

ĐẠI HỌC HUẾ
TRƯỜNG ĐẠI HỌC KINH TẾ



HỌC PHẦN: CÔNG NGHỆ CHUỖI KHỐI VÀ ỨNG DỤNG
BÁO CÁO TỔNG KẾT HỌC PHẦN

GIẢNG VIÊN HƯỚNG DẪN : TS. NGUYỄN MINH ĐỨC
SINH VIÊN THỰC HIỆN : PHẠM TUẤN LINH
MÃ SINH VIÊN : 21A5020701

Thừa Thiên Huế - Năm 2025

DANH MỤC TỪ VIẾT TẮT

STT	Chữ viết đầy đủ	Chữ viết tắt
1	Credit Score Center, hay Trung tâm Thông tin Tín Dụng	CIC
2	Decentralized application, hay Ứng dụng phi tập trung	DApp
3	One Time Password, hay Mật khẩu sử dụng một lần	OTP
4	Ethereum Virtual Machine, hay Máy ảo Ethereum	EVM
5	User Interface / User Experience, hay Giao diện người dùng / Trải nghiệm người dùng	UI/UX

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN VỀ BLOCKCHAIN, HỢP ĐỒNG THÔNG

MINH VÀ HỆ THỐNG CIC	3
1.1. Khái niệm chung về blockchain và hợp đồng thông minh	3
1.1.1. Tổng quan về blockchain.....	3
1.1.2. Tổng quan về hợp đồng thông minh.....	4
1.2. Tổng quan về ứng dụng của blockchain	5
1.2.1. Khái niệm hệ thống CIC	5
1.2.2. Tầm quan trọng của blockchain trong lĩnh vực ngân hàng.....	6

CHƯƠNG 2: ỨNG DỤNG BLOCKCHAIN TRIỂN KHAI HỢP ĐỒNG

QUẢN LÝ ĐIỂM TÍN DỤNG CIC	8
2.1. Đặc điểm kỹ thuật của ứng dụng	8
2.1.1. Yêu cầu hệ thống và mô hình kiến trúc	8
2.1.2. Các công nghệ và công cụ được sử dụng	11
2.1.3. Mô tả chi tiết về hợp đồng thông minh.....	11
2.2. Thiết kế và triển khai	12
2.2.1. Quy trình thiết kế hợp đồng thông minh.....	12
2.2.2. Triển khai hợp đồng thông minh trên testnet.....	15
2.2.3. Vấn đề kỹ thuật đã gặp phải	20

CHƯƠNG 3: ĐỊNH HƯỚNG VÀ GIẢI PHÁP HOÀN THIỆN HỢP ĐỒNG

THÔNG MINH QUẢN LÝ ĐIỂM TÍN DỤNG CIC.....	21
3.1. Kết luận và hướng phát triển	21
3.1.1. Tổng kết các kết quả đạt được	21
3.1.2. Đề xuất hướng phát triển trong tương lai	21
3.2. Tài liệu tham khảo và phụ lục.....	22
3.2.1. Danh sách các nguồn tài liệu tham khảo đã sử dụng	22
3.2.2. Phụ lục: Toàn bộ code của hợp đồng thông minh	23

CHƯƠNG 1: TỔNG QUAN VỀ BLOCKCHAIN, HỢP ĐỒNG THÔNG MINH VÀ HỆ THỐNG CIC

1.1. Khái niệm chung về blockchain và hợp đồng thông minh

1.1.1. Tổng quan về blockchain

Blockchain (hay block chain - chuỗi khối) là một hệ thống lưu trữ và truy cập dữ liệu thông tin được nối mạng phi tập trung và đa cơ quan. Có thể gọi blockchain là một sổ cái công khai, phi tập trung, bất biến. Tuy không ai là chủ sở hữu duy nhất của hệ thống dữ liệu này, nhưng tất cả mọi máy hoạt động trong hệ thống đều sở hữu một bản sao của dữ liệu. Việc này cho thấy Blockchain không có cơ sở dữ liệu trung tâm và tại một thời điểm thì mỗi người trong mạng lưới đều giữ chính xác cùng một bản sao. Blockchain chỉ có thể nối thêm, dữ liệu sau khi thêm vào không thể bị xóa hoặc sửa đổi sau này. Do đó, khi một khối được thêm vào blockchain thì dữ liệu tại khối đó không thể bị phủ nhận sau này. Blockchain có những chức năng chính như sau: Thứ nhất, Blockchain hoạt động như một cấu trúc dữ liệu, tức chuỗi các khối dữ liệu. Mỗi một khối này đều chứa thông tin giao dịch và được liên kết với khối trước bằng mã hash, tạo thành một chuỗi không thể thay đổi.

