ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO MÔN
CÔNG NGHỆ CHUỖI KHỐI

CAO HỌC KHÓA 32 - 2022

Lab 3: Solidity Smart Contract Development

Giáo viên hướng dẫn: **PGS. TS. Nguyễn Đình Thúc**

Học viên thực hiện: **22C11026 – Lâm Phạm Bá Tùng**

22C11022 – Nguyễn Trương Tấn Sâm

22C11049 – Trương Công Triều

TPHCM - 1/2024

MỤC LỤC

1 Giới thiệu

Trong kỷ nguyên hiện đại, Blockchain đã nhanh chóng trở thành một xu hướng không thể bỏ qua. Phát triển hợp đồng thông minh bằng ngôn ngữ lập trình Solidity đã trở thành chìa khóa quan trọng cho việc khai thác và sử dụng nền tảng Ethereum, cũng như nhiều loại blockchain khác, mở ra những cơ hội và ứng dụng mới trong nhiều lĩnh vực khác nhau. Đồ án này tập trung vào việc nghiên cứu và quá trình thiết lập một Smart Contract đơn giản trên mạng Ethereum, qua đó khám phá tiềm năng và ứng dụng thực tế của công nghệ blockchain.

Thư mục đồ án có cấu trúc như sau:

```
|   report.pdf
\---code
    |   vote.sol
```

Trong đó:

- report.pdf: Báo cáo chi tiết về đồ án.
- code: Thư mục chính chứa mã nguồn đồ án.
- vote.sol: File code Solidity của smart contract

2 Tổng quan về lý thuyết

Chương này sẽ trình bày một cách tổng quan về Ethereum, Solidity và Smart contract

2.1 Ethereum

Ethereum được giới thiệu lần đầu vào năm 2013 bởi Vitalik Buterin, một lập trình viên và đồng sáng lập của Bitcoin Magazine. Ethereum khác biệt so với Bitcoin ở chỗ nó không chỉ là một mạng lưới thanh toán mà còn là một nền tảng cho các ứng dụng phi tập trung (dApps) và các hợp đồng thông minh (smart contracts). Ethereum chính thức hoạt động từ tháng 7 năm 2015. [1]

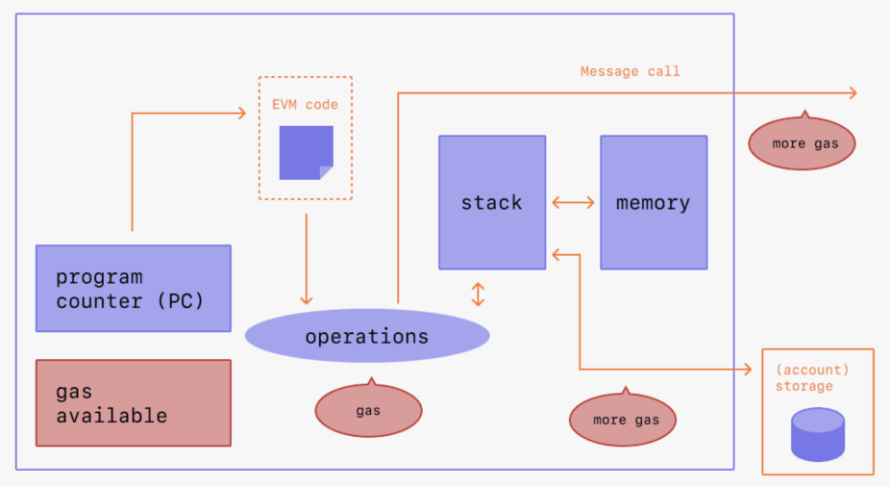
2.1.1 Đặc điểm kỹ thuật của Ethereum

Cấu trúc và cách hoạt động của blockchain Ethereum có thể được mô tả qua ba thành phần chính:

- **Các khối trong Blockchain Ethereum**: Mỗi khối trên chuỗi Ethereum chứa một danh sách các giao dịch đã được xác nhận.
- **Giao dịch trên Ethereum**: Giao dịch trên Ethereum không chỉ giới hạn ở việc chuyển đổi ETH từ tài khoản này sang tài khoản khác. Chúng còn bao gồm các hoạt động phức tạp hơn như tương tác với hợp đồng thông minh, tạo hoặc thực hiện các

hành động trong dApps. Mỗi giao dịch yêu cầu một lượng "gas" nhất định, là phí để thực hiện và xử lý giao dịch trên mạng lưới.

