bitcoin. Tuy nhiên, blockchain không chỉ giới hạn trong việc hỗ trợ giao dịch tiền mã hóa mà còn có thể ứng dụng rộng rãi vào các ngành công nghiệp khác nhờ vào khả năng cung cấp một hệ thống lưu trữ dữ liệu an toàn, minh bạch và không thể thay đổi.

Báo cáo này sẽ tập trung vào việc giải thích khái niệm, nguyên lý hoạt động của blockchain, cùng với các ứng dụng tiềm năng và các thách thức trong việc triển khai công nghệ này trong thực tế. Đồng thời, báo cáo cũng sẽ phân tích những cơ hội và thách thức mà blockchain đem lại, giúp người đọc hiểu rõ hơn về tác động sâu rộng của công nghệ này trong kỷ nguyên số hiện nay.

Báo cáo này gồm 3 chương với nội dung chính như sau:

- Chương 1: Tổng quan về Blockchain bao gồm các nội dung khái quát về giới thiệu chung, lịch sử ra đời, các phiên bản, phân loại, các thành phần chính, nguyên lý hoạt động, các đặc điểm nổi bật và hạn chế của Blockchain.
- Chương 2: Tìm hiểu về Bitcoin và ứng dụng của Blockchain trong đời sống.
- Chương 3: Khảo sát các nền tảng và cài đặt thử nghiệm nền tảng Hyperledger Fabric.

CHƯƠNG 1. TỔNG QUAN VỀ BLOCKCHAIN

1.1. Giới thiệu chung về Blockchain

1.1.1. Giới thiệu về Blockchain

Blockchain là cuốn sổ cái kỹ thuật số chống giả mạo được triển khai theo mô hình phân tán (tức là không có kho lưu trữ trung tâm) và thường không cần một đơn vị đáng tin cậy chứng thực (như ngân hàng, công ty, chính phủ). Ở mức độ cơ bản, nó cho phép một cộng đồng người dùng ghi các giao dịch vào cuốn sổ cái chia sẻ, mà trong đó, với sự điều hành bình thường của mạng Blockchain thì không giao dịch nào có thể bị thay đổi sau khi xuất bản. Vào năm 2008, ý tưởng Blockchain được kết hợp với một vài công nghệ và khái niệm điện toán khác để tạo ra đồng tiền mã hóa hiện đại: tiền điện tử được bảo vệ bởi các cơ chế mật mã học thay vì nhờ vào bên chứng thực hoặc kho lưu trữ trung tâm.



Hình 1.1. Giới thiệu về Blockchain

Công nghệ này được biết đến rộng rãi vào năm 2009 với sự ra đời của mạng Bitcoin – một trong những đồng tiền mã hóa hiện đại đầu tiên. Ở hệ thống Bitcoin và các hệ thống tương tự, việc chuyển thông tin kỹ thuật số với đại diện là tiền điện tử diễn ra trong một hệ thống phân tán. Người dùng Bitcoin có thể ký chữ ký số và chuyển tài sản của mình sang người khác và Bitcoin ghi lại các giao dịch này công khai, cho phép những người tham gia mạng xác minh độc lập tính hợp lệ của giao dịch. Công nghệ Blockchain do đó được xem là giải pháp chung cho các đồng tiền mã hóa sau này.

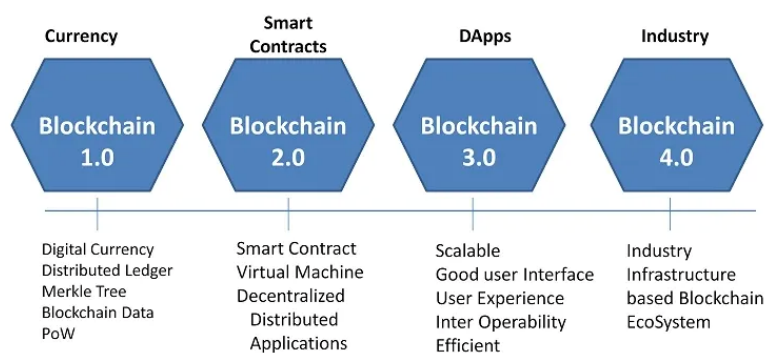
Blockchain có thể được định nghĩa như sau: “Blockchain là cuốn sổ cái kỹ thuật số của các giao dịch được ký bằng mật mã. Mỗi khối được liên kết mã hóa với khối trước nó sau khi được xác thực thì trải qua một quyết định đồng thuận. Khi một khối mới thêm vào, khối cũ hơn trở nên khó bị chỉnh sửa. Cuốn sổ cái sau đó được sao chép đến toàn bộ mạng và bất kỳ xung đột nào được giải quyết tự động thông qua các quy tắc được thiết lập.”

1.1.2. Lịch sử ra đời

Ý tưởng chính đứng sau công nghệ Blockchain này nổi lên vào cuối những năm 1980, đầu năm 1990. Vào năm 1989, Leslie Lamport đã phát triển giao thức Paxos. Năm 1990, ông có bài báo *The Part-Time Parliament* được gửi đến ACM Transaction on Computer Systems; bài báo được phát hành lần cuối vào năm 1998. Bài báo miêu tả một mô hình đồng thuận giúp đạt được thỏa thuận trên một kết quả của mạng lưới máy tính – nơi mà các máy tính hoặc bản thân mạng có thể không ổn định. Năm 1991, một chuỗi thông tin được ký đã được dùng như một cuốn sổ cái điện tử cho các tài liệu có chữ ký kỹ thuật số theo cách mà không dễ dàng để tài liệu được ký nào bị thay đổi. Các khái niệm này được kết hợp và áp dụng vào tiền điện tử năm 2008 và được miêu tả trong bài báo, *Bitcoin: A Peer to Peer Electronic Cash System*, được xuất bản giả bởi Satoshi Nakamoto, và sau đó năm 2009 với sự ra đời của tiền điện tử Bitcoin.

Việc sử dụng Blockchain cho phép Bitcoin được triển khai theo kiểu phân tán, như vậy không có người dùng đơn lẻ điều khiển được tiền điện tử và không có khuyết điểm tồn tại đơn lẻ. Lợi ích chính là cho phép các giao dịch trực tiếp giữa những người dùng mà không cần bên thứ ba đáng tin cậy. Nó cũng cho phép phát hành tiền mới theo cách được định nghĩa đến những người quản lý việc xuất bản các khối mới và duy trì bản sao của sổ cái, những người đó được gọi là *miners* ở Bitcoin. Bằng cách sử dụng cơ chế đồng thuận để duy trì và một cơ chế tự kiểm soát được tạo ra để đảm bảo rằng chỉ có các giao dịch và các khối hợp lệ mới được thêm vào Blockchain.

1.1.3. Các phiên bản



Hình 1.2. Các phiên bản của Blockchain

a. Blockchain 1.0: Tiền mã hoá (Cryptocurrency)

Mục tiêu của công nghệ này là cung cấp cách thức giao dịch, thanh toán qua không gian mạng trực tiếp và an toàn. Việc triển khai công nghệ sổ cái phân tán (Distributed Ledger Technology - DLT) đã dẫn đến ứng dụng đầu tiên là *Cryptocurrency* (tiền mã hóa). Bitcoin chính là ví dụ nổi bật nhất của nền tảng này.

b. Blockchain 2.0: Hợp đồng thông minh (Smart Contract)

Hợp đồng thông minh được phát triển dựa trên nền tảng Blockchain 2.0. Đây là các chương trình máy tính miễn phí thực thi tự động và kiểm tra các điều kiện được xác định trước đó như hỗ trợ, xác minh. Ví dụ điển hình cho nền tảng này là *Ethereum* – một giao thức cho phép người dùng tạo ra những hợp đồng thông minh thay thế những phiên bản truyền thống.



Hình 1.3. Hợp đồng thông minh (Smart Contract).

c. Blockchain 3.0: Ứng dụng phi tập trung (DApps)

Phiên bản này mở rộng ứng dụng của Blockchain ra ngoài tài chính, bao gồm y tế, bầu cử, quản lý tài nguyên, logistics và nhiều lĩnh vực khác. Mục tiêu của Blockchain 3.0 là tận dụng tính minh bạch và phi tập trung của công nghệ chuỗi khối để cải thiện hiệu quả và an toàn trong công việc.

d. Blockchain 4.0: Công nghệ chuỗi khối cho ngành công nghiệp (Blockchain For Industry)

Mục tiêu của Blockchain 4.0 là giải quyết toàn bộ vấn đề của ba thế hệ trước. Nền tảng này sẽ hỗ trợ doanh nghiệp giải thích các chiến lược và phương pháp để cải thiện hiệu quả kinh doanh.

1.2. Phân loại Blockchain

Blockchain có thể được phân loại theo nhiều tiêu chí khác nhau, trong đó ba tiêu chí chính là quyền truy cập, cơ chế đồng thuận và chức năng. Về cơ bản, Blockchain có hai loại: *Blockchain công khai (Public Blockchain)* và *Blockchain riêng tư (Private Blockchain)*. Tuy nhiên, cũng có những biến thể của hai loại Blockchain này, chúng gọi là *Blockchain liên hợp (Consortium Blockchain)*.

1.2.1. Blockchain công khai (Public Blockchain)

Blockchain công khai là nền tảng sổ cái phi tập trung mở rộng đến bất kỳ ai muốn xuất bản các khối mà không cần quyền chứng thực. Nền tảng Blockchain công khai thường là phần mềm mã nguồn mở, có sẵn miễn phí cho mọi người muốn tải xuống.

Vì ai cũng có quyền xuất bản các khối nên bất kỳ ai cũng có thể đọc Blockchain cũng như phát hành các giao dịch trên Blockchain (các giao dịch nằm trong các khối được xuất bản). Người dùng có ý đồ xấu có thể xuất bản các khối nhằm đánh sập hệ thống. Để ngăn chặn điều

này, mạng Blockchain công khai thường triển khai các thỏa thuận đa bên hay còn gọi là hệ thống “*đồng thuận*”, yêu cầu người dùng chi tiêu hoặc duy trì tài nguyên khi muốn xuất bản các khối. Đồng thời, hệ thống “*đồng thuận*” thường thúc đẩy các hành vi đúng đắn thông qua việc trao thưởng cho các nhà xuất bản các khối tuân thủ giao thức với loại tiền điện tử tương ứng.

* Ưu điểm:

- Phi tập trung: Mức độ phi tập trung cao giúp giảm rủi ro từ các điểm lỗi tập trung và tăng cường tính bảo mật.
- Minh bạch: Tất cả các giao dịch đều hiển thị công khai, nâng cao tính minh bạch và sự tin cậy.
- Bất biến: Một khi dữ liệu được ghi lại, nó không thể bị thay đổi hoặc xóa, đảm bảo bản ghi vĩnh viễn.

* Nhược điểm:

- Vấn đề mở rộng: Các blockchain công khai thường gặp khó khăn về khả năng mở rộng, với thông lượng giao dịch hạn chế và thời gian xử lý chậm.
- Tiêu thụ năng lượng: Một số cơ chế đồng thuận, như Bằng chứng công việc (Proof of Work PoW), yêu cầu sức mạnh tính toán và năng lượng lớn.

1.2.2. Blockchain riêng tư (Private Blockchain)

Đối với nền tảng Blockchain này người tham gia chỉ có quyền đọc dữ liệu, quyền ghi thuộc về bên thứ ba tin cậy. Một số trường hợp tổ chức có thể cho phép hoặc không cho phép người dùng đọc dữ liệu. Mọi thay đổi trên Blockchain đều do bên thứ ba toàn quyền quyết định.

Blockchain riêng tư có ưu thế là có thời gian xác nhận giao dịch khá nhanh vì quá trình xác thực giao dịch chỉ cần một lượng nhỏ thiết bị của tổ chức tin cậy. Ví dụ một dạng Blockchain riêng tư là đồng Ripple, hệ thống cho phép 20% các nút gặp lỗi, chỉ cần 80% còn lại vận hành ổn định là được.

* Ưu điểm:

- Hiệu suất và tốc độ: Xử lý giao dịch nhanh hơn và thông lượng cao hơn so với các blockchain công khai do số lượng nút ít hơn và yêu cầu đồng thuận thấp hơn.
- Quyền riêng tư: Giao dịch và dữ liệu chỉ hiển thị cho các thành viên được ủy quyền, tăng cường bảo mật thông tin.
- Kiểm soát: Kiểm soát tập trung cho phép quản lý dễ dàng hơn và tuân thủ các quy định một cách hiệu quả.

* Nhược điểm:

- Tính tập trung: Ít phi tập trung hơn các blockchain công khai, điều này có thể dẫn đến các điểm lỗi tập trung và giảm lợi ích bảo mật.
- Mức độ tin cậy: Yêu cầu các thành viên tin tưởng vào cơ quan trung tâm hoặc liên minh quản lý blockchain.
- Giới hạn minh bạch: Tính minh bạch giảm làm cho việc kiểm tra dữ liệu từ các bên bên ngoài trở nên khó khăn hơn.

1.2.3. Blockchain liên hợp (Consortium Blockchain)

Được biết tới là một dạng Blockchain riêng tư nhưng tích hợp thêm một số tính năng của Blockchain công khai. Blockchain liên hợp thường được sử dụng trong các môi trường doanh

ng nghiệp để đảm bảo sự linh hoạt và kiểm soát cao đối với quyền truy cập và quyền thực hiện giao dịch.

Ví dụ: Facebook sở hữu đồng Libra. Đây là đồng tiền Crypto và hệ sinh thái Blockchain có quy trình vận hành kỹ lưỡng và thiết kế mới hơn so với các đồng tiền Crypto trước đó.

* Ưu điểm:

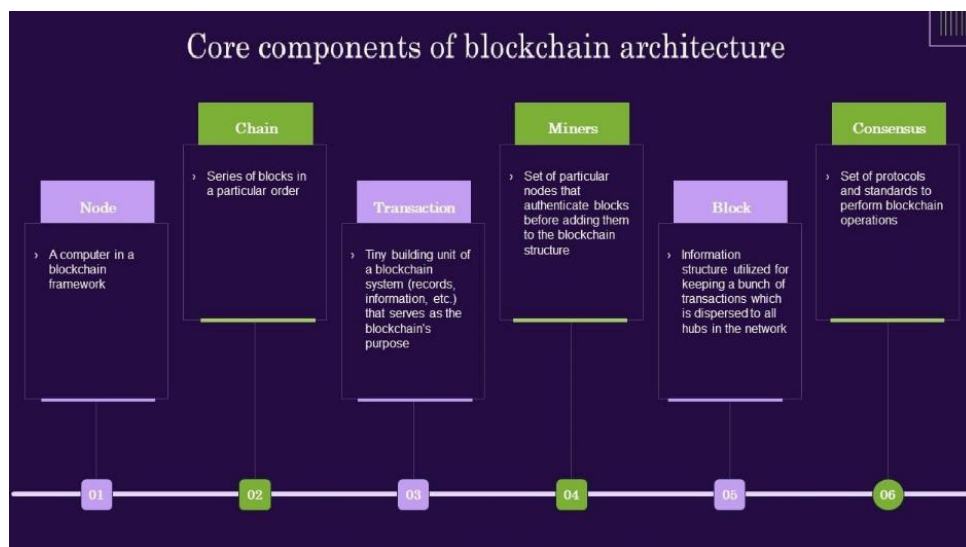
- Hiệu suất: Hiệu suất và hiệu quả cao hơn so với các blockchain công khai nhờ số lượng nút ít hơn và các cơ chế đồng thuận được tối ưu hóa.
- Quản lý chia sẻ: Quyền quản lý được chia sẻ giữa các thành viên trong liên minh, giúp tăng cường sự tin cậy và hợp tác.
- Quyền riêng tư và bảo mật: Tăng cường quyền riêng tư và bảo mật so với blockchain công khai, vì quyền truy cập được giới hạn.

* Nhược điểm:

- Quản trị phức tạp: Quyết định có thể trở nên phức tạp do có nhiều bên liên quan với lợi ích có thể xung đột.
- Giảm phí tập trung: Dù phí tập trung hơn blockchain riêng tư, nhưng số lượng thành viên hạn chế vẫn làm giảm một số lợi ích của tính phi tập trung.
- Khả năng tương tác: Gặp thách thức khi tích hợp với các mạng hoặc hệ thống blockchain khác.

1.3. Các thành phần của Blockchain

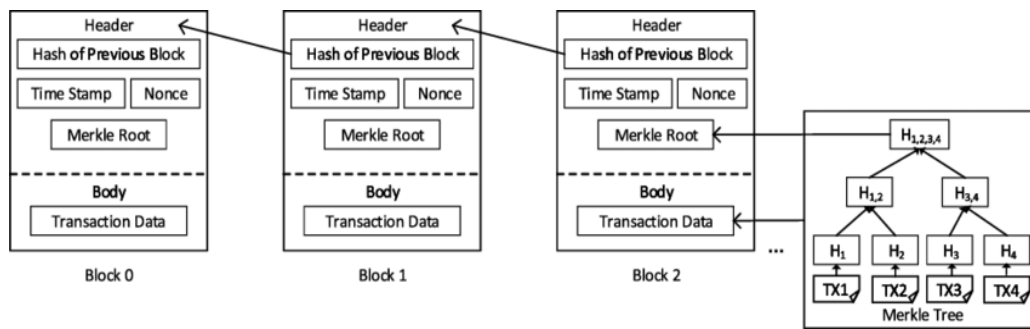
Blockchain gồm 6 thành phần chính:



Hình 1.4. Các thành phần chính của Blockchain

1.3.1. Khối (Block)

Các khối (blocks) là các tệp được lưu trữ bởi một Blockchain, nơi dữ liệu giao dịch được ghi lại vĩnh viễn. Một khối ghi lại một phần hoặc tất cả các giao dịch gần đây chưa được xác thực bởi mạng lưới. Khi dữ liệu được xác thực, khối sẽ được đóng lại. Sau đó, một khối mới sẽ được tạo ra để các giao dịch mới có thể được nhập vào và xác thực.