Thứ hai, Blockchain là giao thức mạng: Các giao thức blockchain cho phép các nút (nodes) trong mạng giao tiếp với nhau, xác thực và đồng bộ hóa thông tin mà không cần một bên trung gian. Điều này giúp duy trì tính toàn vẹn của dữ liệu qua mạng lưới phi tập trung.

Thứ ba, Blockchain đóng vai trò là một hộp đen bảo mật, nơi lưu trữ tất cả giao dịch mà không cho phép sửa đổi sau khi đã ghi lại. Tính năng này đảm bảo tính minh bạch và khả năng kiểm tra khi cần.

Thứ tư, Blockchain là một hệ thống phân tán, trong đó không có một điểm duy nhất để kiểm soát hay quản lý. Việc này giúp tăng cường tính bền vững trong hoạt động an ninh mạng và chống lại các cuộc tấn công. Mỗi nút trong mạng giữ một bản sao của sổ cái, làm giảm nguy cơ mất mát dữ liệu.

Cuối cùng, Blockchain đóng vai trò như một khung lập trình. Blockchain cung cấp cấu trúc để xây dựng các ứng dụng phi tập trung thông qua các hợp đồng thông minh. Điều này cho phép các lập trình viên phát triển các ứng dụng mà không cần trung gian.

Những yếu tố này giúp blockchain cung cấp một giải pháp an toàn, đáng tin cậy và hiệu quả cho việc giao tiếp và lưu trữ thông tin, đặc biệt là trong lĩnh vực ngân hàng.

1.1.2. Tổng quan về hợp đồng thông minh

Hợp đồng thông minh là một chương trình tự thực hiện tự động hóa các hành động cần thiết trong giao dịch blockchain. Sau khi hoàn tất, các giao dịch có thể theo dõi và không thể đảo ngược. Hợp đồng thông minh cho phép thực hiện các giao dịch và thỏa thuận đáng tin cậy giữa các bên khác nhau mà không cần đến bên trung gian.

Hợp đồng thông minh được xác lập dựa trên nền tảng của công nghệ blockchain mà bản chất blockchain là một chuỗi khối liên kết với nhau bằng các hàm băm với tính bảo mật cao nên hợp đồng thông minh theo đó cũng có độ tin cậy vượt trội, bảo vệ các chủ thể tham gia xác lập khỏi những sự cố không mong muốn như giả mạo, lừa đảo, xâm nhập bất hợp pháp. Một khi xác lập hợp đồng thông minh, đồng nghĩa với việc các bên tự nguyện đặt niềm tin vào độ bảo mật của blockchain thay vì bất kỳ bên trung gian nào. Nó không bị bất kỳ một tổ chức thứ ba nào nắm quyền kiểm soát vì thế tránh được nhiều rủi ro có thể xảy ra khi bên thứ ba tham gia vào hợp đồng.

Các hợp đồng thông minh được lưu trữ trên nền tảng blockchain và có sự dán nhãn về thời gian cho tất cả các hành động được thực thi. Một khi các dữ liệu được nhập trực tiếp vào các khối của blockchain thì nó không thể bị thay đổi hoặc sửa chữa bởi bất kỳ ai do sự hoạt động của thuật toán đồng thuận và mã hash liên kết các khối chứa thông tin lại với nhau.

Nếu muốn thay đổi thông tin nhằm gây bất lợi cho bất kỳ bên nào sẽ phải đối mặt với một hệ thống bảo mật cực kỳ cao và quá trình sửa đổi toàn bộ các mã hash liên kết các khối gần như là bất khả thi. Đồng thời bởi tính phi tập trung nên các thông tin không thể bị thất lạc do đều được mã hóa trên một cuốn sổ cái chung, đáp ứng được sự minh bạch và an toàn. Nhờ sự ghi nhớ, bảo mật cao mà ai cũng có thể dễ dàng truy dấu được nguồn gốc của tất cả các giao dịch, không có khả năng đảo ngược giao dịch và mọi dấu vết đều được ghi nhận một cách rõ ràng, cụ thể trên blockchain. Điều này vô hình chung tạo được sự trung thực của các bên khi tham gia vào bất kỳ một hợp đồng thông minh nào.