- **Ethereum Virtual Machine (EVM)**: Ethereum có chứa một loại máy tính ảo gọi là Ethereum Virtual Machine (EVM). EVM hoạt động như một máy tính toàn cầu, với mọi máy tính (nút) tham gia vào mạng Ethereum đều lưu trữ bản sao và thống nhất trạng thái của nó. Trong EVM, các hợp đồng thông minh và giao dịch trên mạng Ethereum được thực thi. Các nút trong mạng Ethereum có nhiệm vụ lưu trữ thông tin và xác nhận các trạng thái của EVM, giúp đảm bảo tính toàn vẹn và đồng bộ của hệ thống. [2]



Hình 2.1. Cấu trúc EVM. Nguồn [2]

2.1.2 Ether – Đồng tiền mã hóa của Ethereum

Ether (ETH) là đồng tiền mã hóa của Ethereum và đóng vai trò trung tâm trong việc duy trì và vận hành mạng lưới Ethereum. ETH được sử dụng để trả phí giao dịch trên mạng Ethereum. Mỗi giao dịch trên Ethereum, bao gồm cả việc thực thi hợp đồng thông minh, yêu cầu một lượng "gas" nhất định. Gas này được tính bằng ETH và phụ thuộc vào độ phức tạp và tài nguyên tính toán cần thiết cho giao dịch

Trong cơ chế đồng thuận Proof of Stake (PoS), ETH được sử dụng như một phương tiện để đảm bảo an ninh mạng. Người dùng có thể stake ETH của họ để trở thành người xác nhận (validator) giao dịch. Điều này giúp bảo vệ mạng lưới khỏi các hành vi gian lận do việc họ phải cam kết một lượng lớn ETH như một hình thức bảo đảm. [2]

2.2 Solidity

Solidity [3] là ngôn ngữ lập trình hướng đối tượng với tác vụ chính là dùng cho việc phát triển smart contracts trên Ethereum. Đây là ngôn ngữ hướng hợp đồng, nhằm tạo lập hợp đồng thông minh chịu trách nhiệm lưu trữ logic lập trình xử lý giao dịch trên blockchain. Vì vậy nó đã trở thành công cụ quan trọng để triển khai các dự án blockchain đa dạng, từ tài chính phi tập trung đến quản lý chuỗi cung ứng và nhiều ứng dụng khác.

Solidity là một ngôn ngữ lập trình sử dụng cú pháp dạng ngoặc nhọn (curly-bracket) được thiết kế để tương tác với EVM. Nó có kiểu dữ liệu tĩnh, hỗ trợ kế thừa, thư viện và nhiều tính năng khác.

Solidity lấy cảm hứng từ nhiều ngôn ngữ lập trình nổi tiếng.

- Ảnh hưởng từ C++: Solidity chịu ảnh hưởng sâu sắc nhất từ C++, thể hiện qua cú pháp khai báo biến, vòng lặp for, chuyển đổi kiểu, cũng như nhiều chi tiết khác.
- Ảnh hưởng từ JavaScript (ban đầu): Ở giai đoạn đầu của ngôn ngữ, Solidity từng một phần được ảnh hưởng bởi JavaScript. Điều này thể hiện qua việc sử dụng phạm vi biến theo cấp độ hàm và từ khóa var. Tuy nhiên, ảnh hưởng từ JavaScript đã giảm đi từ phiên bản 0.4.0 trở đi. Hiện tại, điểm tương đồng chính với JavaScript là việc định nghĩa hàm sử dụng từ khóa function. Solidity cũng hỗ trợ cú pháp và ngữ nghĩa nhập (import) tương tự như trong JavaScript.
- Ảnh hưởng từ Python: Các modifiers của Solidity được thêm vào với mục tiêu mô phỏng decorators của Python, nhưng với một chức năng hạn chế hơn. Ngoài ra, việc kế thừa đa hình và từ khóa super được lấy từ Python, cũng như cú pháp và ngữ nghĩa tổng quát của việc gán và sao chép giữa các kiểu giá trị và tham chiếu.

Solidity được áp dụng trong nhiều trường hợp khác nhau như: bỏ phiếu, đấu giá mù, mua hàng từ xa an toàn…

2.3 Smart Contracts

Smart contracts là những chương trình máy tính hoạt động một cách tự động trên mạng blockchain, như Ethereum. Chúng được sử dụng để tự động hóa việc thực thi các thỏa thuận kỹ thuật số mà không cần đến sự can thiệp của bên thứ ba.