Hình 1.5. Cấu trúc của 1 block

Mỗi khối gồm 2 thành phần chính:

i. Tiêu đề khối (Block header)

Tiêu đề khối (Block header) là một thành phần quan trọng của mỗi khối trong blockchain. Nó giúp nhận diện và bảo mật một khối trong toàn bộ chuỗi blockchain. Tiêu đề khối được băm lặp đi lặp lại như một phần của cơ chế **Proof of Work (PoW)**, được sử dụng để xác thực khối và tạo phần thưởng khai thác cho các thợ mỏ (miners) đóng góp tài nguyên tính toán của họ.

Thành phần của tiêu đề khối:

Hash của block: Là một mã định danh duy nhất (mã băm) được tạo từ nội dung của block. Bất kỳ thay đổi nhỏ nào trong block cũng sẽ làm thay đổi mã băm này.

Hash của block trước: Giúp liên kết block hiện tại với block trước đó, tạo thành chuỗi (chain). Đây là lý do Blockchain có tính bất biến và bảo mật cao.

Dấu thời gian (Time Stamp): Đây là hệ thống xác minh dữ liệu vào khối và chỉ định thời gian hoặc ngày tạo cho các tài liệu kỹ thuật số. Dấu thời gian là một chuỗi ký tự xác định duy nhất tài liệu hoặc sự kiện và cho biết thời điểm tài liệu hoặc sự kiện đó được tạo.

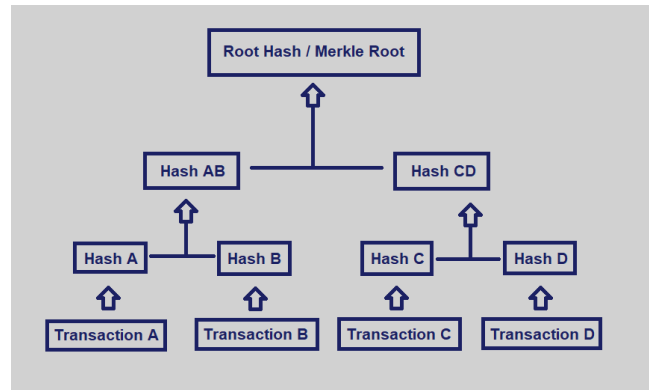
Nonce: Một số nonce chỉ được sử dụng một lần. Nó là một phần quan trọng trong bằng chứng công việc (Proof of Work – PoW) trong khối. Nó được so sánh với mục tiêu hiện tại để xem nó có nhỏ hơn hoặc bằng mục tiêu hiện tại hay không. Những người đào, kiểm tra và tính toán nhiều nonce mỗi giây cho đến khi họ tìm ra nonce có giá trị hợp lệ.

Merkle Root: Đây là một loại cấu trúc dữ liệu để kết nối các khối dữ liệu khác nhau. Cây Merkle (Merkle Tree) lưu trữ tất cả các giao dịch trong một khối bằng cách tạo ra một dấu vân tay kỹ thuật số của toàn bộ giao dịch. Nó cho phép người dùng xác minh xem một giao dịch có thể được bao gồm trong khối hay không.

ii. Thân khối (Block body)

Phần thân của khối (Block body) có thể hiểu được là khoang chứa hàng của một chiếc xe tải. Nó chứa tất cả các giao dịch được xác nhận với khối. Mỗi giao dịch thường bao gồm các thông tin về địa chỉ người gửi, địa chỉ người nhận, số lượng tài sản, phí giao dịch và chữ ký số. Các giao dịch trong một khối không chỉ trong một danh sách, mà trong một thứ gọi là cây Merkle (Merkle Tree).

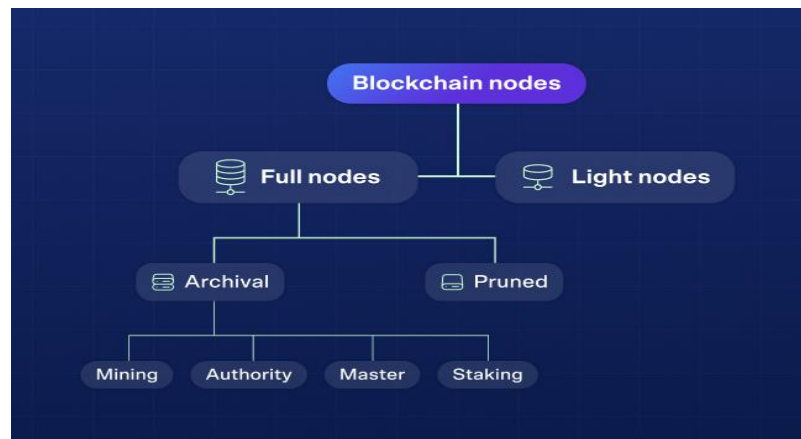
Cây Merkle được đặt tên theo nhà toán học Ralph Merkle. Đầu tiên, dữ liệu sẽ được băm. Sau đó, các mã băm được sẽ được băm một lần nữa và hợp nhất lại. Cuối cùng, Cây Merkle được hợp nhất thành một hàm băm duy nhất. Hàm băm cuối cùng này cũng được gọi là hàm băm gốc – gốc của cây. Nó đại diện cho tất cả các thông tin của nó trên “những chiếc lá” (giao dịch cá nhân) và “các cành cây” (các mã băm của lá) trong một chuỗi tương đối ngắn.



Hình 1.6. Cây Merkle

1.3.2. Nút (Node)

Một nút Blockchain là một phần quan trọng của mạng Blockchain, chịu trách nhiệm duy trì sổ cái phân tán bằng cách xử lý các giao dịch và xác thực các khối mới. Các nút này là duy nhất và được xác định thông qua một mã định danh duy nhất được gắn vào chúng. Mạng lưới càng có nhiều nút thì càng trở nên phi tập trung và an toàn. Có nhiều loại nút Blockchain, tuy nhiên có 2 loại chính là *full node* và *light node*.



Hình 1.7. Phân loại Node

i. Full node

Full node hoạt động như một máy chủ (server) trong mạng lưới phi tập trung. Nhiệm vụ chính của full node bao gồm duy trì sự đồng thuận giữa các node khác và xác minh các giao dịch. Full node cũng lưu trữ một bản sao của sổ cái blockchain, do đó nó an toàn hơn và cho phép các chức năng nâng cao như quyền biểu quyết cho các đề xuất trong blockchain.

Full node có thể chia thành các loại như:

- Full node lưu trữ (Archival full node): Là các node lưu trữ toàn bộ lịch sử giao dịch của blockchain, từ khối đầu tiên đến khối mới nhất. Những node này thực thi các quy tắc đồng thuận và xác thực các giao dịch. Blockchain có thể được xây dựng lại từ một **full node lưu trữ** nếu cần thiết, vì nó chứa đầy đủ dữ liệu giao dịch lịch sử. Full node lưu trữ có thể bao gồm các loại node như node khai thác (Mining node), node quản trị (Authority node), node cược (Staking node) và master node.

- Full node cắt giảm (Pruned full node): Là phiên bản rút gọn của **full node lưu trữ**. Chúng tải toàn bộ blockchain, xác thực nó, nhưng chỉ lưu trữ các khối gần đây trong một giới hạn kích

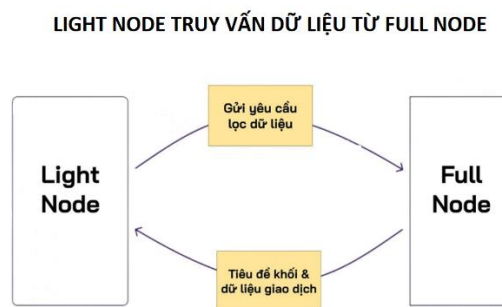
thước nhất định và xóa bỏ dữ liệu cũ. Việc này giúp giảm yêu cầu về dung lượng lưu trữ trong khi vẫn thực hiện được các chức năng như các node full khác.

Một số đặc điểm chính của một full node:

- Lưu trữ đầy đủ dữ liệu blockchain.
- Tham gia xác thực khối, xác minh tất cả các khối và trạng thái.
- Tất cả các trạng thái có thể bắt nguồn từ full node.
- Cung cấp dữ liệu theo yêu cầu cho các light node.

ii. Light node

Light node, hay Simple Payment Verification (SPV) node, tức node xác minh thanh toán đơn giản. Light node có thể được xem là phiên bản rút gọn của Full node. Thay vì lưu trữ toàn bộ blockchain, nó dựa vào các full node khác để xác thực giao dịch và truy vấn thông tin để tạo các tiêu đề khối (block header).



Hình 1.8. Quá trình truy vấn dữ liệu từ full node

Việc chạy light node giúp tiết kiệm bộ nhớ và không yêu cầu nhiều tài nguyên, tuy nhiên, khả năng xác thực giao dịch, tham gia quá trình đồng thuận và bảo mật blockchain của nó cũng sẽ bị hạn chế so với full node. Vì vậy, light node thường được sử dụng trong ứng dụng yêu cầu tài nguyên thấp như ví tiền điện tử.

1.3.3. Chuỗi (Chain)

Trong **blockchain**, **chuỗi (chain)** là một phần cốt lõi của cấu trúc hệ thống, nơi các khối (blocks) được liên kết với nhau theo một thứ tự nhất định, tạo thành một chuỗi liên tục, không thể thay đổi. Mỗi khối trong chuỗi chứa thông tin về các giao dịch và một tham chiếu (hash) tới khối trước đó, giúp xác định và bảo vệ tính toàn vẹn của dữ liệu. Cấu trúc chuỗi này không chỉ đảm bảo sự nhất quán của các dữ liệu trong blockchain mà còn góp phần bảo mật mạng lưới và ngăn chặn hành vi gian lận như **double spending**.

1.3.4. Thợ đào (Miners)

Thợ đào (miners) là những người tham gia vào quá trình khai thác blockchain, xác thực giao dịch và tạo ra các khối mới. Công việc của miners là **xác thực giao dịch, tạo khối mới**, tham gia vào **cơ chế đồng thuận** của blockchain, và duy trì **bảo mật** mạng lưới. Trong các blockchain như **Bitcoin**, **Ethereum** (trước khi chuyển sang Proof of Stake), miners hoạt động trên cơ chế **Proof of Work (PoW)**. Cơ chế này yêu cầu miners phải giải quyết một bài toán mã hóa (bài toán băm) để chứng minh rằng họ đã bỏ ra một lượng công sức tính toán đáng kể. Việc này gọi

là "Proof of Work" (chứng minh công việc). Sau khi giải được bài toán, miner có quyền tạo ra một khối mới và nhận phần thưởng. **Phần thưởng** cho thợ đào bao gồm **phần thưởng khối** và **phí giao dịch**.

- **Phần thưởng khối** (block reward): Đây là số tiền mà miners nhận được khi họ thành công trong việc tạo ra và xác nhận một khối mới.

- **Phí giao dịch**: Ngoài phần thưởng khối, miners còn nhận được phí giao dịch từ người dùng khi họ thực hiện các giao dịch trên mạng. Những khoản phí này cũng được thêm vào phần thưởng của thợ đào.

1.3.5. Giao dịch (Transaction)

Một giao dịch đại diện một sự tương tác giữa các bên tham gia. Với các đồng tiền mã hóa, một giao dịch đại diện cho việc chuyển tiền giữa những người dùng Blockchain. Đối với môi trường kinh doanh, một giao dịch có thể là một cách ghi lại các hoạt động xảy ra trên tài sản kỹ thuật số hoặc vật lý.

Mỗi một khối trong Blockchain có thể không chứa hoặc chứa nhiều giao dịch. Trong một vài Blockchain, việc cung cấp liên tục các khối mới (kể cả với giao dịch không) là quan trọng cho việc duy trì bảo mật mạng Blockchain bởi nó ngăn chặn những người dùng xấu xa khỏi “bắt được” và tạo một chuỗi khác thay thế. Một người dùng gửi thông tin đến mạng Blockchain, thông tin được gửi có thể bao gồm địa chỉ người gửi (hoặc số nhận dạng có liên quan khác), khóa công khai của người gửi, chữ ký số, đầu vào và đầu ra giao dịch.

Các giao dịch thường được ký bằng khóa riêng của người gửi và có thể được xác minh bất kỳ lúc nào bằng khóa chung được liên kết.

1.3.6. Cơ chế đồng thuận (Consensus)

Là một quá trình mà các thành viên trong mạng lưới (các nút) đạt được sự thống nhất về một bộ dữ liệu chung và đồng bộ, đặc biệt là về thứ tự các giao dịch và khối trong blockchain. Cơ chế đồng thuận đảm bảo rằng tất cả các bản sao của blockchain trong mạng lưới là giống nhau và các giao dịch mới được thêm vào chuỗi đều hợp lệ. Vì blockchain là một hệ thống phi tập trung, không có một cơ quan trung ương để xác nhận và kiểm tra tính hợp pháp của giao dịch, vì vậy cơ chế đồng thuận rất quan trọng để duy trì sự minh bạch, bảo mật và tin cậy của mạng lưới.

1.4. Một số mô hình đồng thuận phổ biến.

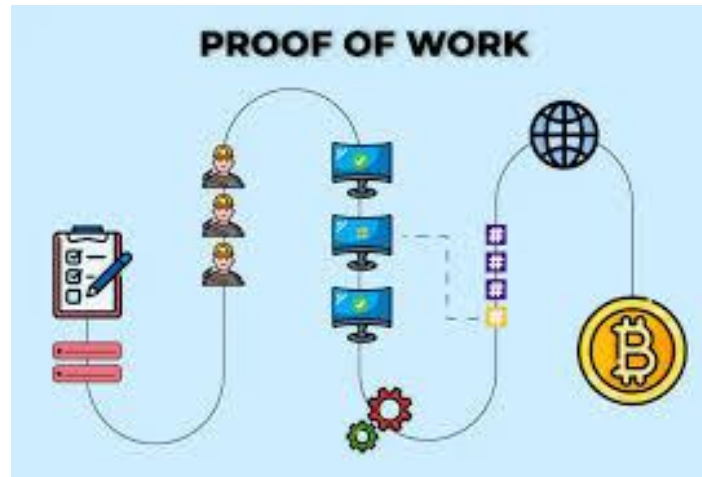
1.4.1. Mô hình đồng thuận Bằng chứng công việc (Proof of Work – PoW)

Mô hình **Bằng chứng công việc (Proof of Work - PoW)** là một cơ chế đồng thuận được sử dụng bởi các mạng blockchain, trong đó quá trình thêm một khối mới vào chuỗi dựa trên việc giải quyết một câu đố tính toán khó khăn. Người dùng nào giải được câu đố này trước tiên sẽ có quyền xuất bản khối tiếp theo. "Bằng chứng" cho công việc đã thực hiện chính là lời giải của câu đố, được thiết kế sao cho khó tìm nhưng dễ xác minh. Điều này đảm bảo rằng các nút đầy đủ khác trong mạng có thể dễ dàng xác thực đề xuất khối.

Trong PoW, một phương pháp phổ biến là yêu cầu giá trị băm (hash) của tiêu đề khối nhỏ hơn một giá trị mục tiêu cụ thể. Để đạt được điều này, các nút liên tục thực hiện các thay đổi nhỏ (chẳng hạn thay đổi giá trị **nonce**) trong tiêu đề khối nhằm tìm một giá trị băm đáp ứng yêu

cầu mục tiêu. Mỗi lần thử nghiệm yêu cầu tính toán giá trị băm của toàn bộ tiêu đề khối, điều này tiêu tốn rất nhiều tài nguyên tính toán. Độ khó của câu đố có thể được điều chỉnh theo thời gian bằng cách thay đổi giá trị mục tiêu, nhằm kiểm soát tần suất các khối được xuất bản.

Thông thường, độ khó được điều chỉnh bằng cách thay đổi số lượng số không đứng đầu trong giá trị băm. Tăng số lượng số không đứng đầu làm tăng độ khó, vì ít lời giải đáp ứng được mục tiêu hơn; ngược lại, giảm số lượng số không đứng đầu làm giảm độ khó, vì có nhiều lời giải hợp lệ hơn. Việc điều chỉnh này giúp đảm bảo tốc độ tạo khối ổn định, bất chấp sự gia tăng về năng lực tính toán và số lượng nút tham gia.



Hình 1.9. Mô hình Proof of Work (PoW)

Mục đích của việc điều chỉnh độ khó là để ngăn chặn bất kỳ thực thể nào thống trị quá trình sản xuất khối, duy trì tính phi tập trung và an toàn của mạng. Tuy nhiên, nỗ lực tính toán để giải các câu đố tiêu tốn lượng lớn tài nguyên. Vì lý do này, các hoạt động khai thác thường được đặt tại các khu vực có nguồn cung điện rẻ dồi dào.

Một đặc điểm quan trọng khác của PoW là công việc thực hiện trên một câu đố không ảnh hưởng đến khả năng giải các câu đố hiện tại hoặc tương lai, vì các câu đố này độc lập với nhau. Điều này có nghĩa là khi một người dùng nhận được một khối hoàn chỉnh và hợp lệ từ người khác, họ được khuyến khích từ bỏ công việc hiện tại và bắt đầu xây dựng từ khối mới nhận được, vì họ biết rằng các nút khác cũng sẽ xây dựng dựa trên khối đó.

Ví dụ minh họa:

Xét một câu đố sử dụng thuật toán SHA-256, trong đó máy tính phải tìm giá trị băm (hash) đáp ứng tiêu chí mục tiêu (được gọi là độ khó):

$$SHA256("blockchain" + Nonce) = \text{Giá trị băm bắt đầu bằng "000000"}$$

Trong ví dụ này, chuỗi văn bản "blockchain" được nối với một giá trị nonce, sau đó tính toán giá trị băm. Giá trị nonce được sử dụng chỉ bao gồm các số. Đây là một câu đố tương đối đơn giản, và kết quả mẫu như sau:

- $SHA256("blockchain0") =$
0xbd4824d8ee63fc82392a6441444166d22ed84eaa6dab11d4923075975acab938
(chưa giải được)
- $SHA256("blockchain1") =$

0xdb0b9c1cb5e9c680dfff7482f1a8efad0e786f41b6b89a758fb26d9e223e0a10
(chưa giải được)

• ...