1.2. Tổng quan về ứng dụng của blockchain

1.2.1. Khái niệm hệ thống CIC

CIC (Credit Score Center) là tên viết tắt Tiếng Anh của Trung tâm thông tin tín dụng quốc gia Việt Nam, nay trực thuộc ngân hàng nhà nước Việt Nam. Trung tâm này có chức năng chính là thu thập, lưu trữ, phân tích dự báo thông tin tín dụng cho các cá nhân, tổ chức, doanh nghiệp tại Việt Nam. Những thông tin báo cáo do CIC cung cấp giúp giảm thiểu tỷ lệ rủi ro trong quá trình cho vay và quản lý tín dụng ở các ngân hàng và tổ chức tín dụng.

Điểm tín dụng CIC là chỉ số về độ tín nhiệm, khả năng trả nợ của khách hàng là người vay. Được đánh giá thông qua việc xem lịch sử vay vốn tại ngân hàng, tổ chức tài chính. Điểm tín dụng được ghi nhận tại trung tâm thông tin dụng quốc gia. Điểm tín dụng càng cao thì khả năng tiếp cận khoản vay cao. Ngược lại nếu điểm tín dụng thấp thì người vay khó có khả năng tiếp cận khoản vay.

CIC thu thập những thông tin sau đây:

- Số tiền đã vay, đang vay, đã từng vay
- Mục đích sử dụng khoản tiền khi vay
- Hợp đồng tín dụng ký kết với những ngân hàng nào
- Thời gian trả nợ, lịch sử thanh toán khoản vay
- Tình trạng hiện tại của những khoản nợ
- Tài sản thế chấp ngân hàng khi vay.

Căn cứ vào những thông tin tài liệu thu thập, CIC tiến hành phân loại nợ xấu thành từng nhóm để giúp ngân hàng và tổ chức tín dụng có thể tra cứu lịch sử tín dụng của mỗi cá nhân, tổ chức, doanh nghiệp. Theo quy định pháp luật tại Điều 10 Thông tư 11/2021/TT-NHNN, việc phân loại nợ dành cho Tổ chức tín dụng và chi nhánh ngân hàng nước ngoài theo phương pháp định lượng thành 05 nhóm như sau:

- a) Nhóm 1 (Nợ đủ tiêu chuẩn):
- b) Nhóm 2 (Nợ cần chú ý):
- c) Nhóm 3 (Nợ dưới mức tiêu chuẩn):
- d) Nhóm 4 (Nợ nghi ngờ)
- đ) Nhóm 5 (Nợ có nguy cơ mất vốn):

1.2.2. Tầm quan trọng của blockchain trong lĩnh vực ngân hàng

Hệ thống thông tin tín dụng điện tử được CIC đưa vào sử dụng từ cuối năm 2001, nhanh chóng mang lại hiệu quả, tạo khả năng phát triển mạnh mẽ. Đến nay, mở rộng hệ thống tới tất cả các chi nhánh TCTD trên cả nước; hàng triệu thông tin được cập nhật hàng ngày; hơn 99% giao dịch thu thập, khai thác thông tin tự động. Vì đã hoạt động được 23 năm, hệ thống CIC đến nay đã bộc lộ nhiều hạn chế, thiếu sót. Tiêu biểu trong đó là hệ thống này do Trung tâm thông tin tín dụng quốc gia Việt Nam thuộc Ngân hàng Trung ương quản lý nhưng quyền cập nhật và tra cứu hồ sơ hệ thống CIC lại được cấp phát cho cán bộ của các ngân hàng thương mại. Do đó, nhiều vụ việc cán bộ ngân hàng làm lộ thông tin tín dụng CIC của các cá nhân và doanh nghiệp đã gây xôn xao dư luận trong thời gian qua. Gần đây nhất có thể kể tới vụ 2 cá nhân khởi kiện cán bộ ngân hàng Nam Á Bank ngày 14/7/2024 vì cho rằng người này đã trích xuất thông tin CIC của họ và cung cấp, tiết lộ cho người không liên quan. Việc mua bán thông tin khách hàng để phục vụ mục đích cá nhân vẫn đang là vấn đề nhức nhối trong ngành tài chính nói chung và ngân hàng nói riêng. Đây là hành vi vi phạm quy định về bảo vệ thông tin cá nhân và quy định về hoạt động tín dụng. Nếu các thông tin như lịch sử tín dụng hay mục đích của khoản vay rơi vào tay kẻ xấu có thể gây ra hậu quả khôn lường. Đơn giản nhất có thể kể đến việc bị làm phiền bởi telesales cho ngân hàng hoặc công ty tài chính nhằm tiếp thị các khoản vay, dịch vụ tài chính khác.