2.3.1 Nguyên tắc hoạt động của Smart Contracts

Nguyên tắc hoạt động của smart contracts trên Ethereum:

- **Tự động thực thi**: Một smart contract được lập trình để tự động thực thi các hành động hoặc chức năng khi các điều kiện được xác định trong mã của nó được đáp ứng. Điều này có nghĩa là không cần sự can thiệp hoặc giải thích từ con người để thực hiện các giao dịch hoặc hành động được lập trình trong hợp đồng.
- **Chính xác và dự đoán được kết quả**: Do các điều kiện và hành động được xác định rõ ràng trong mã của hợp đồng thông minh, kết quả của mỗi giao dịch hoặc tương tác với hợp đồng là dự đoán được và chính xác. Điều này loại bỏ các khả năng của sự sai lệch do giải thích con người hoặc lỗi trong việc thực thi thủ công.
- **Độc lập và không cần trung gian**: Smart contracts thực hiện các giao dịch một cách độc lập mà không cần đến sự trung gian của các bên thứ ba như ngân hàng, luật sư, hoặc các tổ chức tài chính.
- **Tự động hóa quy trình**: Các quy trình được tự động hóa thông qua smart contracts, giúp cải thiện hiệu quả và giảm thiểu rủi ro do lỗi con người trong các giao dịch và thỏa thuận.
- **Tính minh bạch**: Mọi giao dịch thực hiện thông qua smart contracts được ghi lại trên blockchain, cho phép các bên liên quan có thể kiểm tra và xác minh lịch sử giao dịch một cách minh bạch và đáng tin cậy.

2.3.2 Cấu trúc xây dựng Smart Contracts

Quy Trình Xây Dựng Smart Contracts trên mạnh Ethereum gồm các bước sau:

- **Viết code**: Sử dụng Solidity để viết mã cho smart contract, xác định các chức năng và điều kiện thực thi.
- **Biên dịch**: Sau khi viết xong, mã cần được biên dịch thành bytecode để EVM có thể thực thi.
- **Thử nghiệm và gỡ lỗi**: Trước khi triển khai smart contract trên mạng lưới Ethereum, nên tiến hành kiểm thử và gỡ lỗi trong môi trường giả lập để đảm bảo rằng contract hoạt động như mong đợi và không chứa lỗi.
- **Chọn nền tảng triển khai**: Có nhiều nền tảng hỗ trợ triển khai smart contracts như Remix – IDE, dành riêng cho phát triển smart contracts trên Ethereum, cho phép viết, kiểm tra, triển khai và gỡ lỗi smart contracts trực tiếp.
- **Kết nối với mạng Ethereum**: Để triển khai, nền tảng cần kết nối với mạng Ethereum.

- **Triển khai Smart Contract**: Được thực hiện bằng cách gửi một giao dịch chứa bytecode của hợp đồng đến một địa chỉ trên mạng Ethereum.

2.3.3 Ứng dụng của Smart Contracts

Smart contracts trên Ethereum mở ra một loạt các ứng dụng tiềm năng và phổ biến, thay đổi cách thức hoạt động của nhiều lĩnh vực như:

- **NFTs**: Smart contracts cho phép tạo ra và quản lý NFTs, đại diện cho quyền sở hữu đối với các tài sản kỹ thuật số như nghệ thuật, âm nhạc. Chúng giúp đảm bảo tính duy nhất và chứng minh quyền sở hữu của NFTs.
- **Hệ thống bình chọn phi tập trung**: Smart Contracts có thể được sử dụng để tạo ra các hệ thống bình chọn minh bạch và không thể thao túng, tăng cường tính công bằng và chính xác trong các quy trình bỏ phiếu.

3 Cài đặt

Chương này mô tả smart contract nhóm cài đặt, cách triển khai cũng như kết quả test của smart contract.