• SHA256("blockchain10730895") =
0x000000ca1415e0bec568f6f605fcc83d18cac7a4e6c219a957c10c6879d67587
(đã giải được)

Để giải câu đố này, phải thử 10.730.896 giá trị (mất 54 giây trên phần cứng tương đối cũ, bắt đầu từ 0 và thử từng giá trị một).

Tăng độ khó: Nếu tăng độ khó bằng cách thêm một số 0 ở đầu giá trị mục tiêu ("0000000"), cùng phần cứng trên phải thử 934.224.175 giá trị để giải quyết câu đố (mất 1 giờ, 18 phút, 12 giây):

• SHA256("blockchain934224174") =
0x0000000e2ae7e4240df80692b7e586ea7a977eacbd031819d0e603257edb3a81

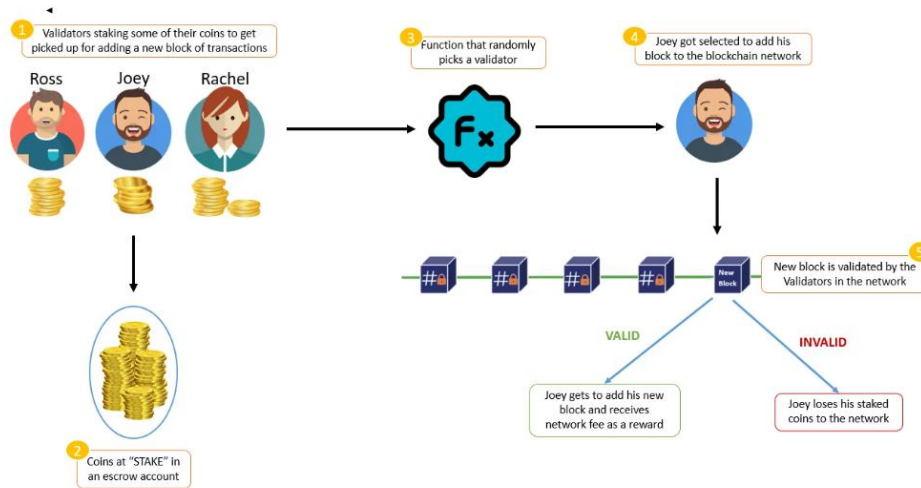
⇒ Điều này minh họa rằng độ khó tăng lên đáng kể khi yêu cầu thêm mỗi số 0 đứng đầu.

Hiện tại không có cách tắt nào được biết đến để giải quyết quá trình này; các nút xuất bản phải tiêu tốn công sức tính toán, thời gian và tài nguyên để tìm giá trị **nonce** phù hợp với mục tiêu. Thông thường, các nút xuất bản cố gắng giải câu đố tính toán khó khăn này để nhận phần thưởng nào đó (thường dưới dạng tiền mã hóa được cung cấp bởi mạng blockchain). Việc khen thưởng cho quá trình mở rộng và duy trì blockchain được gọi là **hệ thống phần thưởng** hoặc **mô hình khuyến khích**.

Khi một nút xuất bản hoàn thành công việc này, nó sẽ gửi khối của mình kèm giá trị nonce hợp lệ đến các nút đầy đủ (full nodes) trong mạng blockchain. Các nút đầy đủ nhận được sẽ kiểm tra xem khối mới có đáp ứng yêu cầu của câu đố hay không, sau đó thêm khối này vào bản sao blockchain của mình và gửi lại khối đến các nút ngang hàng khác. Theo cách này, khối mới được phân phối nhanh chóng trong mạng lưới các nút tham gia.

1.4.2. Mô hình đồng thuận Bằng chứng cổ phần (Proof of Stake – PoS)

Mô hình **Proof of Stake (PoS)** dựa trên ý tưởng rằng càng nhiều cổ phần (stake) mà người dùng đầu tư vào hệ thống, họ càng có khả năng mong muốn hệ thống thành công và ít có khả năng muốn làm hỏng nó. Cổ phần thường là một lượng tiền điện tử mà người dùng mạng blockchain đã đầu tư vào hệ thống (thông qua các phương thức khác nhau, như khóa nó thông qua một loại giao dịch đặc biệt, gửi nó đến một địa chỉ cụ thể, hoặc giữ nó trong phần mềm ví đặc biệt). Sau khi đã đặt cược (staked), tiền điện tử thường không thể chi tiêu được nữa.



Hình 1.10. Cơ chế đồng thuận Proof of Stake (PoS).

Các mạng blockchain sử dụng **Proof of Stake** sử dụng lượng cổ phần mà người dùng có để xác định việc xuất bản các khối mới. Do đó, khả năng một người dùng mạng blockchain xuất bản một khối mới sẽ phụ thuộc vào tỷ lệ cổ phần của họ so với tổng số lượng tiền điện tử được đặt cược trong toàn bộ mạng blockchain. Với mô hình đồng thuận này, không cần phải thực hiện các phép toán tính toán tài nguyên cao (bao gồm thời gian, điện năng và sức mạnh xử lý) như trong **Proof of Work**. Vì mô hình đồng thuận này sử dụng ít tài nguyên hơn, một số mạng blockchain đã quyết định từ bỏ phần thưởng tạo khối; thay vào đó, hệ thống được thiết kế sao cho tất cả tiền điện tử đã được phân phối giữa các người dùng thay vì tạo ra tiền điện tử mới với tốc độ liên tục. Trong các hệ thống như vậy, phần thưởng cho việc xuất bản khối thường là thu nhập từ các khoản phí giao dịch mà người dùng cung cấp.

Các phương thức sử dụng cổ phần có thể thay đổi. Dưới đây là bốn phương pháp phổ biến:

- 1. Lựa chọn ngẫu nhiên người dùng có cổ phần:** Mạng blockchain sẽ xem xét tất cả người dùng có cổ phần và chọn một trong số họ dựa trên tỷ lệ cổ phần của họ so với tổng lượng cổ phần của hệ thống. Ví dụ, nếu người dùng có 42% cổ phần trong toàn bộ mạng blockchain, họ sẽ được chọn 42% thời gian, những người có 1% sẽ được chọn 1% thời gian.
- 2. Hệ thống bỏ phiếu nhiều vòng (Byzantine fault tolerance proof of stake):** Hệ thống này phức tạp hơn, nơi mạng blockchain sẽ chọn nhiều người dùng có cổ phần để tạo ra các khối đề xuất. Sau đó, tất cả người dùng có cổ phần sẽ bỏ phiếu cho các khối đề xuất. Có thể có nhiều vòng bỏ phiếu trước khi một khối mới được quyết định. Phương pháp này cho phép tất cả người dùng có cổ phần có tiếng nói trong quá trình chọn lựa khối cho mỗi khối mới.
- 3. Hệ thống tuổi coin (coin age proof of stake):** Cổ phần tiền điện tử có đặc tính tuổi. Sau một khoảng thời gian nhất định (ví dụ 30 ngày), tiền điện tử đã đặt cược có thể được tính vào khả năng người sở hữu được chọn để xuất bản khối tiếp theo. Sau khi sử dụng, tuổi của tiền điện tử được đặt cược sẽ được đặt lại và không thể sử dụng lại cho đến khi hết thời gian quy định. Phương pháp này cho phép những người dùng có cổ phần nhiều hơn có thể xuất bản nhiều khối hơn nhưng không thể chiếm ưu thế hoàn toàn, vì họ có một thời gian làm mát được gắn vào mỗi đồng coin được tính cho việc tạo khối. Các đồng coin cũ hơn và nhóm coin lớn hơn sẽ tăng khả

năng được chọn để xuất bản khối tiếp theo. Để ngăn chặn việc người dùng tích trữ tiền điện tử đã có tuổi, thường sẽ có một giới hạn tối đa đối với khả năng chiến thắng.

4. Hệ thống đại biểu (delegate system): Người dùng bỏ phiếu cho các nút để trở thành các nút xuất bản – nghĩa là tạo khối thay mặt cho họ. Quyền bỏ phiếu của người dùng trong mạng blockchain được liên kết với cổ phần của họ, vì vậy cổ phần lớn hơn sẽ có quyền bỏ phiếu có trọng số hơn. Các nút nhận được nhiều phiếu bầu nhất sẽ trở thành nút xuất bản và có thể xác minh và xuất bản các khối. Người dùng mạng blockchain cũng có thể bỏ phiếu chống lại một nút xuất bản đã được thiết lập, nhằm loại bỏ họ khỏi nhóm các nút xuất bản. Ngoài ra, người dùng mạng blockchain cũng bỏ phiếu cho các đại biểu, những người tham gia vào việc quản trị blockchain. Các đại biểu sẽ đề xuất các thay đổi và cải tiến, và người dùng mạng blockchain sẽ bỏ phiếu cho chúng.

Dưới hệ thống ***Proof of Stake***, người “giàu” có thể dễ dàng đặt cược nhiều tài sản kỹ thuật số hơn, kiếm thêm tài sản kỹ thuật số cho bản thân; tuy nhiên, để có được phần lớn tài sản kỹ thuật số trong một hệ thống như vậy đòi hỏi chi phí rất cao trong việc kiểm soát.

1.4.3. Mô hình đồng thuận Round Robin

Mô hình ***Round Robin*** là một mô hình đồng thuận được sử dụng bởi một số mạng blockchain có quyền hạn. Trong mô hình này, các nút (nodes) sẽ thay phiên nhau tạo các khối. Mô hình ***Round Robin*** có lịch sử lâu dài trong kiến trúc hệ thống phân tán. Để xử lý các tình huống khi một nút xuất bản không có sẵn để xuất bản khối trong lượt của nó, các hệ thống này có thể bao gồm một giới hạn thời gian để các nút có sẵn có thể xuất bản các khối, nhằm đảm bảo rằng các nút không có sẵn sẽ không gây tắc nghẽn trong quá trình xuất bản khối.

Mô hình này đảm bảo rằng không có một nút nào tạo ra phần lớn các khối. Nó có lợi thế từ một cách tiếp cận đơn giản, không có các câu đố mật mã học phức tạp và yêu cầu năng lượng thấp. Tuy nhiên, vì mô hình này yêu cầu sự tin cậy giữa các nút, nó không hoạt động tốt trong các mạng blockchain không có quyền hạn mà phần lớn các đồng tiền điện tử sử dụng. Lý do là vì các nút độc hại có thể liên tục thêm các nút khác vào hệ thống để tăng cơ hội xuất bản khối mới của chúng. Trong trường hợp xấu nhất, họ có thể lợi dụng điều này để làm suy yếu hoạt động chính xác của mạng blockchain.

1.4.4. Mô hình đồng thuận Proof of Authority (PoA)

Mô hình ***Proof of Authority (PoA)***, còn được gọi là ***Proof of Identity***, dựa vào sự tin tưởng một phần vào các nút xuất bản thông qua mối liên kết của chúng với các danh tính thế giới thực. Các nút xuất bản phải chứng minh và xác minh danh tính của họ trong mạng blockchain (ví dụ, các tài liệu nhận dạng đã được xác minh và công chứng và đưa vào blockchain). Ý tưởng ở đây là các nút xuất bản đang “đặt cược” danh tính/uy tín của chúng để xuất bản các khối mới.

Người dùng mạng blockchain sẽ trực tiếp ảnh hưởng đến uy tín của nút xuất bản dựa trên hành vi của nút đó. Các nút xuất bản có thể mất uy tín nếu hành động của chúng không được người dùng mạng blockchain đồng thuận, cũng như có thể gia tăng uy tín nếu hành động của chúng phù hợp với sự đồng thuận của người dùng mạng blockchain. Uy tín thấp sẽ làm giảm khả năng xuất bản khối của nút đó. Vì vậy, các nút xuất bản sẽ có động lực để duy trì uy tín cao.

Thuật toán này chỉ áp dụng cho các mạng blockchain có quyền hạn (permissioned blockchain) với mức độ tin tưởng cao.

1.5. Nguyên lý hoạt động của Blockchain

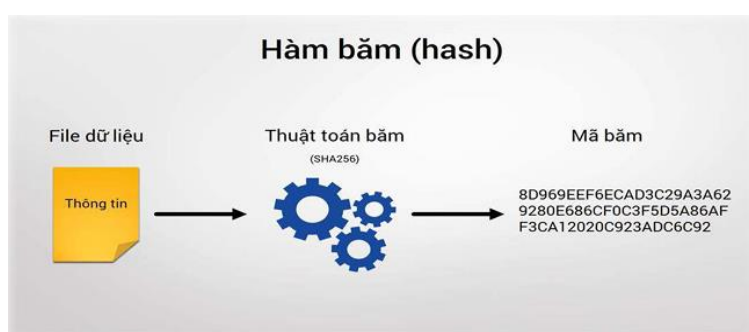
Blockchain hoạt động như cuốn sổ cái ghi chép mọi giao dịch trong mạng lưới, được bảo vệ bởi hệ thống bảo mật tiên tiến và minh bạch cho tất cả mọi người. Mỗi nút trên Blockchain đều sở hữu quyền truy cập vào tất cả các thông tin dữ liệu trong mỗi block. Cụ thể, hệ thống chuỗi khối sẽ hoạt động dựa trên 4 nguyên lý là: *mã hóa, quy tắc sổ cái, nguyên lý tạo khối và bảo mật thông tin*.

1.5.1. Nguyên lý mã hoá

Hai nguyên lý chính đó là *hàm băm* và *mã hóa RSA*. Hệ thống Blockchain được thiết kế theo cách không yêu cầu sự tin cậy và bảo đảm bởi độ tin cậy có được thông qua các hàm mã hóa toán học đặc biệt. Đặc biệt đó là bảo mật duy nhất gồm các khóa công khai và khóa riêng tư trong mã hóa RSA. Nếu mã hóa bằng khóa công khai thì chỉ có chủ sở hữu của khóa riêng tư trùng với khóa công khai mới có thể giải mã và nắm được nội dung dữ liệu có trong đó. Ngược lại, khi không có bất kỳ trường hợp nào có thể mở được khóa. Hàm băm đóng vai trò nối tiếp liên kết các khối lại với nhau thành một chuỗi. Ngoài ra để tối ưu về mặt bảo mật thì Blockchain cũng áp dụng cả *nguyên lý PoW*.

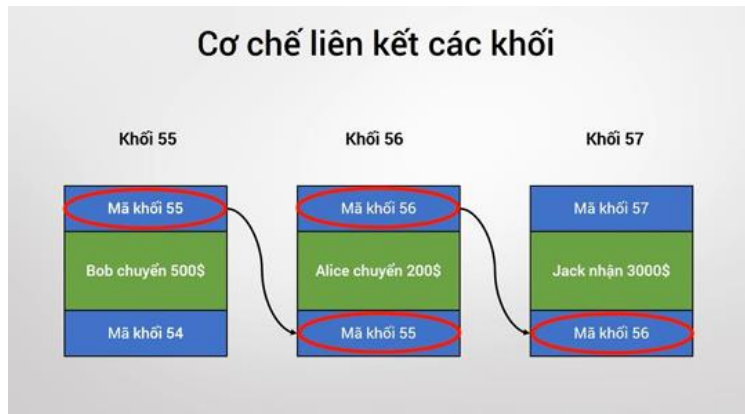
a. Hàm băm

Trong ngành mật mã học, một hàm băm mật mã là một hàm băm với một số tính năng bảo mật nhất định để phù hợp việc sử dụng trong nhiều ứng dụng bảo mật thông tin đa dạng, chẳng hạn như chứng thực (authentication) và kiểm tra tính nguyên vẹn của thông điệp (message integrity). Một hàm băm nhận đầu vào là một chuỗi ký tự dài (hay thông điệp) có độ dài tùy ý và tạo ra kết quả là một chuỗi ký tự có độ dài cố định, đôi khi được gọi là tóm tắt thông điệp (message digest) hoặc chữ ký số (digital fingerprint). Hiểu 1 cách đơn giản là từ 1 file dữ liệu ban đầu sau khi trải qua thuật toán băm thì sẽ tạo ra những chuỗi ký tự ngẫu nhiên nhất định và đặc điểm nổi bật của hàm băm đó là một khi ta đã chuyển đổi file dữ liệu thành ký tự ngẫu nhiên thì không thể có bất kỳ cách nào để có thể khôi phục lại từ dữ liệu đã băm thành file dữ liệu ban đầu.



Hình 1.11. Nguyên lý hoạt động hàm băm

Hàm băm có vai trò xác thực các khối với nhau để đảm bảo về tính liên kết giữa các khối, qua đó tạo nên tính ổn định và an toàn đối với mỗi chuỗi khối. Mỗi khối sẽ được tạo ra bằng hai hàm băm đầu và cuối với cơ chế là hàm băm đầu của khối trước sẽ là hàm băm cuối của khối liền kề sau nó.

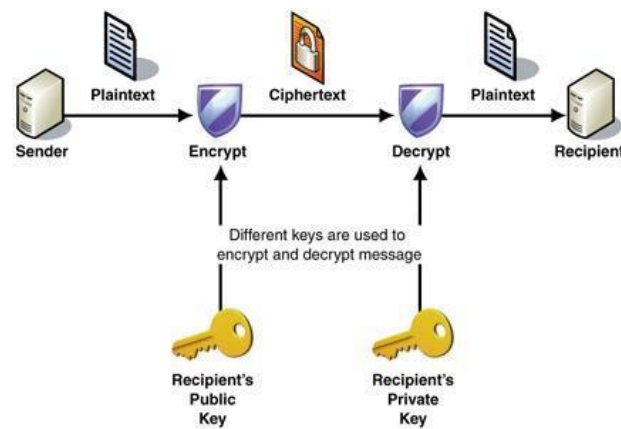


Hình 1.12. Cơ chế liên kết giữa các khối

b. Mã hoá RSA

Trong mật mã học, RSA là một thuật toán mật mã hóa khóa công khai. Đây là thuật toán đầu tiên phù hợp với việc tạo ra chữ ký điện tử đồng thời với việc mã hóa. Nó đánh dấu một sự tiến bộ vượt bậc của lĩnh vực mật mã học trong việc sử dụng khóa công cộng.