Việc sử dụng cơ sở dữ liệu để quản lý điểm tín dụng đã để lộ ra nhiều nhược điểm như có quá nhiều bên tham gia (tức các ngân hàng thương mại và công ty tài chính thực hiện hoạt động tín dụng) và sự minh bạch, chính xác của thông tin chỉ tập trung vào một bên (cán bộ một ngân hàng gửi yêu cầu lên CIC thì hồ sơ tín dụng của cá nhân hoặc tổ chức sẽ được cập nhật mà việc xác thực phụ thuộc duy nhất vào CIC). Có thể thấy, trong trường hợp này ta nên ưu tiên sử dụng blockchain để lưu trữ dữ liệu thay vì sử dụng cơ sở dữ liệu truyền thống. Việc triển khai hợp đồng thông minh để quản lý điểm tín dụng có những ưu điểm sau: tính minh bạch cao trong lưu trữ và xử lý thông tin, có nhiều bên tham gia và cần loại bỏ trung gian (CIC), đảm bảo tính bất biến, dữ liệu không thể bị sửa đổi hoặc gian lận và có thể truy cập phân tán, đồng bộ thông tin trên nhiều hệ thống khác nhau.

Hạn chế của hợp đồng thông minh triển khai trong lĩnh vực này có thể kể tới là chi phí lưu trữ và tính toán hàm trong hợp đồng khá cao, đặc biệt khi khối lượng dữ liệu là rất lớn. Tuy nhiên, việc này đồng thời đáp ứng được chủ trương tinh gọn bộ máy nhà nước. Trung tâm tín dụng quốc gia Việt Nam đã hoạt động trong thời gian dài nhưng không khắc phục được những hạn chế, bất cập còn tồn đọng trong hoạt động quản lý tín dụng. Hơn nữa, các vụ kiện cáo như trên kéo dài mà không được giải quyết thỏa đáng đã làm tổn kém rất nhiều tiền bạc và thời gian của các bên liên quan. Do đó, việc giải thể trung tâm này và chuyển ngân sách nhà nước sang xây dựng hệ thống quản lý điểm tín dụng trên blockchain là hoàn toàn khả thi.

Hợp đồng thông minh để quản lý điểm tín dụng là một chương trình tự động được triển khai trên blockchain, dùng để quản lý và đánh giá điểm tín dụng của cá nhân hoặc tổ chức. Nó thay thế các phương pháp truyền thống trong việc phân loại rủi ro tín dụng bằng cách lưu trữ, xử lý thông tin và áp dụng các thuật toán đánh giá dựa trên các tiêu chí cụ thể như lịch sử thanh toán, dư nợ, độ dài tín dụng và các thông tin khác.

Từ những lý do trên, tác giả sẽ xây dựng hợp đồng lưu trữ điểm tín dụng tại Chương 2 đáp ứng những yêu cầu sau:

i, Bảo mật dữ liệu: blockchain sử dụng mã hóa để bảo vệ dữ liệu người dùng. Hợp đồng thông minh quản lý điểm tín dụng cần thiết kế để thực hiện các biện pháp kiểm soát truy cập nghiêm ngặt để chỉ những người có thẩm quyền mới có thể truy cập thông tin tín dụng.

ii, Chính xác: hệ thống lưu trữ điểm tín dụng cần cập nhật điểm tín dụng thường xuyên dựa trên thông tin từ các tổ chức tài chính và ngân hàng. Việc sử dụng blockchain để xác thực dữ liệu từ nhiều nguồn khác nhau sẽ đảm bảo tính chính xác và đáng tin cậy.

iii, Minh bạch: Cung cấp thông tin chi tiết về cách tính toán điểm tín dụng, bao gồm các yếu tố ảnh hưởng đến điểm số. Cho phép người dùng xem lịch sử thay đổi điểm tín dụng của họ.

CHƯƠNG 2: ỨNG DỤNG BLOCKCHAIN TRIỂN KHAI HỢP ĐỒNG QUẢN LÝ ĐIỂM TÍN DỤNG CIC

2.1. Đặc điểm kỹ thuật của ứng dụng

2.1.1. Yêu cầu hệ thống và mô hình kiến trúc

2.1.1.1. Yêu cầu hệ thống

Để triển khai hợp đồng quản lý điểm tín dụng trên blockchain, về yêu cầu hệ thống, phần On-chain cần có:

- Mạng blockchain: Ethereum hoặc bất kỳ blockchain nào hỗ trợ EVM (Ethereum Virtual Machine) và Solidity. Ethereum sử dụng cơ chế đấu giá để quyết định thứ tự xử lý giao dịch. Khi mạng chậm hoặc tắc nghẽn, các miner/validator ưu tiên giao dịch có phí cao hơn, giá gas sẽ tăng để đảm bảo giao dịch triển khai hợp đồng không bị trì hoãn hoặc thất bại. Vì vậy, hệ thống này cần băng thông mạng tốt để tối đa phí giao dịch. Có thể triển khai trên mạng riêng (private blockchain) nếu cần bảo mật cao.