3.1 Lựa chọn vấn đề

Đề tài nhóm lựa chọn là **Decentralized Voting System**. Mục đích của smart contract này là cung cấp một cách minh bạch và an toàn để tiến hành các cuộc bầu cử hoặc bỏ phiếu. Các tính năng của smart contract này gồm:

- Tạo phiên bỏ phiếu: Cho phép người dùng tạo ra một phiên bỏ phiếu mới với thời gian bắt đầu và kết thúc cụ thể.
- Thêm ứng cử viên: Người dùng có thể thêm danh sách các ứng cử viên vào phiên bỏ phiếu.
- Đăng ký cử tri: Cho phép người dùng (địa chỉ Ethereum) đăng ký làm cử tri trong một phiên bỏ phiếu cụ thể.
- Bỏ phiếu: Cho phép các cử tri đã đăng ký bỏ phiếu cho ứng cử viên của họ trong phiên bỏ phiếu. Việc bỏ phiếu chỉ hợp lệ nếu cử tri thoả các điều kiện sau: đã đăng ký làm cử tri, bỏ phiểu khi phiên bỏ phiếu đang diễn ra và chưa bỏ phiếu lần nào.
- Xem số phiếu của ứng cử viên: Cho phép truy xuất số phiếu mà một ứng cử viên cụ thể đã nhận được sau khi phiên bỏ phiếu kết thúc. Trong khi phiên bầu vẫn đang diễn ra, không thể xem được số phiếu của mỗi ứng cử viên.

3.2 Chi tiết cài đặt

Các cấu trúc dữ liệu được dùng trong smart contract

Candidate: Cấu trúc dữ liệu này lưu trữ thông tin về một ứng cử viên, bao gồm:

- id: ID duy nhất của ứng cử viên.
- name: Tên của ứng cử viên.
- voteCount: Số phiếu bầu mà ứng cử viên đã nhận được.

VotingSession: Đây là cấu trúc dữ liệu chính, lưu trữ thông tin về một phiên bỏ phiếu cụ thể. Nó bao gồm:

- startTime và endTime: Thời gian bắt đầu và kết thúc của phiên bỏ phiếu.
- candidates: Một mapping từ ID của ứng cử viên đến cấu trúc dữ liệu Candidate.
- candidatesCount: Số lượng ứng cử viên trong phiên bỏ phiếu.
- voters: Mapping để theo dõi xem một địa chỉ cụ thể đã bỏ phiếu chưa.
- registeredVoters: Mapping để theo dõi những người đã đăng ký làm cử tri.

votingSessions: Một mapping từ ID phiên bỏ phiếu đến cấu trúc VotingSession.

votingSessionsCount: Biến để theo dõi tổng số phiên bỏ phiếu đã được tạo.

Tiến trình làm việc của một phiên bỏ phiếu như sau

Khởi tạo phiên bỏ phiếu

- Gọi hàm createVotingSession, cung cấp thời gian bắt đầu (_startTime), thời gian kết thúc (_endTime), và một mảng tên các ứng cử viên (candidateNames).
- Contract kiểm tra để đảm bảo thời gian kết thúc sau thời gian bắt đầu.
- Mỗi tên trong candidateNames được thêm vào phiên bỏ phiếu mới thông qua hàm addCandidateToSession.

Đăng ký cử tri

- Gọi hàm registerVoter, cung cấp ID phiên bỏ phiếu (sessionID).
- Contract kiểm tra xem người dùng đã đăng ký trước đó hay chưa.
- Đăng ký người dùng là cử tri cho phiên bỏ phiếu đó.

Quá trình bỏ phiếu

- Gọi hàm vote, cung cấp ID phiên bỏ phiếu (sessionID) và ID của ứng cử viên (_candidateId).
- Contract thực hiện các kiểm tra:
    - Xác nhận thời gian bỏ phiếu đang diễn ra.

- Kiểm tra xem người dùng có phải là cử tri đã đăng ký không.
- Đảm bảo người dùng chưa bỏ phiếu trước đó.
- Xác nhận ID ứng cử viên hợp lệ.

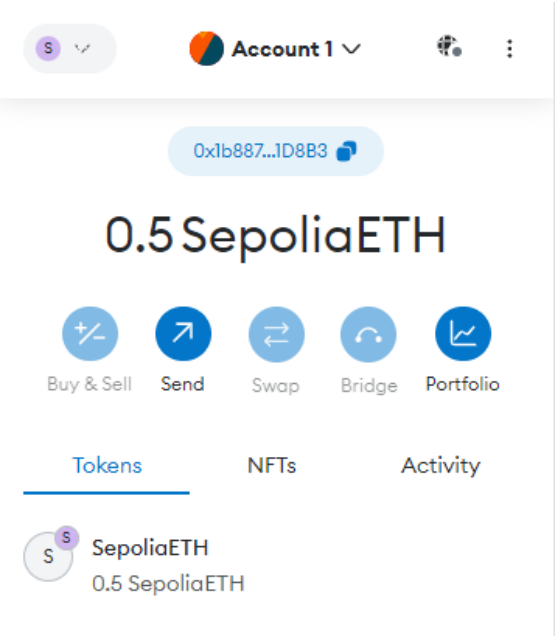
- Tăng số phiếu cho ứng cử viên được chọn.
- Ghi nhận người dùng đã bỏ phiếu.