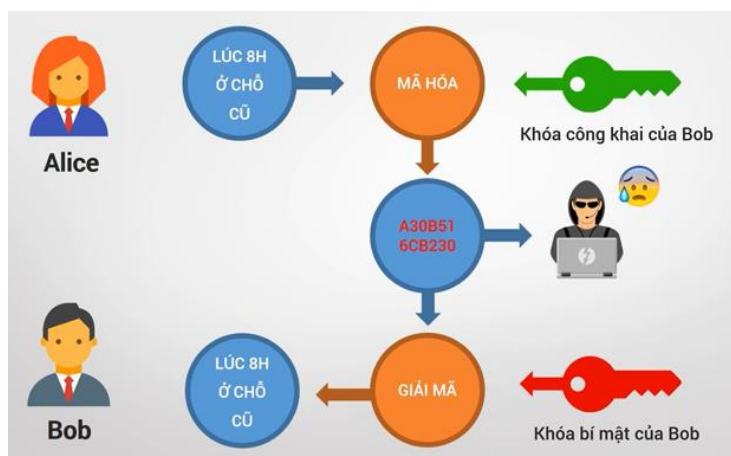
Thuật toán RSA có hai khóa: khoá công khai và khoá bí mật. Khóa công khai được công bố rộng rãi cho mọi người và được dùng để mã hóa. Những thông tin được mã hóa bằng khóa công khai chỉ có thể được giải mã bằng khóa bí mật tương ứng. Nói cách khác, mọi người đều có thể mã hóa nhưng chỉ có người biết khóa bí mật mới có thể giải mã được.



Hình 1.13. Nguyên lý hoạt động mã hoá RSA

Ta có thể mô phỏng trực quan một hệ mật mã khoá công khai như sau:

- Alice muốn gửi cho Bob một thông tin mật mà Alice muốn duy nhất Bob có thể đọc được.
- Để làm được điều này, Alice dùng khóa công khai của Bob để mã hóa rồi gửi lên trên mạng lưới. Lúc này kể cả hacker có hack được thì cũng chỉ đọc được những chuỗi ký tự ngẫu nhiên. Sau khi mã hóa xong thì ngay cả Alice cũng không thể mở lại được - không đọc lại hay sửa thông tin trong thư được nữa.
- Sau khi Bob nhận được thông tin của Alice thì Bob sẽ dùng khóa bí mật của Bob để giải mã nội dung của bức thư.



Hình 1.14. Mô tả hoạt động của RSA

Trong ví dụ này, bức thư với khóa công khai của Bob sẽ được công khai cho toàn bộ mạng lưới biết nhưng chiếc chìa khóa chính là khóa bí mật của riêng một mình Bob.

1.5.2. Quy tắc cuốn sổ cái

Cơ sở dữ liệu là Blockchain và mỗi nút trên Blockchain có quyền truy cập vào toàn bộ Blockchain. Không một nút hoặc máy tính nào điều chỉnh thông tin chứa trong đó. Mọi nút đều có thể xác thực các bản ghi của Blockchain. Tất cả điều này được thực hiện mà không có một hoặc một vài trung gian kiểm soát mọi thứ. Các giao dịch diễn ra ngang hàng (P2P), trực tiếp giữa 2 bên, không thông qua một bên thứ ba. Thông tin về những gì đang xảy ra trên Blockchain được lưu trữ trên mỗi nút sau đó được chuyển đến các nút lân cận. Bằng cách này, thông tin lan truyền qua toàn bộ mạng.

Bất cứ ai cũng có khả năng nhìn thấy mọi giao dịch và giá trị băm của nó. Tất cả những gì bạn thấy trên Blockchain là bản ghi các giao dịch giữa các địa chỉ Blockchain. Mỗi nút trong Blockchain đều đang lưu giữ một bản sao của sổ cái. Do vậy, mỗi nút đều biết số dư tài khoản của bạn là bao nhiêu. Hệ thống Blockchain chỉ ghi lại mỗi giao dịch được yêu cầu chứ không hề theo dõi số dư tài khoản của bạn.

Sau khi ghi lại giao dịch trên Blockchain và Blockchain đã được cập nhật, thì không thể thay đổi hồ sơ của giao dịch này. Hồ sơ của một giao dịch cụ thể được liên kết với hồ sơ trước. Các bản ghi Blockchain là vĩnh viễn, chúng được sắp xếp theo thứ tự thời gian và chúng đã cập nhật ở tất cả các nút khác.

1.5.3. Nguyên lý tạo khối

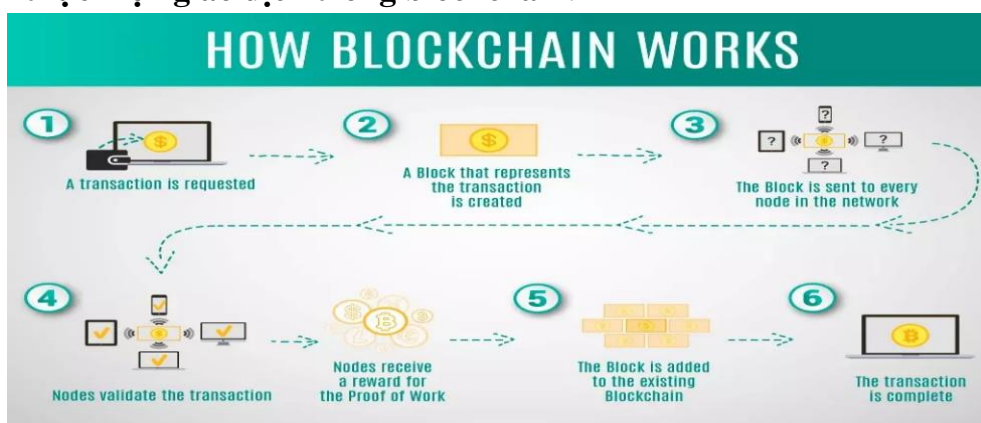
Các dữ liệu hay giao dịch sau khi được đưa lên hệ thống Blockchain sẽ được nhóm vào các khối khác nhau. Những giao dịch chưa được thực hiện cũng được tạo thành 1 khối riêng và coi như chúng chưa được xác nhận. Mỗi nút có thể nhóm các giao dịch với nhau thành một khối và gửi nó vào mạng lưới như một hàm ý cho các khối tiếp theo được gắn vào sau đó. Để được thêm vào Blockchain, mỗi khối phải chứa một đoạn mã đóng vai trò như một đáp án cho một vấn đề toán học phức tạp được tạo ra bằng hàm mã hóa băm không thể đảo ngược.

Sau khi tạo thành khối, các client sẽ gửi thông tin cho các peer khác để xem xét. Nếu các peer khác cùng đồng thuận cho phép thì một khối mới sẽ được thêm vào chuỗi hiện có.

1.5.4. Thuật toán bảo mật Blockchain

Mạng lưới Blockchain sử dụng nhiều biện pháp để bảo mật giao dịch, ngăn chặn gian lận. Khi có bất đồng về khối đại diện cuối cùng của chuỗi, cơ chế này sẽ kích hoạt để đảm bảo tính toàn vẹn của dữ liệu. Một thuật toán bảo mật khác là mỗi khối chứa một lệnh tham chiếu của khối trước đó, tạo thành một chuỗi liên kết mật mã. Việc thêm một khối mới vào chuỗi đòi hỏi giải quyết một bài toán toán học phức tạp, khiến việc gian lận hay thao túng dữ liệu trong những giao dịch tồn tại trong Blockchain càng lâu thì tính bảo mật càng cao. Nguyên nhân của điều này là quá trình thêm một block vào hệ thống có thể mất vài phút đến vài giờ, tùy thuộc vào mạng lưới. Sau khi được xác nhận, giao dịch sẽ trở nên an toàn và không thể đảo ngược.

* **Quá trình thực hiện giao dịch trong blockchain:**



Hình 1.15. Cách thức thực hiện giao dịch của Blockchain

1. *Giao dịch được yêu cầu:* Một giao dịch mới đi vào mạng lưới blockchain. Tất cả thông tin cần truyền đi đều được mã hóa kép bằng khóa công khai và khóa riêng tư.

2. *Xác minh giao dịch:* Giao dịch sau đó được truyền đến mạng lưới máy tính ngang hàng phân bố trên toàn thế giới. Tất cả các nút trên mạng sẽ kiểm tra tính hợp lệ của giao dịch như số dư đủ để thực hiện giao dịch hay không.

3. *Hình thành một khối mới:* Trong một mạng lưới blockchain thông thường có nhiều nút và nhiều giao dịch được xác minh cùng một lúc. Khi giao dịch được xác minh và tuyên bố là giao dịch hợp lệ, nó sẽ được thêm vào mempool. Tất cả các giao dịch được xác minh tại một nút cụ thể tạo thành một mempool và nhiều mempool như vậy tạo thành một khối.

4. *Thuật toán đồng thuận:* Các nút tạo thành một khối sẽ cố gắng thêm khối vào mạng blockchain để làm cho nó trở thành một phần của chuỗi. Tuy nhiên, nếu mọi nút đều được phép thêm khối theo cách này thì nó sẽ làm gián đoạn hoạt động của mạng blockchain. Để giải quyết vấn đề này, các nút sử dụng cơ chế đồng thuận để đảm bảo rằng mọi khối mới được thêm vào blockchain là phiên bản duy nhất được tất cả các nút trong blockchain đồng ý và chỉ có các khối hợp lệ mới được gắn vào blockchain. Nút được chọn để thêm một khối vào blockchain sẽ nhận được phần thưởng và do đó chúng ta gọi họ là "thợ đào" (miners). Thuật toán đồng thuận tạo ra một mã băm cho khối đó, mã này là yêu cầu bắt buộc để thêm khối vào blockchain.

5. *Thêm khối mới vào blockchain:* Sau khi khối mới tạo có giá trị băm và đã được xác thực, giờ đây nó đã sẵn sàng để được thêm vào blockchain. Trong mỗi khối, có một giá trị băm của khối trước đó và đó là cách các khối được liên kết với nhau về mặt mã hóa để tạo thành một blockchain. Một khối mới được thêm vào đầu cuối của blockchain.

6. *Giao dịch hoàn tất:* Ngay khi khối được thêm vào blockchain, giao dịch được hoàn tất và thông tin chi tiết về giao dịch này được lưu trữ vĩnh viễn trong blockchain. Bất kỳ ai cũng có thể lấy thông tin chi tiết về giao dịch và xác nhận giao dịch.

1.6. Đặc điểm nổi bật của Blockchain

1.6.1. Tính bất biến

Tính bất biến (Immutability) có nghĩa là Blockchain là một mạng lưới vĩnh viễn và không thể thay đổi. Công nghệ Blockchain hoạt động thông qua một tập hợp các nút (nodes). Khi một giao dịch được ghi lại trên Blockchain, nó không thể bị chỉnh sửa hoặc xóa bỏ. Điều này làm cho Blockchain trở thành một sổ cái bất biến và chống giả mạo, mang lại mức độ an ninh và tin cậy cao.

Mỗi nút trong mạng lưới đều giữ một bản sao của sổ cái kỹ thuật số. Để thêm một giao dịch mới, mỗi nút sẽ kiểm tra tính hợp lệ của giao dịch. Nếu đa số các nút đồng ý rằng giao dịch là hợp lệ, giao dịch đó sẽ được thêm vào mạng lưới. Điều này có nghĩa là không ai có thể thêm bất kỳ khối giao dịch nào vào sổ cái mà không có sự chấp thuận của đa số các nút.

Bất kỳ bản ghi nào đã được xác thực đều không thể thay đổi và không thể chỉnh sửa. Điều này đảm bảo rằng bất kỳ người dùng nào trong mạng lưới cũng không thể sửa đổi, thay đổi hoặc xóa dữ liệu đã ghi.

1.6.2. Tính phân tán

Tất cả các thành viên trong mạng đều có một bản sao của sổ cái để đảm bảo tính minh bạch hoàn toàn. Một sổ cái công khai sẽ cung cấp đầy đủ thông tin về tất cả các thành viên trong mạng và các giao dịch. Sức mạnh tính toán được phân bổ trên nhiều máy tính giúp đảm bảo kết quả tốt hơn.

Sổ cái phân tán là một trong những đặc điểm quan trọng của blockchain vì những lý do sau:

- Trong sổ cái phân tán, việc theo dõi các thay đổi trở nên dễ dàng vì các thay đổi được lan truyền rất nhanh.
- Mỗi nút (node) trong mạng blockchain đều phải duy trì sổ cái và tham gia vào quá trình xác thực.
- Bất kỳ thay đổi nào trong sổ cái sẽ được cập nhật chỉ trong vài giây hoặc vài phút. Nhờ không có sự tham gia của các trung gian, quá trình xác thực các thay đổi diễn ra nhanh chóng.
- Nếu một người dùng muốn thêm một khối mới, các nút tham gia khác phải xác minh giao dịch. Để một khối mới được thêm vào mạng blockchain, nó phải được đa số các nút trong mạng phê duyệt.
- Trong mạng blockchain, không có nút nào nhận được sự ưu tiên hay đối xử đặc biệt. Tất cả đều phải tuân thủ quy trình tiêu chuẩn để thêm khối mới vào mạng.

1.6.3. Tính phi tập trung

Công nghệ Blockchain là một hệ thống phi tập trung, nghĩa là không có cơ quan trung ương nào kiểm soát mạng lưới. Thay vào đó, mạng lưới được tạo thành từ một số lượng lớn các nút (nodes) làm việc cùng nhau để xác minh và xác thực các giao dịch. Mỗi nút trong mạng blockchain đều có một bản sao giống hệt của sổ cái.

Tính chất phi tập trung mang lại nhiều lợi ích cho mạng lưới blockchain:

- Vì mạng blockchain không phụ thuộc vào tính toán của con người, nên hệ thống được tổ chức hoàn chỉnh và có khả năng chịu lỗi cao.
- Nhờ tính phi tập trung, mạng blockchain ít có khả năng gặp sự cố. Việc tấn công hệ thống trở nên tốn kém hơn đối với hacker, do đó nguy cơ thất bại của mạng lưới giảm đáng kể.
- Không có bên thứ ba tham gia, do đó không có rủi ro phát sinh từ các trung gian trong hệ thống.
- Tính phi tập trung của blockchain giúp tạo ra hồ sơ minh bạch cho mọi thành viên trong mạng lưới. Vì vậy, mọi thay đổi đều có thể theo dõi được và có tính cụ thể hơn.
- Người dùng có quyền kiểm soát tài sản của mình mà không cần dựa vào bên thứ ba để quản lý hoặc duy trì tài sản.

1.6.4. Bảo mật

Tất cả các bản ghi trên blockchain đều được mã hóa riêng lẻ. Việc sử dụng mã hóa tạo thêm một lớp bảo mật cho toàn bộ quy trình trên mạng blockchain. Mặc dù không có cơ quan trung ương, điều đó không có nghĩa là bất kỳ ai cũng có thể tùy ý thêm, cập nhật hoặc xóa dữ liệu trên mạng.

Mọi thông tin trên blockchain đều được băm (hashed) bằng các thuật toán mã hóa, nghĩa là mỗi dữ liệu đều có một danh tính duy nhất trên mạng. Mỗi khối (block) đều chứa một mã băm (hash) riêng biệt và mã băm của khối liền trước nó. Nhờ tính chất này, các khối được liên kết mật mã với nhau.

Bất kỳ nỗ lực nào để chỉnh sửa dữ liệu sẽ yêu cầu thay đổi tất cả các mã băm, điều này gần như không thể thực hiện được.

1.6.5. Thanh toán nhanh chóng

Hệ thống ngân hàng truyền thống thường gặp nhiều vấn đề, chẳng hạn như mất nhiều ngày để xử lý một giao dịch sau khi hoàn tất các thủ tục thanh toán, và dễ bị tổn thương bởi các yếu tố tham nhũng. Ngược lại, blockchain cung cấp khả năng thanh toán nhanh hơn so với hệ thống ngân hàng truyền thống. Tính năng này giúp cuộc sống trở nên thuận tiện hơn.

CHƯƠNG 2. ỨNG DỤNG CỦA BLOCKCHAIN

2.1. BITCOIN – Ứng dụng đầu tiên của Blockchain

2.1.1. Khái niệm Bitcoin

Bitcoin là một loại tiền mã hóa, được phát minh bởi một cá nhân hoặc một tổ chức vô danh tên Satoshi Nakamoto dưới dạng phần mềm mã nguồn mở từ năm 2009. Bitcoin có thể được trao đổi trực tiếp bằng thiết bị kết nối Internet mà không cần thông qua một tổ chức tài chính trung gian nào.

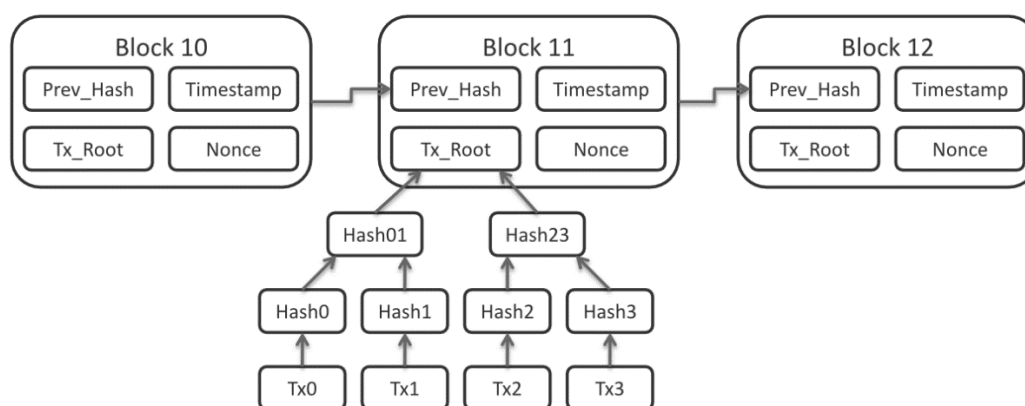
Bitcoin không chỉ là một phương tiện thanh toán mà còn được xem như một tài sản đầu tư. Nó có tính chất phi tập trung, nghĩa là không có một tổ chức nào kiểm soát nó.