- Hợp đồng thông minh: Triển khai mã Solidity đã tối ưu hóa, bao gồm logic xử lý các chức năng tính cập nhật và tính điểm tín dụng, quản lý OTP (One time password) và truy xuất dữ liệu.

Về phần Off-chain, tức những tác vụ được xử lý bên ngoài blockchain và là thành phần hỗ trợ việc tương tác giữa người dùng cuối và blockchain, yêu cầu hệ thống bao gồm:

- Giao diện người dùng (UI/UX): Giao diện web hoặc ứng dụng di động cho CIC admin và người dùng. Cần đáp ứng những chức năng trong hợp đồng như trên: nhập thông tin cá nhân, xác thực OTP và kiểm tra điểm tín dụng.

- Máy chủ (Back-end server): Hỗ trợ việc kết nối với blockchain thông qua API (như Web3.js hoặc ethers.js) và có thể xử lý các yêu cầu xác thực và logic bổ sung

- Cơ sở dữ liệu (như MySQL, MongoDB): để lưu trữ thông tin không nhạy cảm như lịch sử truy vấn, nhật ký hoạt động và đồng bộ hóa với blockchain để giảm tải truy cập trực tiếp - tức cập nhật cơ sở dữ liệu theo thông tin blockchain.

Để kết hợp cả tất cả yêu cầu trên, có thể thiết kế và đưa vào sử dụng DApps (Decentralized Applications - ứng dụng phi tập trung, sử dụng hợp đồng thông minh và chạy trên mạng blockchain). DApps có thể kết hợp giữa các thành phần hoạt động

trên blockchain và hoạt động ngoài blockchain. Mỗi thành phần này trong DApps sẽ có những tác vụ cơ bản sau:

- On-chain: Tính toán điểm tín dụng, phân loại CIC và lưu trữ thông tin quan trọng.

- Off-chain: Xử lý nhập liệu từ UI/UX, đồng bộ và lưu trữ dữ liệu, tích hợp dịch vụ SMS gateway để lấy OTP và API blockchain vào máy chủ.

2.1.1.2. Mô hình kiến trúc

Layer 1: DApps

Công nghệ sử dụng đề xuất: Web3.js hoặc ethers.js.

Chức năng: Giao diện phi tập trung (web hoặc mobile) giúp CIC admin và người dùng nhập thông tin tín dụng, tra cứu điểm tín dụng, và xác thực OTP.

Kết quả: Trực tiếp kết nối với hợp đồng thông minh trên blockchain.

Layer 2: Middleware API (Back-end)

Công nghệ sử dụng đề xuất: Node.js.

Chức năng: Nhận dữ liệu từ front-end để xử lý yêu cầu và xác thực cơ bản, kết nối với blockchain qua API RPC hoặc thư viện Web3.js/ether.js, Ghi nhật ký hoạt động và đồng bộ dữ liệu không nhảy cảm vào cơ sở dữ liệu off-chain để tăng tốc độ xử lý.

Kết quả: Dữ liệu được gửi đến blockchain hoặc trả về kết quả cho người dùng.

Layer 3: Blockchain

Công nghệ sử dụng: Ethereum (hoặc Sepolia testnet trong trường hợp này).

Chức năng: Lưu trữ Điểm tín dụng, OTP, và trạng thái hợp lệ. Tính toán, xử lý các yêu cầu như tính điểm tín dụng và phân loại CIC. Ghi và lưu lại các thay đổi dữ liệu để đảm bảo tính minh bạch và bất biến.

Kết quả: Đảm bảo thông tin an toàn, không bị giả mạo, và có thể truy vết.

Mô hình này không chỉ tối ưu hóa hiệu suất mà còn đồng thời đảm bảo tính bảo mật và minh bạch của dữ liệu tín dụng thông qua sự kết hợp hài hòa giữa các lớp on-chain và off-chain. Nếu sử dụng cả 3 Layer như vậy, Layer 2 đóng vai trò là cầu nối giữa DApps và blockchain, cung cấp khả năng xử lý logic phức tạp và quản lý dữ liệu off-chain.