Truy xuất kết quả bỏ phiếu

- Gọi hàm getCandidateVoteCount, cung cấp ID phiên bỏ phiếu (sessionID) và ID ứng cử viên (_candidateId).
- Contract kiểm tra xem phiên bỏ phiếu đã kết thúc và ID ứng cử viên có hợp lệ.
- Trả về số phiếu mà ứng cử viên đã nhận được.

3.3 Test

Tạo một ví MetaMask và chọn ETH testnet (Sepolia testnet). Sau đó truy cập Sepolia faucet để nhận token.

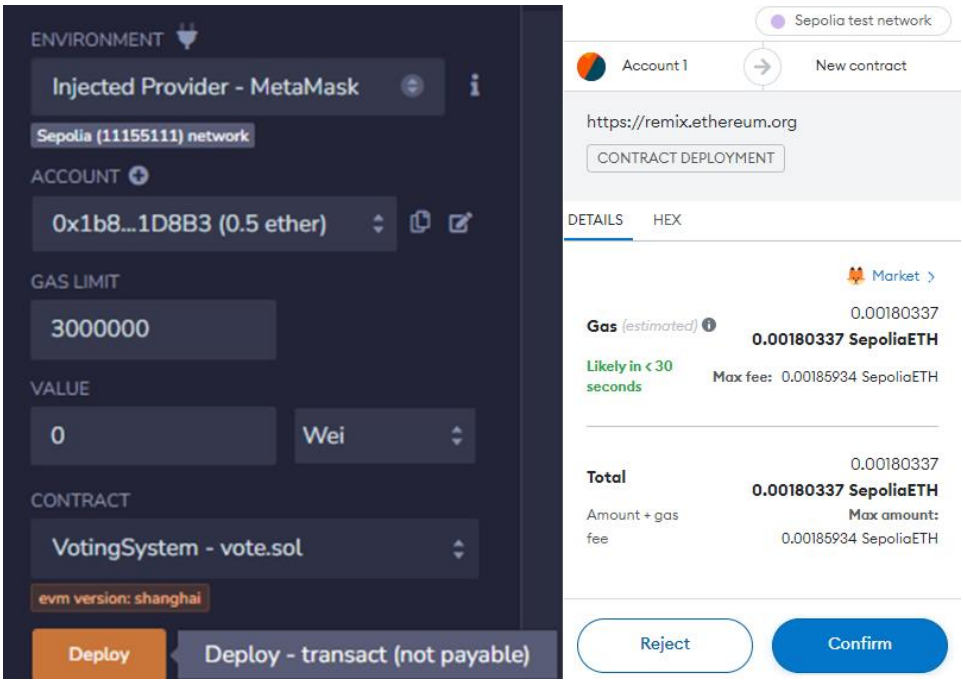


Sau khi nhận token thành công. Tiến hành triển khai smart contract. Nhóm sử dụng Remix để triển khai và test smart contract. Để triển khai smart contract, thực hiện các bước sau.