Hình 2.1. Logo Bitcoin

2.1.2. Mối liên hệ giữa Blockchain và Bitcoin

Blockchain là một cuốn sổ cái ghi lại tất cả các giao dịch. Dữ liệu trong cuốn sổ cái liên tục được mạng lưới máy tính ngang hàng trên thế giới cập nhật và bảo trì. Giao dịch khi A gửi X bitcoin cho B được ghi lại trên toàn hệ thống, tất cả các máy tính trong mạng này sẽ xác minh và ghi lại giao dịch đó vào cuốn sổ cái rồi cấp phát dữ liệu này tới các máy tính khác. Blockchain là một cơ sở dữ liệu phân tán vô chủ; các máy tính liên tục thực hiện việc kiểm toán độc lập bằng cách xác minh dữ liệu nhận tới và so sánh với chữ ký của giao dịch đó.



Hình 2.2. Dữ liệu trong khối Bitcoin

2.1.3. Tài khoản và Ví Bitcoin

Cấu trúc ví Bitcoin : Ví Bitcoin bao gồm địa chỉ công khai (để nhận bitcoin) và khóa riêng tư (để gửi bitcoin). Địa chỉ công khai được tạo ra từ khóa công khai, trong khi khóa riêng tư phải được bảo mật.

Sự quan trọng của khóa riêng tư : Khóa riêng tư là yếu tố quyết định quyền sở hữu bitcoin. Nếu người dùng mất khóa riêng tư, họ sẽ không thể truy cập vào số bitcoin đó, dẫn đến việc mất mát vĩnh viễn.

2.1.4. Tính bảo mật

Cũng giống như các chính phủ phụ thuộc vào quyền lực của quân đội và cơ quan hành pháp để đảm bảo an ninh tiền tệ, qua đó mang lại lòng tin và giá trị cho tiền pháp định, thì độ an toàn của hệ thống Bitcoin phụ thuộc vào khả năng xử lý (hashing power) của toàn bộ mạng lưới blockchain để chống lại các nguy cơ phá hoại đồng thời mang lại giá trị, niềm tin cho Bitcoin. Đã có nhiều vụ trộm Bitcoin thành công xảy ra nhưng tất cả đều có một đặc điểm chung là do nạn nhân để lộ khóa riêng tư cho kẻ tấn công. Cho tới nay, giao thức Bitcoin vẫn chưa hề có lỗ hổng bảo mật nào để làm mất Bitcoin của người dùng mà không dùng đến khóa riêng tư.

2.1.5. Tính riêng tư

Bitcoin là loại tiền tệ bán ẩn danh, tức là số tiền không gắn với thực thể trong thế giới thật, mà gắn với địa chỉ Bitcoin. Tuy chủ sở hữu địa chỉ Bitcoin không được xác định rõ ràng, nhưng bù lại các giao dịch lại được công khai. Ngoài ra, các giao dịch có thể được liên kết với cá nhân hoặc công ty thông qua việc phân tích dòng giao dịch (ví dụ: nếu các giao dịch chi tiêu từ nhiều nguồn đầu vào thì có thể các nguồn đó đều chung chủ) và kết hợp dữ liệu đến từ các nguồn đã được định danh (các sàn giao dịch Bitcoin có thể được yêu cầu lưu trữ thông tin cá nhân người sử dụng). Mặc dù vậy, cũng như tiền mặt, việc xác định địa chỉ Bitcoin nào gắn với người nào là tương đối khó. Để tăng tính riêng tư, mỗi giao dịch cần sử dụng một địa chỉ Bitcoin mới.

2.1.6. Tìm hiểu về đào Bitcoin

a. Khái niệm đào Bitcoin

Đào Bitcoin (Bitcoin mining) là quá trình xác minh và ghi lại các giao dịch Bitcoin vào blockchain, đồng thời tạo ra các khối mới. Để khối mới được chấp nhận bởi mạng lưới, nó cần phải chứa bằng chứng công việc (Proof of Work). Proof of Work yêu cầu thợ đào tìm kiếm một số nonce (số ngẫu nhiên), mà khi kết hợp với nội dung của khối và được xử lý qua hàm băm (hash function), sẽ tạo ra một giá trị nhỏ hơn số target của mạng lưới. Điều này có nghĩa là việc xác nhận khối mới rất dễ dàng cho các máy tính, nhưng lại đòi hỏi thời gian và tài nguyên lớn để thực hiện.

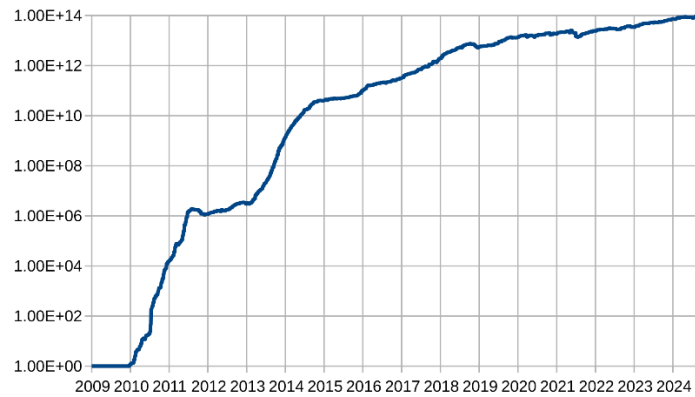
b. Quy trình đào Bitcoin

Quá trình khai thác Bitcoin có thể được hiểu rõ hơn qua hai yếu tố chính: nonce và target. Thợ đào phải thử nghiệm với nhiều giá trị nonce khác nhau cho đến khi tìm ra giá trị tạo ra một mã băm thỏa mãn điều kiện nhỏ hơn số target. Điều này làm cho quá trình đào trở nên cạnh tranh và tốn thời gian, vì thợ đào phải thử hàng triệu giá trị nonce để đạt được kết quả mong muốn.

c. Độ khó của việc đào Bitcoin

Mỗi 2016 khối được tạo ra (tương đương khoảng 14 ngày), độ khó đào Bitcoin sẽ được tự động điều chỉnh dựa trên khả năng xử lý của toàn bộ mạng lưới. Mục tiêu của việc điều chỉnh này là giữ cho thời gian trung bình giữa các khối mới được tạo ra là khoảng 10 phút. Nhà phát minh của Bitcoin là Satoshi Nakamoto đã xây dựng các quy tắc khai thác theo cách mà mạng

lưới khai thác càng nhiều thì lúc đó độ khó càng tăng cao. Việc đoán câu trả lời cho vấn đề toán học khai thác càng phức tạp. Từ tháng 3 năm 2014 đến tháng 3 năm 2015, số lượng nonce trung bình mà thợ đào phải thử nghiệm để tạo ra một khối mới đã tăng từ 16,4 tỷ tỷ lên 200,5 tỷ tỷ, cho thấy sự gia tăng độ khó trong việc đào Bitcoin.



Hình 2.3. Độ khó trong việc đào Bitcoin qua các năm

d. Bảo mật và khó khăn trong đào Bitcoin

Hệ thống Proof of Work không chỉ giúp xác nhận giao dịch mà còn bảo vệ blockchain khỏi các cuộc tấn công. Để thay đổi một khối trong blockchain, kẻ tấn công cần phải thay đổi tất cả các khối phía sau nó, điều này đòi hỏi hơn 50% sức mạnh xử lý của toàn bộ mạng lưới. Sự khó khăn này làm cho việc tấn công mạng lưới Bitcoin trở nên rất khó khăn và tốn kém.

e. Phần thưởng và tổng số Bitcoin đang lưu hành

Đây là phần thưởng chính mà người đào nhận được khi tìm thấy và thêm một khối mới vào blockchain. Ban đầu, phần thưởng này là 50 BTC cho mỗi khối khi Bitcoin mới được giới thiệu vào năm 2009. Theo quy định của giao thức Bitcoin, phần thưởng này sẽ giảm một nửa sau mỗi 210.000 khối (khoảng 4 năm). Cuối cùng, tổng số bitcoin sẽ đạt ngưỡng 21 triệu bitcoin vào năm 2140, khi đó thợ đào sẽ chỉ nhận được phí giao dịch.

2.1.7. Sự khác biệt giữa bitcoin và blockchain

- Bitcoin là một tiền tệ, trong khi blockchain là công nghệ:

Bitcoin là một loại tiền tệ kỹ thuật số được sử dụng cho các giao dịch trực tuyến. Trong khi đó, blockchain là công nghệ lưu trữ thông tin phi tập trung và không thể sửa đổi. Blockchain có thể ứng dụng trong nhiều ngành nghề, lĩnh vực khác nhau như y tế, bất động sản, chuỗi cung ứng...vv

- Bitcoin có giá trị thị trường, trong khi blockchain không:

Giá trị của bitcoin phụ thuộc vào sự đánh giá của thị trường. Nó có thể tăng hoặc giảm do yếu tố cung cầu và tâm lý thị trường. Trong khi đó, blockchain không có giá trị thị trường và không chịu sự ảnh hưởng trực tiếp từ yếu tố thị trường.

- Bitcoin là ứng dụng đầu tiên của blockchain:

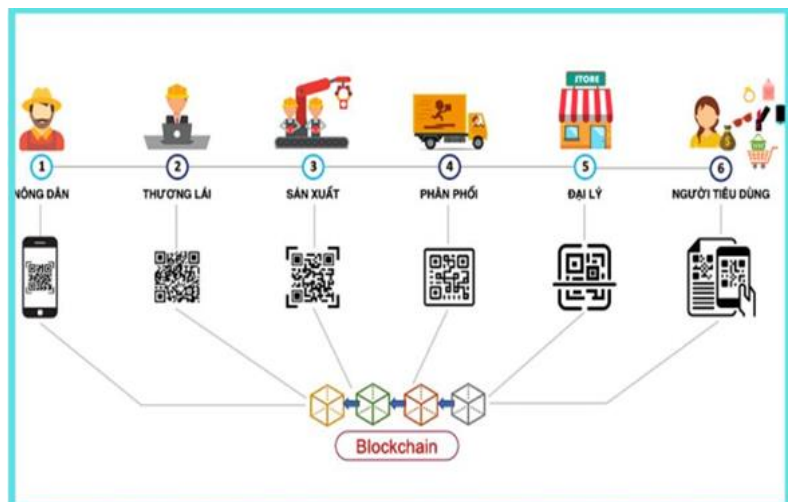
Bitcoin là ứng dụng đầu tiên sử dụng công nghệ blockchain. Tuy nhiên, hiện nay đã xuất hiện nhiều loại tiền tệ kỹ thuật số khác sử dụng công nghệ blockchain hoặc các biến thể của nó.

2.2. Ứng dụng của Blockchain trong đời sống

2.2.1. Sản xuất

Trong quá trình sản xuất, chúng ta cần một cuốn sổ cái để giám sát quy trình sản xuất, hàng tồn kho, phân phối, chất lượng, thông tin giao dịch... Blockchain sẽ thay một thiết bị thông minh cấp quyền quản lý hiệu quả nhằm gia tăng đáng kể năng suất cho các quy trình quản lý chuỗi công ứng. Đối với người tiêu dùng có thể kiểm tra thông tin sản phẩm đó có phải hàng chính hãng hay không sẽ ngăn chặn toàn bộ những sản phẩm nhái, hàng giả trên thị trường. Một số ứng dụng của Blockchain trong sản xuất:

- Truy xuất nguồn gốc sản phẩm được sản xuất qua các khâu.
- Theo dõi nguồn cung cấp nguyên liệu sản xuất trong công nghiệp.
- Quản lý hàng tồn kho, kho bãi sản xuất.
- Theo dõi lịch trình sản xuất, số lượng hàng mua vào và bán ra.



Hình 2.4. Ứng dụng blockchain trong sản xuất

2.2.2. Y tế

Trong thời buổi công nghệ 4.0, các quốc gia trên thế giới cũng như Việt Nam đã đẩy mạnh triển khai số hoá thông tin trong quá trình quản lý dữ liệu, trong đó có lĩnh vực chăm sóc sức khỏe. Blockchain được áp dụng để quản lý tài sản và lưu trữ thông tin về sức khỏe người bệnh, quản lý kho, đơn đặt hàng, thanh toán cho các thiết bị y tế cũng như dược phẩm. Tuy có nhiều thiết bị thông minh để giám sát các dịch vụ này nhưng còn nhiều hạn chế về tính bảo mật thông tin cá nhân của bệnh nhân. Vì thế, Blockchain là một lựa chọn được ưu tiên.

Một số ứng dụng của Blockchain trong lĩnh vực Y tế:

- Ứng dụng phát triển bao gồm theo dõi và quản lý bệnh lý (như thuốc thông minh, thiết bị đeo có thể đo các chỉ số về sức khỏe và đưa ra phản hồi) và tăng cường quản lý chất lượng.
- Quản lý chuỗi cung ứng thuốc, thiết bị y tế: Theo dõi đầu vào, nguồn gốc, hạn sử dụng của các vật tư y tế.
- Tăng cường tính minh bạch và tự động hóa trong các giao dịch khám chữa bệnh; xuất xứ xét nghiệm lâm sàng; quyền sở hữu dữ liệu sức khỏe của bệnh nhân.



Hình 2.5. Ứng dụng y tế

2.2.3. Giáo dục

Khi áp dụng Blockchain vào giáo dục thông tin lưu trữ trên chuỗi khối không chỉ là dữ liệu bảng điểm mà còn cả quá trình đào tạo, kinh nghiệm thực tế, lịch sử tuyển dụng của từng cá nhân. Tránh trường hợp các ứng viên gian lận trong quá trình xin cấp học bổng, thăng chức...; khai gian trình độ học vấn, kinh nghiệm làm việc, kỷ luật. Không những thế, với tính năng hợp đồng thông minh, Blockchain còn cho phép thực thi tự động các điều khoản trong quy chế đào tạo, xử lý các trường hợp vi phạm quy chế, cải tiến những hạn chế trong quá trình giảng dạy nếu học viên có ý kiến phản hồi.

Một số ứng dụng của Blockchain trong lĩnh vực Giáo dục:

- Theo dõi và lưu trữ bảng điểm và bằng cấp của sinh viên và thông tin của các đơn vị đào tạo.
- Xem xét cá nhân/ứng viên có phù hợp với công việc giảng dạy hay không, từ đó đưa ra quyết định mời cá nhân đó làm việc.
- Hệ thống quản lý mức độ đánh giá sự uy tín trong nghiên cứu khoa học.
- Ghi lại cơ sở dữ liệu bảo mật về dữ liệu học tập và điểm số cho các hệ thống học trực tuyến, đánh giá năng lực của một cá nhân dựa trên các yêu cầu tuyển sinh đầu vào.

2.2.4. Dịch vụ tài chính & ngân hàng

Với đặc thù ngành công nghiệp tài chính và ngân hàng dễ xảy ra tình trạng tập trung quyền lực, xâm phạm dữ liệu người dùng, tính bảo mật, do đó với công nghệ Blockchain hiện nay sẽ giải quyết được những vấn đề này. Nhờ tính năng hợp đồng thông minh mà có thể bỏ qua khâu trung gian, giúp tiết kiệm chi phí, đẩy nhanh các giao dịch, hạn chế các rủi ro tài chính trong quá trình thanh toán, cải tiến các hệ thống quản lý thông tin công nghệ cũ...

Một số ứng dụng của Blockchain trong lĩnh vực Tài chính & Ngân hàng:

- Xác thực thông tin khách hàng, khả năng tín dụng: Cho phép giao dịch ngay cả không có trung gian xác minh.
- Mạng lưới sẽ xác minh và thanh toán những giao dịch ngang hàng, công việc này được thực hiện liên tục nên sổ cái luôn được cập nhật.
- Quản lý rủi ro, hạn chế rủi ro trong thanh toán vì trực trực kỹ thuật, vỡ nợ trước khi thanh toán giao dịch.
- Hệ thống quản lý thông minh: Blockchain cho phép liên tục đổi mới, lặp lại và cải tiến, dựa trên sự đồng thuận trong mạng lưới.



Hình 2.6. Ứng dụng trong tài chính & ngân hàng

2.2.5. Thương mại điện tử

Theo nhiều chuyên gia, thị trường bán lẻ hiện nay đang dần chuyển qua hình thức thương mại trực tuyến đặc biệt là với sự phát triển của các sản phẩm thương mại điện tử. Điều đó đặt ra vấn đề về tính bảo mật, quản lý chuỗi cung ứng, quá trình vận chuyển hàng hoá đến người tiêu dùng,

chi phí từ cách làm truyền thống tạo nên nhiều rào cản giữa người tiêu dùng và nhà sản xuất. Blockchain giải quyết vấn đề đó bằng các hợp đồng thông minh, tạo điều kiện cho các bên ký kết dễ dàng, liên kết với các doanh nghiệp đa quốc gia với chi phí tiết kiệm nhờ lược bỏ trung gian, giải pháp thanh toán cũng được gắn trực tiếp trên các website, sàn thương mại điện tử.

Một số ứng dụng của Blockchain trong lĩnh vực Thương mại điện tử:

- Quản lý thông tin dữ liệu khách hàng.
- Theo dõi thông tin, tình trạng sản phẩm thông qua số serial, QR.
- Xây dựng hệ thống thanh toán và chấp nhận ví điện tử, khách hàng thân thiết, thẻ quà tặng, tri ân khách hàng....
- Vận hành và quản lý chuỗi cung ứng

2.2.6. Truyền thông và viễn thông

Bằng cách triển khai các giải pháp Blockchain trên nền tảng đám mây sẽ giúp các nhà cung cấp dịch vụ truyền thông tối ưu hóa các quy trình hiện có trong khi tăng cường bảo mật mạng, rà soát lại toàn bộ quy trình vận hành, các quy trình như chuyển vùng và quản lý danh tính trong mô hình kinh doanh của mình. Từ đó cải thiện và phát triển dịch vụ tốt hơn.