Ưu điểm:

- Middleware API xử lý các yêu cầu nhẹ hoặc logic phức tạp, giảm tải cho blockchain, giúp cải thiện hiệu suất.
- Phù hợp với hệ thống lớn hoặc có nhiều người dùng đồng thời như hệ thống tra cứu điểm tín dụng.
- Dễ dàng bổ sung tính năng như thống kê, lịch sử truy vấn, mà không phải ghi tất cả vào blockchain.
- Dữ liệu không cần bảo mật cao có thể lưu trữ trong cơ sở dữ liệu off-chain, tránh phí gas blockchain.

Nhược điểm:

- Sử dụng 3 Layer sẽ làm mô hình phức tạp, kiến trúc cần thiết kế kỹ lưỡng để đảm bảo đồng bộ và bảo mật.
 - Chi phí vận hành cao hơn, do đòi hỏi tài nguyên máy chủ để duy trì Layer 2.
- Ngoài ra, phụ thuộc vào yêu cầu của hệ thống, có thể chỉ dùng Layer 1 và 3. Khi đó, DApps giao tiếp trực tiếp với blockchain mà không cần trung gian.

Ưu điểm của mô hình dùng 2 Layer là có tính minh bạch cao, do mọi giao dịch và truy vấn đều thực hiện trực tiếp trên blockchain. Kiến trúc mô hình được đơn giản hóa: Loại bỏ lớp trung gian, giảm chi phí và rủi ro bảo mật.

Nhược điểm của mô hình này là hiệu suất thấp do Blockchain xử lý tất cả các yêu cầu, dẫn đến thời gian phản hồi chậm khi tải cao. Với lượng lớn người dùng, việc ghi dữ liệu hoặc thực hiện giao dịch trực tiếp trên blockchain trở nên đắt đỏ.

Tuy nhiên, tác giả khuyến nghị nên sử dụng kiến trúc mô hình có 3 Layer, do đặc điểm của hệ thống lưu trữ điểm tín dụng: Hệ thống có số lượng lớn người dùng hoặc yêu cầu truy cập đồng thời rất cao, vì CIC cung cấp dịch vụ cho cả một quốc gia. Ngoài ra, việc sử dụng 3 Layer sẽ tăng tốc độ phản hồi, giảm phí gas, có thể lưu trữ dữ liệu không quan trọng ngoài blockchain.

Các bước xử lý khi sử dụng mô hình kiến trúc 3 Layer:

Bước 1: CIC admin gửi yêu cầu qua giao diện.

Bước 2: Backend chuyển yêu cầu tới hợp đồng thông minh thông qua API.

Bước 3: Hợp đồng thông minh xử lý logic tính toán hoặc truy xuất thông tin.

Bước 4: Kết quả được trả về giao diện để hiển thị.

2.1.2. Các công nghệ và công cụ được sử dụng

Để thiết kế và triển khai hợp đồng thông minh này, tác giả đã sử dụng những công nghệ và công cụ sau:

- Solidity (Phiên bản 0.8.0): Đây là ngôn ngữ lập trình bậc cao được thiết kế và tối ưu hóa để viết hợp đồng thông minh chạy trên Ethereum Virtual Machine (EVM).

- Remix: IDE trực tuyến giúp viết, biên dịch, kiểm thử và triển khai hợp đồng thông minh. Remix hỗ trợ kết nối với nhiều mạng Ethereum như mainnet, testnet (Rinkeby, Sepolia).

- MetaMask: Ví tiền điện tử và công cụ duyệt blockchain để thực hiện giao dịch, quản lý tài sản, tương tác tốt với các DApps. MetaMask có thể tương tác với Remix để triển khai hợp đồng thông minh trực tiếp qua trình duyệt.

- Etherscan Sepolia Testnet: Đây là trình duyệt blockchain giúp kiểm tra, giám sát và xác thực giao dịch trên mạng thử nghiệm Sepolia. Trình duyệt này có thể xác minh và xuất bản (verify & publish) mã nguồn hợp đồng thông minh công khai, đồng thời theo dõi giao dịch, hợp đồng đã triển khai. Cần sử dụng trình duyệt này để kiểm tra chi tiết về phí gas và các logic thuật toán trong hợp đồng, đảm bảo các giao dịch và hợp đồng thông minh hoạt động chính xác trên testnet trước khi triển khai trên mainnet.

- Sepolia Testnet: Đây là mạng thử nghiệm của Ethereum dùng để triển khai và kiểm thử hợp đồng thông minh mà không tiêu tốn tài nguyên thật. Sepolia chạy trên cơ chế đồng thuận giống Ethereum (PoS - Proof of Stake) và cho phép sử dụng ETH miễn phí (SepoliaETH) để thực hiện giao dịch. Cần sử dụng Sepolia để kiểm thử mã nguồn hợp đồng trong môi trường an toàn trước khi triển khai chính thức.