1. Truy cập vào website https://remix.ethereum.org/ và upload file chưa smart contract ở thẻ File Explorer.

2. Chuyển qua thẻ Solidity compiler để compile smart contract

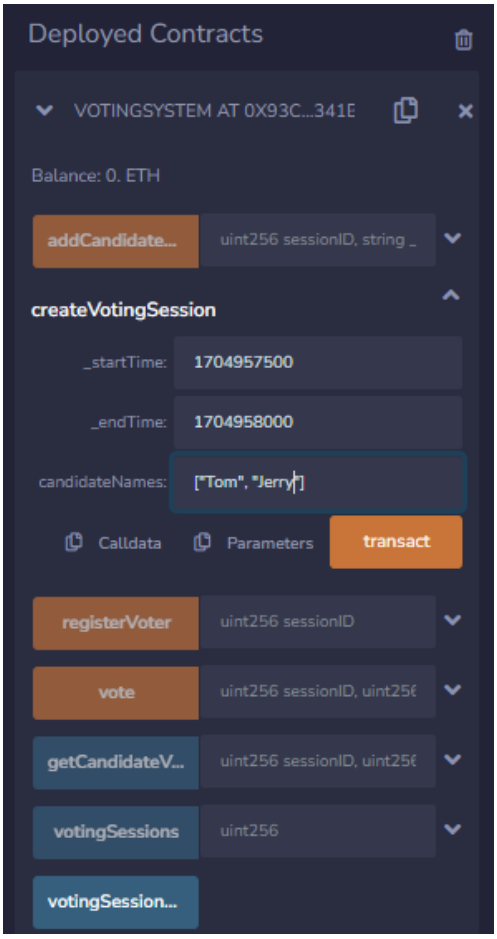
3. Chuyển qua thẻ Deploy & Run transactions, chọn Environment là “Injected Provider – MetaMask” để kết nối với ví MetaMask. Sau đó chọn Account là ví đã nhận token và deploy. Ví MetaMask sẽ yêu cầu xác nhận, khi deploy smart contract sẽ tốn phí. Chọn xác nhận



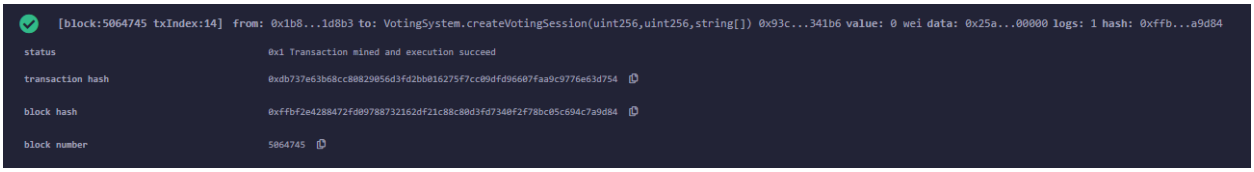
Sau khi triển khai smart contract thành công, contract vừa được triển khai sẽ xuất hiện trong mục Deployed Contracts. Tiếp theo, nhóm test xem smart contract đã hoạt động đúng theo thiết kế hay chưa.

Tạo một Vote session với input như sau:

- Thời gian bắt đầu: 1704957500
- Thời gian kết thúc: 1704958000
- Ứng viên: ["Tom", "Jerry"]



Khi tạo vote session thành công, sẽ nhận được thông báo như sau



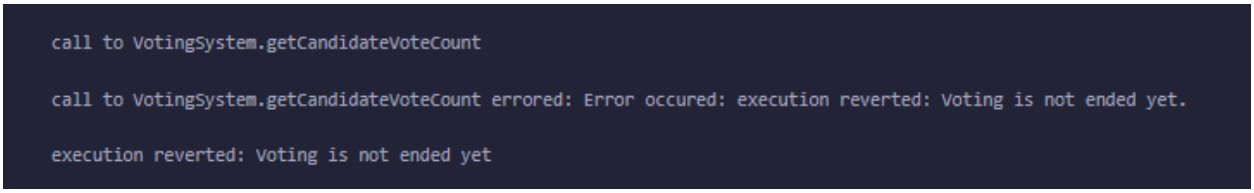
Tiến hành vote cho cử tri có ID 2, tuy nhiên vì account này chưa đăng ký làm cử tri nên xảy ra lỗi.

Đăng ký làm cử tri

Vote lại lần nữa cho ứng viên có ID 2 và thành công

Vote thêm một lần nữa để kiểm tra xem smart contract có thể ngăn chặn một ví vote nhiều lần hay không. Lần vote này không thành công, như vậy smart contract đã thành công ngăn chặn.

Tiến hành xem số phiếu đã được vote của ứng viên, tuy nhiên Vote session chưa kết thúc nên không thể xem được số phiếu.



Sau khi Vote session đã kết thúc, xem số phiếu lần nữa và thành công.

4 Kết luận

Trong quá trình nghiên cứu và phát triển đồ án, nhóm đã có cái nhìn khái quát về hệ sinh thái Ethereum, ngôn ngữ lập trình Solidity và cách thức hoạt động của smart contracts. Khả năng tương tác linh hoạt giữa Smart Contracts và Ethereum Blockchain mở ra không gian rộng lớn cho sự sáng tạo và triển khai các ứng dụng phi tập trung.