Một số ứng dụng của Blockchain trong lĩnh vực Truyền thông và viễn thông:

- Phòng chống gian lận trong chuyển vùng: các thỏa thuận chuyển vùng giữa các nhà khai thác sẽ trở nên minh bạch, các nút được chỉ định có thể đóng vai trò là trình xác nhận (người khai thác) để xác minh từng giao dịch được phát trên mạng.
- Quản lý danh tính và xác thực: khách hàng sẽ chỉ yêu cầu ID ảo để tự xác thực, dẫn đến mức độ hài lòng cao hơn rất nhiều.
- Quá trình chuyển đổi 5G: các quy tắc và thỏa thuận giữa các mạng khác nhau sẽ có dạng hợp đồng thông minh, tự thực hiện có thể kết nối các thiết bị với nhà cung cấp dịch vụ gần nhất đồng thời đánh giá sự liên tục của kết nối và tính phí dịch vụ.
- Kết nối Internet vạn vật (IoT): tạo ra một môi trường an toàn hơn để truyền dữ liệu bằng cách tạo các mạng lưới tự quản ngang hàng an toàn cao.



Hình 2.7. Ứng dụng thương mại điện tử



Hình 2.8. Ứng dụng trong truyền thông

CHƯƠNG 3. MỘT SỐ NỀN TẢNG BLOCKCHAIN VÀ THỬ NGHIỆM BLOCKCHAIN TRÊN HYPERLEDGER FABRIC

3.1. Một số nền tảng Blockchain

3.1.1. Ethereum

Sau sự thành công của Bitcoin, một loại tiền điện tử khác cũng gây tiếng vang trong thị trường số hiện nay là Ethereum. Ethereum cho phép mọi người xây dựng và sử dụng các ứng dụng phi tập trung dựa trên công nghệ Blockchain. Nó là một dự án mã nguồn mở, có thể chuyển đổi và linh hoạt hơn Bitcoin.

Ethereum có các đặc điểm sau:

- Là mạng mở.
- Sử dụng mô hình đồng thuận bằng chứng công việc.
- Có lượng người theo dõi trên Github cao.
- Hỗ trợ các ngôn ngữ như C++, Go và Python.



Hình 3.1. Logo Ethereum

3.1.2. Hyperledger Fabric

Hyperledger Fabric là một nền tảng Blockchain mã nguồn mở được phát triển bởi Linux Foundation. Nó được thiết kế đặc biệt cho các ứng dụng doanh nghiệp, cho phép các tổ chức xây dựng các giải pháp Blockchain tùy chỉnh.

Hyperledger Fabric có các đặc điểm sau:

- Là nền tảng Blockchain riêng tư và cho phép quyền truy cập được kiểm soát.
- Sử dụng mô hình đồng thuận linh hoạt, có thể tùy chỉnh.
- Hỗ trợ các hợp đồng thông minh viết bằng nhiều ngôn ngữ lập trình như Go và Java.
- Có khả năng mở rộng cao và hỗ trợ các giao dịch song song.



Hình 3.2. Logo Hyperledger

3.1.3. Binance Smart Chain (BSC)

Binance Smart Chain là một nền tảng Blockchain được phát triển bởi Binance, hỗ trợ các hợp đồng thông minh và tương thích với Ethereum. BSC nổi bật với phí giao dịch thấp và tốc độ xử lý nhanh.

Binance Smart Chain có các đặc điểm sau:

- Là mạng mở.
 - Sử dụng mô hình đồng thuận bằng chứng cổ phần ủy quyền (DpoS - Delegated Proof of Stake).
 - Tương thích với Máy ảo Ethereum (EVM – Ethereum Virtual Machine).
 - Hỗ trợ các ngôn ngữ lập trình như Solidity và Vyper.
- Chain



Hình 3.3. Logo Binance Smart

3.1.4. Solana

Solana là một nền tảng Blockchain nổi bật với tốc độ xử lý giao dịch rất nhanh và phí giao dịch thấp. Solana được sử dụng rộng rãi trong các ứng dụng tài chính phi tập trung (DeFi – Decentralized Finance) và NFT (Non-Fungible Tokens).

Solana có các đặc điểm sau:

- Là mạng mở.
- Sử dụng mô hình đồng thuận bằng chứng lịch sử (PoH – Proof of History) kết hợp với bằng chứng cổ phần (PoS).
- Có khả năng xử lý hàng nghìn giao dịch mỗi giây.
- Hỗ trợ các ngôn ngữ lập trình như Rust và C.



Hình 3.4. Solana

3.2. Thử nghiệm Blockchain trên môi trường Hyperledger Fabric

Phần cài đặt thử nghiệm được thực hiện trên:

- Máy ảo Ubuntu v24.04.1
- Hyperledger Fabric v2.5.9

3.2.1. Cài đặt các phần mềm Git, cURL, Docker-compose

- Cài đặt Git: `sudo apt-get install git -y`
- Cài đặt cURL: `sudo apt-get install curl -y`
- Cài đặt Docker-compose: `sudo apt -y install docker-compose`
- Kiểm tra phiên bản của các phần mềm vừa cài đặt:

`git --version`

`curl --version`

`docker --version`

`docker-compose --version`

- Khi đã cài đặt thành công ta được kết quả như hình:

```
vht@VHT2901:~$ git --version
git version 2.43.0
vht@VHT2901:~$ curl --version
curl 8.5.0 (x86_64-pc-linux-gnu) libcurl/8.5.0 OpenSSL/3.0.13 zlib/1.3 brotli/1.1.0 zstd/1.5.5 libidn2/2.3.7 libpsl/0.21.2 (+libidn2/2.3.7) libssh/0.10.6/openssl/zlib nghttp2/1.59.0 librtmp/2.3 OpenLDAP/2.6.7
Release-Date: 2023-12-06, security patched: 8.5.0-2ubuntu10.4
Protocols: dict file ftp ftps gopher gophers http https imap imaps ldap ldaps mqtt pop3 pop3s rtmp rtsp scp sftp smb smb
s smtp smtps telnet tftp
Features: alt-svc AsynchDNS brotli GSS-API HSTS HTTP2 HTTPS-proxy IDN IPv6 Kerberos Largefile libz NTLM PSL SPNEGO SSL t
hreadsafe TLS-SRP UnixSockets zstd
vht@VHT2901:~$ docker --version
Docker version 24.0.7, build 24.0.7-0ubuntu4.1
vht@VHT2901:~$ docker-compose --version
docker-compose version 1.29.2, build unknown
vht@VHT2901:~$
```

Hình 3.5. Cài đặt các phần mềm

- Chạy Docker và thiết lập chạy cùng hệ thống:
`sudo systemctl start docker sudo systemctl enable docker`
- Tạo group docker:

`sudo groupadd docker`

- Thêm người dùng vào group docker:

`sudo usermod -aG docker ${USER}`

- Khởi động lại Ubuntu, sau đó chạy đoạn lệnh sau để kiểm tra:

`docker run hello-world`

`docker ps -a`

```
vht@VHT2901:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:91fb4b941da273d5a3273b6d587d62d518300a6ad268b28628f74997b93171b2
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

vht@VHT2901:~$ docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED         STATUS
PORTS         NAMES
8afd5d947984   hello-world  "/hello"                16 seconds ago   Exited (0) 15 seconds ago
go
angry_hamilton
vht@VHT2901:~$
```

Hình 3.6. Thêm người dùng vào group docker và kiểm tra

- Cài đặt Text Editor (nếu cần): `sudo snap install code --classic`

3.2.2. Cài đặt các phần mềm hỗ trợ ngôn ngữ Hyperledger Fabric sử dụng

- Cài đặt NodeJS v18.20.4 và npm v6.14.17:

`curl -sL https://deb.nodesource.com/setup_18.x/ | sudo -E bash`

`—sudo apt-get install -y nodejs`

`sudo apt-get install gcc g++ make`

`sudo npm install npm@6.14.17 -g`

- Kiểm tra bằng câu lệnh: `node -v`
`npm -v`

```
vht@VHT2901:~$ node -v
v18.20.4
vht@VHT2901:~$ npm -v
6.14.17
```

Hình 3.7. Kiểm tra phiên bản Node Js với npm

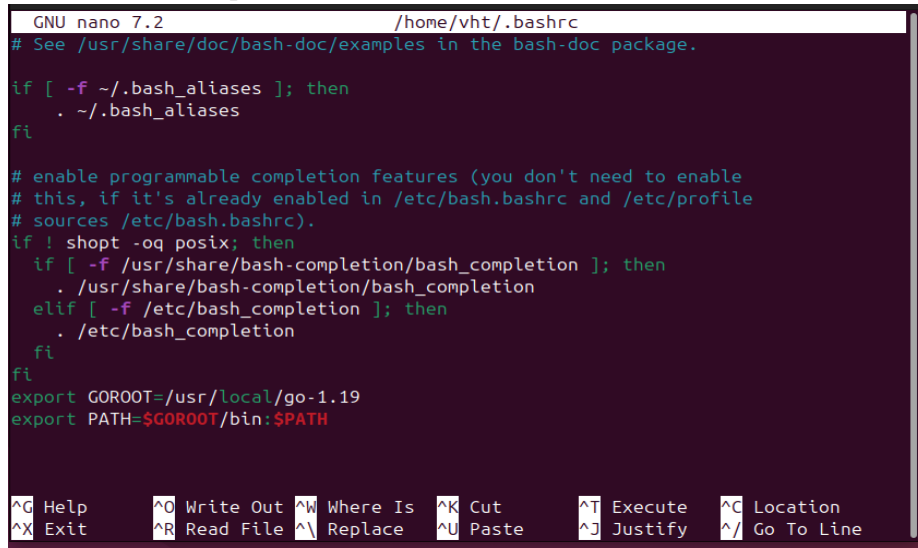
- Cài đặt Go 1.19.2
 - Xóa các phiên bản cũ (nếu có): `sudo apt remove 'golang-*`
 - Tải bản go 1.19.2 bằng wget: `wget https://go.dev/dl/go1.19.2.linux-amd64.tar.gz`
 - Giải nén và copy đến thư mục phát triển:

`tar xf go1.19.2.linux-amd64.tar.gz`

`sudo mv go /usr/local/go-1.19`

Thiết lập biến môi trường:

- Mở file bashrc: `sudo nano ~/.bashrc`
- Copy và thêm đoạn lệnh sau vào cuối file:
`export GOROOT=/usr/local/go-1.19`
`export PATH=$GOROOT/bin:$PATH`



```
GNU nano 7.2 /home/vht/.bashrc
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

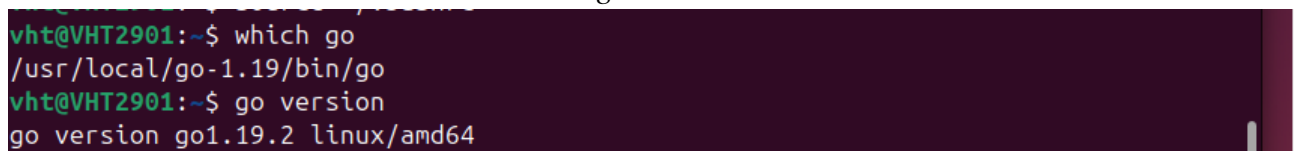
if [ -f ~/.bash_aliases ]; then
  . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
export GOROOT=/usr/local/go-1.19
export PATH=$GOROOT/bin:$PATH

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute  ^C Location
^X Exit      ^R Read File ^\ Replace  ^U Paste     ^J Justify  ^_ Go To Line
```

Hình 3.8. Copy đoạn lệnh

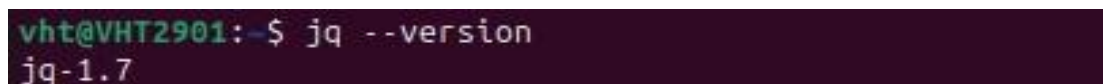
- Lưu thay đổi, thoát khỏi GNU nano và áp dụng thay đổi bằng câu lệnh:
`source ~/.bashrc`
- Kiểm tra đã cài đặt go thành công hay chưa:
`which go`
`go version`



```
vht@VHT2901:~$ which go
/usr/local/go-1.19/bin/go
vht@VHT2901:~$ go version
go version go1.19.2 linux/amd64
```

Hình 3.9. Kiểm tra cài đặt Go và phiên bản cài đặt

- Cài đặt Javascript: `sudo apt-get install jq`
- Kiểm tra phiên bản vừa cài đặt: `jq --version`



```
vht@VHT2901:~$ jq --version
jq-1.7
```

Hình 3.10. Phiên bản của JavaScript mới cài đặt

3.2.3. Tải và cài đặt Hyperledger Fabric

- Tạo root project cho người lập trình Go (thay github_userid bằng tên github củamình):
`mkdir -p`


```

$HOME/go/src/github.com/<github_userid>
cd $HOME/go/src/github.com/<github_userid>

vht@VHT2901:~$ mkdir -p $HOME/go/src/github.com/HTuan2901
vht@VHT2901:~$ cd $HOME/go/src/github.com/HTuan2901
vht@VHT2901:~/go/src/github.com/HTuan2901$

```

Hình 3.11. Tạo 1 root project cho người lập trình Go

- Tải fabric-samples, docker images và binaries:

```
curl -sSL https://bit.ly/2ysbOFE | bash -s
```

3.2.4. Tương tác với hệ thống mạng test-network.

Hệ thống mạng thử nghiệm test-network được xây dựng với mục tiêu tìm hiểu và làm quen với cách thức hoạt động của blockchain. Đoạn mã lệnh trong ‘./network.sh’ nhằm xây dựng một hệ thống mạng thử nghiệm test-network đơn giản. Hệ thống mạng này bao gồm hai peer, mỗi peer một Org, và một node Raft dịch vụ sắp xếp (Ordering Service node- Orderer).

a. Khởi tạo hệ thống mạng

Truy cập thư mục chứa test-network:

```
cd $HOME/go/src/github.com/<github_userid>/fabric-samples/test-network
```

Trước hết chạy đoạn lệnh sau để loại bỏ các container và artifact từ những lần chạy trước:

```
./network.sh down
```

```

letienduong@Ubuntu:~/go/src/github.com/letienduong/fabric-samples/test-network$ ./network.sh down
Using docker and docker-compose
Stopping network
Stopping orderer.example.com ... done
Stopping peer0.org1.example.com ... done
Stopping peer0.org2.example.com ... done
Removing orderer.example.com ... done
Removing peer0.org1.example.com ... done
Removing peer0.org2.example.com ... done

```

Hình 3.12. Tắt các container và artifact

Tiếp đó ta khởi tạo hệ thống mạng Fabric gồm 2 node peer, một node orderer thông qua câu lệnh:

```
./network.sh up
```

```

letienduong@Ubuntu:~/go/src/github.com/letienduong/fabric-samples/test-network$ ./network.sh up
Using docker and docker-compose
Starting nodes with CLI timeout of '5' tries and CLI delay of '3' seconds and using database 'leveldb' with crypto from 'cryptogen'
LOCAL_VERSION=v2.5.9
DOCKER_IMAGE_VERSION=v2.5.9
/home/letienduong/go/src/github.com/letienduong/fabric-samples/test-network/.../bin/cryptogen
Generating certificates using cryptogen tool
Creating Org1 Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-org1.yaml --output=organizations
org1.example.com
+ res=0
Creating Org2 Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-org2.yaml --output=organizations
org2.example.com
+ res=0
Creating Orderer Org Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-orderer.yaml --output=organizations
+ res=0
Generating CCP files for Org1 and Org2
Creating network "fabric_test" with the default driver
Creating volume "compose_orderer.example.com" with default driver
Creating volume "compose_peer0.org1.example.com" with default driver
Creating volume "compose_peer0.org2.example.com" with default driver
Creating peer0.org2.example.com ... done
Creating peer0.org1.example.com ... done
Creating orderer.example.com ... done

```

Hình 3.13. Khởi tạo hệ thống mạng

Quan sát xem hệ thống hiện có các thành phần nào, sử dụng câu lệnh:

docker ps -a

```
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
70fd72da2cc0	hyperledger/fabric-orderer:latest	"orderer"	50 seconds ago	Up 46 seconds	0.0.0.0:7050->7050/tcp,
3->9443/tcp	orderer.example.com				
62802e7e03d3	hyperledger/fabric-peer:latest	"peer node start"	50 seconds ago	Up 47 seconds	0.0.0.0:7051->7051/tcp,
peer0.org1.example.com					
86ce7d73527c	hyperledger/fabric-peer:latest	"peer node start"	50 seconds ago	Up 47 seconds	0.0.0.0:9051->9051/tcp,
peer0.org2.example.com					
678676ec8727	hello-world	"/hello"	2 weeks ago	Exited (0) 2 weeks ago	
priceless_bell					

Hình 3.14. Các container đang chạy

Mỗi node hay người dùng muốn tương tác với mạng Fabric đều phải thuộc một tổ chức (Org) thì mới được tham gia vào hệ thống mạng. Hai node peer thuộc hai tổ chức Org1 và Org2, cùng một node Orderer duy nhất duy trì dịch vụ sắp xếp của hệ thống mạng. Peer lưu trữ sổ cái blockchain và xác minh giao dịch trước khi chúng được gửi lên sổ cái. Peer chạy hợp đồng thông minh gồm những logic nghiệp vụ nhằm quản lý các tài sản trên sổ cái blockchain.

b. Tạo kênh

Kênh là lớp liên lạc bí mật giữa các thành viên trong hệ thống mạng được xác định cụ thể. Mỗi kênh một sổ cái blockchain riêng. Tổ chức nào được mời tham gia kênh thì các peer của tổ chức đó sẽ lưu trữ sổ cái kênh và sẽ xác minh giao dịch trên kênh đó. Để tạo kênh giữa Org1 và Org2, ta có thể thông qua network.sh. Tạo một kênh với tên mặc định:

./network.sh createChannel