2.1.3. Mô tả chi tiết về hợp đồng thông minh

Hợp đồng thông minh để quản lý điểm tín dụng được chia thành các phần chính như sau:

Thứ nhất, thư viện để tính điểm tín dụng: CreditScoreLib. Thư viện này chứa các hàm pure tính toán điểm tín dụng.

- Điểm lịch sử thanh toán (tinhDiemLSTT).

- Điểm lượng dư nợ (tinhDiemLuongDuNo).

- Điểm độ dài lịch sử tín dụng (tinhDiemDoDaiLSTD).

- Điểm số lượng khoản vay (tinhDiemSoLuongKhoanVay).
- Điểm loại hình khoản vay (tinhDiemLoaiHinhKhoanVay).

Thứ hai, hợp đồng chính: QuanLyDiemTinDung. Hợp đồng này có các chức năng chính sau:

- Sử dụng thư viện CreditScoreLib để tính điểm tín dụng và phân loại.
- Tạo mã OTP cho từng hồ sơ, lưu trữ thời gian tạo và kiểm tra thời gian hiệu lực là 5 phút.
- Lưu trữ thông tin và sử dụng căn cước công dân, số điện thoại và OTP để truy xuất thông tin tín dụng.

2.2. Thiết kế và triển khai

2.2.1. Quy trình thiết kế hợp đồng thông minh

2.2.1.1. Quy trình thiết kế thư viện CreditScoreLib

Thư viện CreditScoreLib được thiết kế với mục tiêu tính toán các yếu tố cấu thành điểm tín dụng.



Thư viện được tính tương ứng trên công thức sau, với tổng số điểm là 750:

$$\text{Điểm tín dụng} = \text{Điểm lịch sử thanh toán} (35\% \times 750) + \text{Điểm lượng dư nợ} (30\% \times 750) + \text{Điểm độ dài lịch sử tín dụng} (15\% \times 750) + \text{Điểm số lượng khoản vay} (10\% \times 750) + \text{Điểm loại hình tín dụng} (10\% \times 750)$$

Trong đó:

- Lịch sử thanh toán: được tính dựa trên tỷ lệ số lần khách hàng trả nợ đúng hạn trong thời gian 12 tháng gần nhất.
- Lượng dư nợ: được tính dựa trên tỷ lệ dư nợ hiện tại của khách hàng so với hạn mức tín dụng.
- Độ dài lịch sử tín dụng: được tính dựa trên số năm khách hàng có lịch sử tín dụng.
- Số lượng khoản vay: được tính dựa trên số lượng khoản vay hiện tại của khách hàng.
- Loại hình khoản vay: được tính dựa trên mục đích sử dụng của các khoản vay.

Điểm thành phần chi tiết của từng loại điểm sẽ được ghi rõ tại công thức tính tại từng hàm trong thư viện. Ở đây, ta sử dụng hàm pure vì hàm này không truy cập trạng thái, loại bỏ khả năng truy cập trạng thái không cần thiết, duy trì tính thuần túy của thư viện. Đồng thời, hàm pure có bảo mật tốt do giới hạn phạm vi sử dụng chỉ trong các hợp đồng liên quan, lại có chi phí gas rẻ nhất nên rất phù hợp dùng trong thư viện. Sau khi tính điểm các yếu tố tín dụng này, ta sẽ thực hiện dùng một hàm trong hợp đồng chính để gọi thư viện thực hiện tính và phân loại tín dụng.

5 nhóm CIC sẽ được phân loại dựa trên thang điểm sau:

Nhóm 1: Từ 680 – 750: Rủi ro rất thấp, đây là nhóm khách hàng có điểm tín dụng CIC lý tưởng, đủ điều kiện vay, lãi suất thấp và được phê duyệt hạn mức vay cao.

Nhóm 2: Từ 570 – 679: Rủi ro thấp, khách hàng có khả năng trả nợ đúng hạn, đủ điều kiện vay, được xét duyệt lãi suất thấp.

Nhóm 3: Từ 431 – 569: Rủi ro trung bình, khách hàng đủ điều kiện vay nhưng xét duyệt lãi suất cao.

Nhóm 4: Từ 322 – 430: Rủi ro cao, khách hàng không đủ khả năng trả nợ.

Nhóm 5: Từ 150 – 321: Rủi ro rất cao, khách hàng không đủ điều kiện vay vốn.