Trong quá trình thực hiện đồ án về phát triển Simple Smart Contract, nhóm đã đối mặt với những thách thức đáng kể:

- Solidity là một ngôn ngữ lập trình mới và không được sử dụng trong các mãng các của khoa học máy tính. Điều này tạo ra một thách thức đáng kể cho những người mới, đặc biệt là trong việc hiểu các khái niệm như smart contracts, transactions và gas fees.
- Nhóm cũng chưa có kinh nghiệm trực tiếp trong việc viết và test smart contracts vì vậy gặp phải khó khăn trong việc đảm bảo rằng các contracts được viết một cách chính xác, hiệu quả và an toàn.
- Tuy nhiên nhờ vào nhiều tài liệu và các ví dụ mẫu được chia sẽ rộng rãi, đặc biệt là tài liệu của Solidity, nhóm đã có thể triển khai smart contract theo thiết kế của mình

Bảng đánh giá các tính năng:

Tính năng	Tự đánh giá
Tạo phiên bỏ phiếu: Cho phép người dùng tạo ra một phiên bỏ phiếu mới với thời gian bắt đầu và kết thúc cụ thể.	Đã hoàn thành
Thêm ứng cử viên: Người dùng có thể thêm danh sách các ứng cử viên vào phiên bỏ phiếu.	Đã hoàn thành
Đăng ký cử tri: Cho phép người dùng (địa chỉ Ethereum) đăng ký làm cử tri trong một phiên bỏ phiếu cụ thể.	Đã hoàn thành
Bỏ phiếu: Cho phép các cử tri đã đăng ký bỏ phiếu cho ứng cử viên của họ trong phiên bỏ phiếu. Việc bỏ phiếu chỉ hợp lệ nếu cử tri thoả các điều kiện sau: đã đăng ký làm cử tri, bỏ phiếu khi phiên bỏ phiếu đang diễn ra và chưa bỏ phiếu lần nào.	Đã hoàn thành
Xem số phiếu của ứng cử viên: Cho phép truy xuất số phiếu mà một ứng cử viên cụ thể đã nhận được sau khi phiên bỏ phiếu kết thúc. Trong khi phiên bầu vẫn đang diễn ra, không thể xem được số phiếu của mỗi ứng cử viên.	Đã hoàn thành

Những đề xuất cải tiến smart contract trong tương lai:

- Tính ẩn danh khi bỏ phiếu: Cải tiến smart contract để bảo vệ danh tính của cử tri. Sử dụng các cơ chế mã hóa đặc biệt để cử tri có thể bỏ phiếu mà không lộ danh tính.
- Phát triển cơ chế phát hiện và phòng chống gian lận: Tích các thuật toán phát hiện hành vi bất thường hoặc gian lận trong quá trình bầu cử, ví dụ như 1 người tạo nhiều ví để bầu.
- Hỗ trợ bình chọn nhiều vòng: Cho phép tổ chức các cuộc bầu cử với nhiều vòng, ví dụ như vòng sơ bộ và vòng chung kết.
- Tạo giao diện người dùng: Phát triển ứng dụng web hoặc mobile giúp người dùng dễ dàng tương tác với hợp đồng thông minh khi bầu cử.
- Hỗ trợ Cross-chain: Cho phép hợp đồng thông minh tương tác với các Blockchain khác ngoài Ethereum, tăng cường tính linh hoạt và mở rộng khả năng tiếp cận.

Bài lab này không chỉ cung cấp cho nhóm kiến thức cơ bản về Ethereum và Solidity, mà còn giúp nhóm phát triển kỹ năng thực hành trên Blockchain. Với sự phát triển không ngừng của Solidity và công nghệ blockchain, những kiến thức và kỹ năng mà nhóm đã học được sẽ là nền tảng vững chắc cho sự nghiệp của mình trong lĩnh vực này.

Tài liệu tham khảo

[1] (2023, September 4th). The History of Ethereum: Its Origin and Upgrades. https://worldcoin.org/articles/history-of-ethereum

[2] (2023, August 15th). Intro to Ethereum. https://ethereum.org/en/developers/docs/intro-to-ethereum

[3] (2024). Solidity Documentation - Solidity Authors. https://docs.soliditylang.org/

[4] (2023, August 15th). Introduction to Smart Contracts. https://ethereum.org/en/developers/docs/smart-contracts