```
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ ./network.sh createChannel
Living docker and docker-compose
Generating channel genesis block 'mychannel.block'
Creating channel 'mychannel'
If network is not up, starting nodes with 111 timeout of '1' s and 111 delay of '1' seconds and using network timeout
Starting up network
CORE_PEER0_ORG1=localhost:7051
CORE_PEER0_ORG2=localhost:9051
orderer.example.com is up-to-date
peer0.org1.example.com is up-to-date
peer0.org2.example.com is up-to-date
CONTAINER ID        IMAGE                      COMMAND                  CREATED             STATUS              PORTS
70fd72da2cc0       hyperledger/fabric-orderer:latest   "orderer"               12 minutes ago     Up 11 minutes      0.0.0.0:7050->7050/tcp,
7053->7053/tcp, 0.0.0.0:9443->9443/tcp, :::9443->9443/tcp   orderer.example.com
62802e7e03d3       hyperledger/fabric-peer:latest      "peer node start"       12 minutes ago     Up 12 minutes      0.0.0.0:7051->7051/tcp,
9444->9444/tcp      peer0.org1.example.com
86ce7d73527c       hyperledger/fabric-peer:latest      "peer node start"       12 minutes ago     Up 12 minutes      0.0.0.0:9051->9051/tcp,
5/tcp, :::9445->9445/tcp   peer0.org2.example.com
678676ec8727       hello-world                      "/hello"                 2 weeks ago        Exited (0) 2 weeks ago
priceless_bell
```

Hình 3.15. Câu lệnh tạo kênh

Sau đó hệ thống sẽ tạo khối hình thành (genesis block) của kênh:

```
Living docker and docker-compose
Generating channel genesis block 'mychannel.block'
/bin/configtxgen
/home/letinduong/go/src/github.com/letinduong/fabric-samples/test-network/./bin/configtxgen
+ '[' 0 -eq 1 ']'
+ configtxgen -profile ChannelUsingRaft -outputBlock ./channel-artifacts/mychannel.block -channelID mychannel
2024-11-10 01:28:28.933 +07 0000 INFO [common.tools.configtxgen] main -> Loading configuration
2024-11-10 01:28:28.952 +07 0000 INFO [common.tools.configtxgen.localconfig] completeInitialization -> orderer types: etcdraft
2024-11-10 01:28:28.965 +07 0000 INFO [common.tools.configtxgen.localconfig] completeInitialization -> Orderer.Etcdraft.Options unset, setting to tick_interval:"500ms" election_tick:1
0 heartbeat_tick:1 max_inflight_blocks:5 snapshot_interval_size:16777216
2024-11-10 01:28:28.965 +07 0000 INFO [common.tools.configtxgen.localconfig] Load -> Loaded configuration: /home/letinduong/go/src/github.com/letinduong/fabric-samples/test-network/
configtx/configtx.yaml
2024-11-10 01:28:28.967 +07 0000 INFO [common.tools.configtxgen] doOutputBlock -> Generating genesis block
2024-11-10 01:28:28.967 +07 0000 INFO [common.tools.configtxgen] doOutputBlock -> Creating application channel genesis block
2024-11-10 01:28:28.952 +07 0000 INFO [common.tools.configtxgen] doOutputBlock -> Writing genesis block
+ res=0
```

Hình 3.16. Tạo genesis block của kênh

Tiếp theo, join hai peer của Org1 và Org2 vào kênh:

```

Joining org1 peer to the channel...
Using organization 1
+ peer channel join -b ./channel-artifacts/mychannel.block
+ res=0
2024-11-18 01:28:27.278 +07 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2024-11-18 01:28:27.315 +07 0002 INFO [channelCmd] executeJoin -> Successfully submitted proposal to join channel
Joining org2 peer to the channel...
Using organization 2
+ peer channel join -b ./channel-artifacts/mychannel.block
+ res=0
2024-11-18 01:28:30.452 +07 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2024-11-18 01:28:30.489 +07 0002 INFO [channelCmd] executeJoin -> Successfully submitted proposal to join channel

```

Hình 3.17. Join peer của Org1 và Org2 vào kênh

```

lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ sudo docker exec peer0.org1.example.com peer channel list
[sudo] password for lettienduong:
2024-11-19 08:50:53.056 UTC 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
Channels peers has joined:
mychannel
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ sudo docker exec peer0.org2.example.com peer channel list
2024-11-19 08:51:20.327 UTC 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
Channels peers has joined:
mychannel

```

Hình 3.18. Kiểm tra lại danh sách kênh mà các peer đã join

Bước tiếp theo, các Anchor peer cho các Org sẽ được cài đặt vào kênh.

```

2024-11-18 01:28:31.021 +07 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2024-11-18 01:28:31.036 +07 0002 INFO [channelCmd] update -> Successfully submitted channel update
Anchor peer set for org 'Org1MSP' on channel 'mychannel'

```

Hình 3.19. Anchor peer cho Org1 được thêm vào kênh

```

2024-11-18 01:28:31.576 +07 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2024-11-18 01:28:31.599 +07 0002 INFO [channelCmd] update -> Successfully submitted channel update
Anchor peer set for org 'Org2MSP' on channel 'mychannel'

```

Hình 3.20. Anchor peer cho Org2 được thêm vào kênh

Câu lệnh thành công sẽ xuất hiện: *Channel 'mychannel' joined*

```
Channel 'mychannel' joined
```

Hình 3.21. Câu lệnh khi tạo kênh thành công

c. Triển khai chuỗi mã (chaincode) lên kênh.

Sau khi tạo kênh, khi muốn tương tác với sổ cái kênh ta phải sử dụng các hợp đồng thông minh. Các ứng dụng được chạy bởi các thành viên kênh có thể gọi hợp đồng thông minh để tạo, sửa đổi cũng như trao đổi các tài sản, hoặc truy vấn tài sản. Với Fabric, các hợp đồng thông minh được đóng gói và triển khai lên hệ thống mạng với tên gọi là chuỗi mã (chaincode). Trước khi chuỗi mã được triển khai lên kênh, các thành viên kênh cần chấp thuận định nghĩa chuỗi mã nhằm thiết lập quản lý chuỗi mã. Khi đủ số lượng tổ chức đồng ý, định nghĩa chuỗi mã sẽ được thêm lên kênh, và lúc đó chuỗi mã sẵn sàng có thể sử dụng.

Sau khi sử dụng câu lệnh `network.sh` để tạo kênh, ta có thể chạy chuỗi mã sử dụng câu lệnh:

```
./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-go -ccl go
```

Trong đó:

- `deployCC` là deploy chaincode (triển khai chuỗi mã).
- `-ccn` là chaincode name (tên chuỗi mã) chúng ta đặt ở đây là `basic`.

- ccp là chaincode path (đường dẫn chuỗi mã), chúng ta trở đến vị trí đang lưu trữ code chuỗi mã, ở đây là *fabric-samples/asset-transfer/basic/chaincode-go*.
- ccl là chaincode language (ngôn ngữ cho chuỗi mã), chúng ta sử dụng go.

```
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ ./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-go -ccl go
Using docker and docker-compose
Deploying chaincode on channel 'mychannel'
Executing with the following
- CHANNEL_NAME: mychannel
- CC_NAME: basic
- CC_SRC_PATH: ../asset-transfer-basic/chaincode-go
- CC_SRC_LANGUAGE: go
- CC_VERSION: 1.0
- CC_SEQUENCE: auto
- CC_END_POLICY: NA
- CC_COLL_CONFIG: NA
- CC_INIT_FCN: NA
- DELAY: 3
```

Hình 3.22. Chaincode đang được triển khai lên kênh

```
Installing chaincode on peer0.org1...
Using organization 1
+ peer lifecycle chaincode queryinstalled --output json
+ grep '^basic_1.0:398f97e8c434da57c3fcac5d5e0ef17ca51dc628cf2dd5b6af4247d6499303c7$'
+ jq -r 'try (.installed_chaincodes[].package_id)'
+ test 1 -ne 0
+ peer lifecycle chaincode install basic.tar.gz
+ res=0
2024-11-18 10:00:14.891 +07 0001 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Installed remotely: response:<status:200>
ca51dc628cf2dd5b6af4247d6499303c7\022\tbasic_1.0" >
2024-11-18 10:00:14.893 +07 0002 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Chaincode code package identifier: basic_1.0:398f97e8c434da57c3fcac5d5e0ef17ca51dc628cf2dd5b6af4247d6499303c7
Chaincode is installed on peer0.org1
```

Hình 3.23. Chaincode đã được cài đặt trên peer.org1

```
Install chaincode on peer0.org2...
Using organization 2
+ peer lifecycle chaincode queryinstalled --output json
+ jq -r 'try (.installed_chaincodes[].package_id)'
+ grep '^basic_1.0:398f97e8c434da57c3fcac5d5e0ef17ca51dc628cf2dd5b6af4247d6499303c7$'
+ test 1 -ne 0
+ peer lifecycle chaincode install basic.tar.gz
+ res=0
2024-11-18 10:00:55.961 +07 0001 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Installed remotely: response:<status:200>
ca51dc628cf2dd5b6af4247d6499303c7\022\tbasic_1.0" >
2024-11-18 10:00:55.962 +07 0002 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Chaincode code package identifier: basic_1.0:398f97e8c434da57c3fcac5d5e0ef17ca51dc628cf2dd5b6af4247d6499303c7
Chaincode is installed on peer0.org2
```

Hình 3.24. Chaincode đã được cài đặt trên peer.org2

```
Using organization 1
Checking the commit readiness of the chaincode definition on peer0.org1 on channel 'mychannel'...
Attempting to check the commit readiness of the chaincode definition on peer0.org1, Retry after 3 seconds.
+ peer lifecycle chaincode checkcommitreadiness --channelID mychannel --name basic --version 1.0 --sequence 1 --output json
+ res=0
{
  "approvals": {
    "Org1MSP": true,
    "Org2MSP": true
  }
}
Checking the commit readiness of the chaincode definition successful on peer0.org1 on channel 'mychannel'
```

Hình 3.25. Kiểm tra định nghĩa chaincode trên Org1

```
Using organization 2
Checking the commit readiness of the chaincode definition on peer0.org2 on channel 'mychannel'...
Attempting to check the commit readiness of the chaincode definition on peer0.org2, Retry after 3 seconds.
+ peer lifecycle chaincode checkcommitreadiness --channelID mychannel --name basic --version 1.0 --sequence 1 --output json
+ res=0
{
  "approvals": {
    "Org1MSP": true,
    "Org2MSP": true
  }
}
Checking the commit readiness of the chaincode definition successful on peer0.org2 on channel 'mychannel'
```

Hình 3.26. Kiểm tra định nghĩa chaincode trên Org2

```

Using organization 1
Using organization 2
+ peer lifecycle chaincode commit -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile /home/letienduong/go/src/github.com/organizations/ordererOrganizations/example.com/tlsca/tlsca.example.com-cert.pem --channelID mychannel --name basic --peerAddresses localhost:7051 --tlsRootCertFiles /home/letienduong/go/src/github.com/organizations/peerOrganizations/org1.example.com/tlsca/tlsca.org1.example.com-cert.pem --version 1.0 --sequence 1
+ res=0
2024-11-18 18:01:15.657 +07 0001 INFO [chaincodeCmd] ClientWait -> txid [e30fddb97a833aa62c9a22e85f201dc74dff7a561f438bff5ff51e2598cf708] committed
051
2024-11-18 18:01:15.878 +07 0002 INFO [chaincodeCmd] ClientWait -> txid [e30fddb97a833aa62c9a22e85f201dc74dff7a561f438bff5ff51e2598cf708] committed
051
Chaincode definition committed on channel 'mychannel'

```

Hình 3.27. Xác thực định nghĩa Chaincode trên kênh

```

Using organization 1
Querying chaincode definition on peer0.org1 on channel 'mychannel'...
Attempting to Query committed status on peer0.org1, Retry after 3 seconds.
+ peer lifecycle chaincode querycommitted --channelID mychannel --name basic
+ res=0
Committed chaincode definition for chaincode 'basic' on channel 'mychannel':
Version: 1.0, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vsc, Approvals: [Org1MSP: true, Org2MSP: true]
Query chaincode definition successful on peer0.org1 on channel 'mychannel'
Using organization 2
Querying chaincode definition on peer0.org2 on channel 'mychannel'...
Attempting to Query committed status on peer0.org2, Retry after 3 seconds.
+ peer lifecycle chaincode querycommitted --channelID mychannel --name basic
+ res=0
Committed chaincode definition for chaincode 'basic' on channel 'mychannel':
Version: 1.0, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vsc, Approvals: [Org1MSP: true, Org2MSP: true]
Query chaincode definition successful on peer0.org2 on channel 'mychannel'

```

Hình 3.28. Định nghĩa truy vấn chaincode trên 2 Org

d. Tương tác với hệ thống mạng thông qua chuỗi mã.

Sau khi triển khai hệ thống mạng, ta cần dùng Command-line Interface để peer tương tác. Trước hết phải thiết lập biến môi trường để sử dụng Fabric binary. Truy cập vào thư mục *test-network*. Ở đây ta xác định câu lệnh peer trong thư mục *fabric-samples/bin* bằng cách thêm đường dẫn vào biến *PATH*. Ta cũng cần thiết lập biến *FABRIC_CFG_PATH* để trỏ vào tệp cấu hình *core.yaml*:

```

export PATH=${PWD}/../bin:$PATH
export FABRIC_CFG_PATH=${PWD}/../config/

```

```

letienduong@Ubuntu:~/go/src/github.com/letienduong/fabric-samples/test-network$ export PATH=${PWD}/../bin:$PATH
letienduong@Ubuntu:~/go/src/github.com/letienduong/fabric-samples/test-network$ export FABRIC_CFG_PATH=${PWD}/../config/

```

Hình 3.29. Thiết lập môi trường để sử dụng biến Fabric binary

Tiếp đó ta thiết lập các biến môi trường để tương tác với vai trò là peer CLI của Org1:

- `export CORE_PEER_TLS_ENABLED=true`
- `export CORE_PEER_LOCALMSPID="Org1MSP"`
- `export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt.`
- `export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp`
- `export CORE_PEER_ADDRESS=localhost:7051`

```

letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ export PATH=${PWD}/../bin:$PATH
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ export FABRIC_CFG_PATH=${PWD}/../config/
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ export CORE_PEER_TLS_ENABLED=true
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ export CORE_PEER_LOCALMSPID="Org1MSP"
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ export CORE_PEER_ADDRESS=localhost:7051

```

Hình 3.30. Thiết lập biến môi trường để tương tác với vai trò peer của Org1

Biến môi trường `CORE_PEER_MSPCONFIGPATH` và `CORE_PEER_ROOTCERT_FILE` trở vào các thành phần mật mã của Org1 trong thư mục `organizations`.

Trong mạng `test-network`, một hợp đồng thông minh đã được viết sẵn để ta có thể tương tác với mạng, nằm trong thư mục `chaincode-go`. Các chức năng có hợp đồng thông minh:

- **InitLedger:** Khởi tạo một sổ cái có tài sản cho trước.

```

func (s *SmartContract) InitLedger(ctx contractapi.TransactionContextInterface) error {
    assets := []Asset{
        {ID: "asset1", Color: "blue", Size: 5, Owner: "Tomoko", AppraisedValue: 300},
        {ID: "asset2", Color: "red", Size: 5, Owner: "Brad", AppraisedValue: 400},
        {ID: "asset3", Color: "green", Size: 10, Owner: "Jin Soo", AppraisedValue: 500},
        {ID: "asset4", Color: "yellow", Size: 10, Owner: "Max", AppraisedValue: 600},
        {ID: "asset5", Color: "black", Size: 15, Owner: "Adriana", AppraisedValue: 700},
        {ID: "asset6", Color: "white", Size: 15, Owner: "Michel", AppraisedValue: 800},
    }

    for _, asset := range assets {
        assetJSON, err := json.Marshal(asset)
        if err != nil {
            return err
        }

        err = ctx.GetStub().PutState(asset.ID, assetJSON)
        if err != nil {
            return fmt.Errorf("failed to put to world state. %v", err)
        }
    }

    return nil
}

```

Hình 3.31. InitLedger

- **CreateAsset:** Tạo một tài sản

```

// CreateAsset issues a new asset to the world state with given details.
func (s *SmartContract) CreateAsset(ctx contractapi.TransactionContextInterface, id string, color string, size int, owner string, appraisedValue int) error {

```

Hình 3.32. CreateAsset

- **ReadAsset:** Đọc trạng thái của tài sản được tìm kiếm theo ID

```

// ReadAsset returns the asset stored in the world state with given id.
func (s *SmartContract) ReadAsset(ctx contractapi.TransactionContextInterface, id string) (*Asset, error) {

```

Hình 3.33. ReadAsset

- **UpdateAsset:** Cập nhật trạng thái của tài sản theo các đặc điểm được cung cấp

```

// UpdateAsset updates an existing asset in the world state with provided parameters.
func (s *SmartContract) UpdateAsset(ctx contractapi.TransactionContextInterface, id string, color string, size int, owner string, appraisedValue int) error {

```

Hình 3.34. UpdateAsset

- **DeleteAsset:** Xóa một tài sản theo ID được yêu cầu

```

// DeleteAsset deletes an given asset from the world state.
func (s *SmartContract) DeleteAsset(ctx contractapi.TransactionContextInterface, id string) error {

```

Hình 3.35. DeleteAsset

- **AssetExists:** Kiểm tra sự tồn tại của một tài sản

Nếu tài sản được kiểm tra có tồn tại thì hàm sẽ trả về *true*, nếu không tồn tại sẽ trả về *false*.

```
// AssetExists returns true when asset with given ID exists in world state
func (s *SmartContract) AssetExists(ctx contractapi.TransactionContextInterface, id string) (bool, error) {
    assetJSON, err := ctx.GetStub().GetState(id)
    if err != nil {
        return false, fmt.Errorf("failed to read from world state: %v", err)
    }
    return assetJSON != nil, nil
}
```

Hình 3.36. AssetExists

- **TransferAsset:** Cập nhật chủ mới cho một tài sản.

```
// TransferAsset updates the owner field of asset with given id in world state, and returns the old owner.
func (s *SmartContract) TransferAsset(ctx contractapi.TransactionContextInterface, id string, newOwner string) (string, error) {
```

Hình 3.37. TransferAsset

- **GetAllAssets:** Trả về trạng thái của mọi tài sản trong sổ cái.

```
// GetAllAssets returns all assets found in world state
func (s *SmartContract) GetAllAssets(ctx contractapi.TransactionContextInterface) ([]*Asset, error) {
```

Hình 3.38. GetAllAssets

Ngoài các chức năng trên, chúng ta có thể viết thêm các chức năng phức tạp hơn vào hợp đồng thông minh để sử dụng với nhiều mục đích khác.

- Các thao tác với sổ cái bao gồm đọc dữ liệu, ghi dữ liệu, cập nhật dữ liệu, xóa dữ liệu và kiểm tra tình trạng của sổ cái. Các thao tác này được thực hiện dựa trên hai câu lệnh: *peer invoke* và *peer query*.
- Lệnh *Invoke* là lệnh thực thi giao dịch nhằm thay đổi trạng thái của sổ cái. Ví dụ như chuyển tài sản, cập nhật thông tin hoặc xóa một tài sản. Vì thế invoke yêu cầu các bước xác nhận và đồng thuận. Chính sách chứng thực trên chuỗi mã asset-transfer (basic) cần giao dịch phải được ký bởi Org1 và Org2. Vì vậy chuỗi mã cần thêm tham số trở cả *peer0.org1.example.com* và *peer0.org2.example.com* thông qua flag. *-peerAddresses*.
- Lệnh *query* là các lệnh truy vấn, không làm thay đổi trạng thái của sổ cái, không cần gửi qua orderer hoặc yêu cầu sự đồng thuận của các peer khác nên chỉ cần gửi tới một peer bất kỳ trong mạng để lấy dữ liệu, không cần thêm địa chỉ cụ thể như lệnh invoke.

Dưới đây là một số thao tác cơ bản với sổ cái thông qua hợp đồng thông minh:

1. Khởi tạo sổ cái có tài sản.

```
peer chaincode invoke
```

```
-o localhost:7050
```

```
--ordererTLSHostnameOverride orderer.example.com
```

```
--tls
```

```
--cafile
```

```
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlsacerts/tlsca.example.com-cert.pem"
```

```
-C mychannel
```

```
-n basic
--peerAddresses localhost:7051
--tlsRootCertFiles
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt"

--peerAddresses localhost:9051
--tlsRootCertFiles
"${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt"
-c '{"function": "InitLedger", "Args": []}'
```

Trong đó:

- o là flag xác định dịch vụ sắp xếp endpoint.
- ordererTLSHostnameOverride là chuỗi hostname thay thế khi xác minh kết nối TLS tới orderer.
- tls: liên lạc tới endpoint orderer bằng TLS.
- cafile: đường dẫn đến tệp chứa cert tin cậy được mã hoá PEM cho endpoint orderer.
- C: tên kênh thực hiện chaincode.
- n: tên chaincode.
- peerAddresses: địa chỉ của peer cần kết nối tới
- tlsRootCertFiles: đường dẫn tới tệp root cert TLS của peer kết nối tới. Thứ tự và số lượng cert cụ thể phải phù hợp với --peerAddresses. Dùng nếu kích hoạt flag --tls.
- c: (constructor) Chuỗi gửi tới chaincode dưới dạng JSON (mặc định là "{}")

```
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function": "InitLedger", "Args": []}'
2024-11-10 12:56:22.947 +07 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
```

Hình 3.39. Khởi tạo sổ cái có tài sản

Khi khởi tạo thành công CLI sẽ hiển thị output:

Chaincode invoke successfull. result: status: 200

2. Truy vấn trạng thái của tất cả các tài sản.

peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'

```
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'
[{"AppraisedValue":300,"Color":"blue","ID":"asset1","Owner":"Tomoko","Size":5}, {"AppraisedValue":400,"Color":"red","ID":"asset2","Owner":"Brad","Size":5}, {"AppraisedValue":500,"Color":"green","ID":"asset3","Owner":"Jin Soo","Size":10}, {"AppraisedValue":600,"Color":"yellow","ID":"asset4","Owner":"Max","Size":10}, {"AppraisedValue":700,"Color":"black","ID":"asset5","Owner":"Adriana","Size":15}, {"AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Michel","Size":15}]
```

Hình 3.40. Truy vấn trạng thái của tài sản

3. Đọc trạng thái của một tài sản.

```
peer chaincode query -C mychannel -n basic -c
'{"Args":["ReadAsset","ID"]}'
peer chaincode query -C mychannel -n basic -c
'{"Args":["AssetExists","ID"]}'
```

```

lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["ReadAsset","asset6"]}'
{"AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Michel","Size":15}
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["AssetExists","asset1"]}'
true
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["AssetExists","asset7"]}'
false

```

Hình 3.41. Đọc trạng thái của tài sản asset6

⇒ Khi thực hiện lệnh này với asset1 sẽ cho kết quả true vì asset1 có tồn tại, còn asset7 sẽ trả về false vì asset7 không tồn tại.

4. Tạo tài sản mới.

peer chaincode invoke

-o localhost:7050

--ordererTLSHostnameOverride orderer.example.com

--tls

--cafile

"\${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem"

-C mychannel -n basic

--peerAddresses localhost:7051

--tlsRootCertFiles

"\${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt"

--peerAddresses localhost:9051

--tlsRootCertFiles

"\${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt"

-c '{"function":"CreateAsset","Args":["ID","Color","Size","Owner","Value"]}'

```

lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"CreateAsset","Args":["asset7","black",7,"Tom","1300"]}'
[2024-11-18 13:36:04.133 +07 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'
[{"AppraisedValue":300,"Color":"blue","ID":"asset1","Owner":"Tomoko","Size":5}, {"AppraisedValue":400,"Color":"red","ID":"asset2","Owner":"Brad","Size":5}, {"AppraisedValue":500,"Color":"green","ID":"asset3","Owner":"Jin Soo","Size":10}, {"AppraisedValue":600,"Color":"yellow","ID":"asset4","Owner":"Max","Size":10}, {"AppraisedValue":700,"Color":"black","ID":"asset5","Owner":"Adriana","Size":15}, {"AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Michel","Size":15}, {"AppraisedValue":1300,"Color":"black","ID":"asset7","Owner":"Tom","Size":7}]

```

Hình 3.42. Tài sản 7 đã được tạo thành công

⇒ Ta có thể thấy tài sản 7 đã được thêm vào sổ cái thành công với chủ sở hữu là Tom.

5. Cập nhật thông tin của tài sản

Câu lệnh giống phần tạo tài sản mới, thay thế phần function:

'{"function":"UpdateAsset","Args":["ID","Color","Size","Owner","Value"]}'


```

letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHost
nameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/
tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/pee
rOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organ
izations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"UpdateAsset","Args":["asset7","Whi
te","12","Michel","200"]}'
2024-11-18 13:57:55.375 +07 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["
ReadAsset","asset7"]}'
{"AppraisedValue":200,"Color":"White","ID":"asset7","Owner":"Michel","Size":12}

```

Hình 3.43. Trạng thái của tài sản 7 đã được cập nhật

6. Xóa tài sản.

Câu lệnh giống phần tạo tài sản mới, thay thế phần function:

```
'{"function":"DeleteAsset","Args":["ID"]}'
```

```

letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHost
nameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/
tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/pee
rOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organ
izations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"DeleteAsset","Args":["asset6"]}'
2024-11-18 14:02:49.993 +07 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["
AssetExists","asset6"]}'
false

```

Hình 3.44. Xóa tài sản 6

Sau khi xóa tài sản 6, dùng câu lệnh AssetExists để kiểm tra sự tồn tại của tài sản 6, kết quả trả về là false, tức là tài sản 6 đã không còn tồn tại.

7. Đổi quyền sở hữu tài sản.

Câu lệnh giống phần cập nhật thông tin, thay phần function:

```
'{"function":"TransferAsset","Args":["ID","Owner"]}'
```

```

letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHost
nameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/
tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/pee
rOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organ
izations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"TransferAsset","Args":["asset7","C
hristopher"]}'
2024-11-18 15:02:26.865 +07 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200 payload:"
Michel"
letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["
ReadAsset","asset7"]}'
{"AppraisedValue":200,"Color":"White","ID":"asset7","Owner":"Christopher","Size":12}

```

Hình 3.45. Thay đổi người sở hữu tài sản 7

Sau khi đổi người sở hữu tài sản, ta kiểm tra trạng thái của tài sản 7, chủ sở hữu đã được đổi thành Christopher mà trước đó tài sản 7 thuộc quyền sở hữu của Tom.

Ngoài ra, ta có thể kiểm tra tình trạng của sổ cái bằng câu lệnh:

```
peer channel getinfo -c mychannel
```

```

letinduong@Ubuntu:~/go/src/github.com/letinduong/fabric-samples/test-network$ peer channel getinfo -c mychannel
2024-11-18 18:26:23.871 +07 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
Blockchain info: {"height":13,"currentBlockHash":"EzFRFQhrj6MeHkCf8oLS+R+TFXDSJTLi1UWKRWtSVN0=", "previousBlockHash":"1jaSqHuC3Jgyu02JkD
qJPvUCinLLa0r4qInIoT6V6A="}

```

Hình 3.46. Kiểm tra tình trạng của sổ cái.

Kết quả trả về là thông tin của Blockchain: số lượng block hiện tại trong blockchain là 13, bao gồm cả genesis block. Đồng thời bao gồm cả mã băm của khối hiện tại và của khối trước. Khi giao dịch thành công, số lượng khối trong blockchain sẽ tăng thêm.

Ta có thể đổi sang vai trò peer của Org2 để thực hiện. Điều này thể hiện tính phân tán của sổ cái, đồng thời cũng đảm bảo tính minh bạch, toàn vẹn và nhất quán của công nghệ blockchain. Trước hết cần thiết lập môi trường để hoạt động như peer Org2, thực hiện như với Org1 nhưng thay Org1 thành Org2.

```

lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ export CORE_PEER_TLS_ENABLED=true
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ export CORE_PEER_LOCALMSPID="Org2MSP"
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ export CORE_PEER_TLS_ROOTCERT_FILE=$(PWD)/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ export CORE_PEER_MSPCONFIGPATH=$(PWD)/organizations/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ export CORE_PEER_ADDRESS=localhost:9051
lettienduong@Ubuntu:~/go/src/github.com/lettienduong/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'
[{"AppraisedValue":300,"Color":"blue","ID":"asset1","Owner":"Tonoko","Size":5}, {"AppraisedValue":400,"Color":"red","ID":"asset2","Owner":"Brad","Size":5}, {"AppraisedValue":500,"Color":"green","ID":"asset3","Owner":"Jin Soo","Size":10}, {"AppraisedValue":600,"Color":"yellow","ID":"asset4","Owner":"Max","Size":10}, {"AppraisedValue":700,"Color":"black","ID":"asset5","Owner":"Adriana","Size":15}, {"AppraisedValue":200,"Color":"White","ID":"asset7","Owner":"Christopher","Size":12}]

```

Hình 3.47. Thiết lập môi trường và truy vấn với vai trò peer của Org2

Qua đó, ta hiểu được quá trình giao dịch trong Hyperledger Fabric được thực hiện qua 5 bước:

Bước 1: Đề xuất giao dịch (Transaction Proposal)

Ứng dụng khách (Client Application) gửi một đề xuất giao dịch đến một hoặc nhiều peer. Đề xuất này chứa yêu cầu về việc thay đổi trạng thái của sổ cái.

Bước 2: Phê duyệt giao dịch (Endorsement)

Các peer nhận đề xuất và thực hiện kiểm tra tính hợp lệ dựa trên hợp đồng thông minh (Smart Contract). Nếu hợp lệ, peer sẽ tạo ra một phản hồi có chữ ký số và gửi lại cho ứng dụng khách.

Bước 3: Đồng thuận (Ordering)

Ứng dụng khách gửi các phản hồi được phê duyệt tới orderer. Orderer sắp xếp giao dịch vào thứ tự và tạo thành các block (khối).

Bước 4: Xác thực giao dịch (Validation)

Các peer nhận block từ orderer, xác thực từng giao dịch dựa trên chính sách và hợp đồng thông minh. Giao dịch không hợp lệ sẽ bị loại bỏ.

Bước 5: Ghi vào sổ cái (Commit)

Các giao dịch hợp lệ được ghi vào sổ cái (Ledger), cập nhật trạng thái hiện tại trong World State. Ứng dụng khách nhận được thông báo về kết quả giao dịch.

KẾT LUẬN

Blockchain, công nghệ phân tán và minh bạch, đã trở thành một trong những bước đột phá quan trọng trong lĩnh vực công nghệ thông tin trong những năm gần đây. Khả năng tạo ra một hệ thống lưu trữ dữ liệu an toàn, không thể thay đổi và không cần sự trung gian đã làm cho blockchain trở thành nền tảng lý tưởng cho nhiều ứng dụng, đặc biệt là trong lĩnh vực tài chính, chuỗi cung ứng, y tế và quản lý dữ liệu. Sự phát triển của blockchain đã tạo ra những thay đổi đáng kể trong cách thức giao dịch và quản lý tài sản.

Một trong những ứng dụng nổi bật của blockchain là trong lĩnh vực tài chính với các loại tiền điện tử như Bitcoin - là ứng dụng đầu tiên của blockchain, và cho đến nay vẫn là một trong những loại tiền điện tử phổ biến nhất. Sự phát triển của Bitcoin đã chứng minh rằng blockchain có thể cung cấp một hệ thống giao dịch an toàn và phi tập trung mà không cần đến sự can thiệp của các tổ chức tài chính trung gian. Bitcoin không chỉ là một loại tiền điện tử, mà còn là một hệ sinh thái blockchain hoàn chỉnh. Được tạo ra bởi một cá nhân hay nhóm người dưới biệt danh Satoshi Nakamoto, Bitcoin đã chứng minh rằng một đồng tiền không cần sự quản lý của chính phủ hoặc ngân hàng vẫn có thể hoạt động hiệu quả. Mặc dù giá trị của Bitcoin có sự biến động lớn, nhưng nó vẫn là một minh chứng cho tiềm năng của blockchain trong việc thay đổi cách thức tiền tệ và giao dịch toàn cầu.

Tóm lại, blockchain đã và đang thay đổi cách thức giao dịch và quản lý thông tin trên toàn cầu. Các ứng dụng của blockchain, từ Bitcoin đến các nền tảng doanh nghiệp như Hyperledger Fabric, chứng minh rằng công nghệ này không chỉ là một xu hướng công nghệ mà còn là một giải pháp thực tiễn cho nhiều thách thức trong việc lưu trữ và bảo mật dữ liệu. Việc tiếp tục nghiên cứu và thử nghiệm blockchain sẽ mở ra những cơ hội mới cho các ngành công nghiệp, đồng thời giúp tối ưu hóa quy trình và giảm thiểu chi phí. Sự phát triển không ngừng của blockchain hứa hẹn sẽ tạo ra những thay đổi sâu rộng trong nền kinh tế số trong tương lai gần.

TÀI LIỆU THAM KHẢO

- [1] GeeksforGeeks (29/8/2024). *Blockchain Structures*. Được truy lục từ:
<https://www.geeksforgeeks.org/blockchain-structure/>
- [2] Anastasiia Lastovetska (12/11/2021). *Blockchain Architecture Explained: How It Works & How to Build*. Được truy lục từ:
<https://mlsdev.com/blog/156-how-to-build-your-own-blockchain-architecture>
- [3] GeeksforGeeks (1/7/2024). *How does the Blockchain Work?*. Được truy lục từ:
<https://www.geeksforgeeks.org/how-does-the-blockchain-work/>
- [4] Vy Bùi (24/10/2024). *Node là gì? 5 Bước chạy Node trong Blockchain*. Được truy lục từ:
<https://coin98.net/node-la-gi>
- [5] Tạp chí Bitcoin (8/1/2019). *Tìm hiểu về 'Block' trong Blockchain*. Được truy lục từ:
<https://tapchibitcoin.io/tat-tan-tat-ve-block-trong-blockchain.html>
- [6] Mikel Garcia (6/12/2023). *What is a Blockchain node: Insights and Applications*. Được truy lục từ: <https://www.diadata.org/blog/post/what-is-a-blockchain-node/>
- [7] Webisoft Blog (13/2/2024). *16 Disadvantages of Blockchain: Limitations and Challenges - Webisoft Blog*. Được truy lục từ: <https://webisoft.com/articles/disadvantages-of-blockchain/>
- [8] Dylan J. Yaga, Peter M. Mell, Nik Roby, Karen Scarfone (2018). *Blockchain Technology Overview*. Được truy lục từ: <https://nvlpubs.nist.gov/nistpubs/ir/2018/NIST.IR.8202.pdf>
- [9] Hrchannels (27/2/2023). *08 nền tảng Blockchain uy tín mà bạn nên biết*. Được truy lục từ:
<https://hrchannels.com/uptalent/08-nen-tang-blockchain-uy-tin.html>
- [10] pazz2211 (2022). *Hướng dẫn cài đặt HyperLedger Fabric*. Được truy lục từ:
<https://github.com/pazz2211/fabric/blob/main/README.md>
- [11] Trần Đức Trung (25/6/2024). *Công nghệ Blockchain là gì? Blockchain có phải là tiền ảo không?*. Được truy lục từ: <https://vietnix.vn/blockchain-la-gi/>
- [12] Washija Kazim (28/5/2024). *26 Top Blockchain Applications and Use Cases in 2024*. Được truy lục từ: <https://learn.g2.com/blockchain-applications#retail>
- [13] Nguyễn Tùng Anh (16/11/2022). *Hướng dẫn thao tác với mạng test-network*. Được truy lục từ: <https://youtu.be/DYyUMUREHEs?si=OOaPu9VMwfcTwvOV>
- [14] Wikipedia. *Hyperledger*. Được truy lục từ: <https://en.wikipedia.org/wiki/Hyperledger>
- [15] Wikipedia. *Bitcoin*. Được truy lục từ: <https://vi.wikipedia.org/wiki/Bitcoin>