2.2.1.2. Quy trình thiết kế hợp đồng QuanLyDiemTinDung

Trước khi thiết kế hợp đồng, ta cần xác định lại những yêu cầu đã đặt ra ở phần trên, đó là tính bảo mật dữ liệu, tính chính xác và tính minh bạch. Với hợp đồng này, các tác vụ cơ bản bao gồm:

- Quản lý thông tin khách hàng: Lưu trữ thông tin cơ bản và điểm tín dụng của từng cá nhân.

- Tính toán điểm tín dụng: Dựa trên các tiêu chí như lịch sử thanh toán, dư nợ, thời gian mở tài khoản tín dụng, số lượng khoản vay, và mục đích vay.

- Cấp phát OTP: Xác thực người dùng qua OTP khi truy vấn thông tin tín dụng.

Việc bảo mật thông tin có thể thực hiện bằng cách giới hạn chỉ CIC admin và những người dùng hợp lệ được cấp quyền truy cập mới được phép truy cập thông tin. Ngoài ra, hợp đồng còn sử dụng OTP để đảm bảo chỉ khi có sự đồng ý của khách hàng thì cán bộ ngân hàng mới có thể thực hiện tra cứu điểm tín dụng CIC.

Hợp đồng này bao gồm các biến và kiểu dữ liệu chính:

- Địa chỉ của admin quản lý hệ thống;

- Thông tin của khách hàng, được lưu tại Struct CIC và mapping thôngTinTD;

- Một số mapping khác để lưu mã OTP, thời gian phát hành OTP và đếm số lần phát hành OTP để đảm bảo tính ngẫu nhiên.

Hợp đồng chính bao gồm các hàm chính:

- Hàm `tinVaPhanLoaiTinDung`, được để ở trạng thái private. Do hàm này không cần tương tác với người dùng hay các hợp đồng khác, việc đặt nó ở trạng thái private giúp hạn chế truy cập không cần thiết. Bên cạnh đó, hàm private không thêm vào ABI của hợp đồng, giúp giảm kích thước bytecode và tối ưu hóa chi phí triển khai. Điều này giúp hợp đồng nhẹ hơn và chi phí gas thấp hơn khi triển khai trên blockchain. Hàm này gọi thư viện `CreditScoreLib` để tính toán và phân loại điểm tín dụng. Sau đó hàm phân loại khách hàng thành các nhóm CIC từ 1 đến 5 dựa trên thang điểm được cung cấp ở phần trên.

- Hàm `creditScoreCenter` cho phép admin cập nhật thông tin khách hàng và tính toán, phân loại điểm tín dụng. Hàm này đã được tích hợp với hàm `tinVaPhanLoaiTinDung` ở phần trên, giúp giảm thiểu thời gian và công việc cho CIC admin. Để cập nhật một hồ sơ mới, admin chỉ cần nhập các thông tin của khách hàng. Hàm `creditScoreCenter` sẽ tính và phân loại nhóm CIC của khách hàng và trả về kết quả.

- Hàm `taoVaGuiOTP` dùng để tạo mã OTP mới, đảm bảo ngẫu nhiên thông qua các yếu tố như `block.timestamp`, `msg.sender`, và `nonce`. Vì mỗi block có một giá trị

timestamp khác nhau, yếu tố này đảm bảo tính khác biệt cho mỗi lần gọi hàm trong các block khác nhau. Trong trường hợp có nhiều người dùng gọi hàm, msg.sender giúp tạo ra sự khác biệt giữa các giao dịch từ những người dùng khác nhau. Cuối cùng, Nonce được tăng mỗi lần tạo OTP cho cùng một người dùng (canCuocCongDan). Nonce đảm bảo rằng ngay cả khi các yếu tố khác (như block.timestamp và msg.sender) giống nhau, giá trị OTP vẫn sẽ khác nhau.

- Hàm traCuuThongTinTD cho phép người dùng hợp lệ truy vấn thông tin tín dụng, yêu cầu OTP phải hợp lệ và còn hiệu lực.

2.2.2. Triển khai hợp đồng thông minh trên testnet

Bước 1: Xác minh và xuất bản hợp đồng trên Etherscan

Verify & Publish Contract Source Code

Source code verification provides transparency for users interacting with smart contracts. By uploading the source code, Etherscan will match the compiled code with that on the blockchain. [Read more.](#)

A simple and structured interface for verifying smart contracts that fit in a single file.

1 Enter Contract Details

2 Verify & Publish

✓ Successfully generated Bytecode and ABI for Contract Address
[\[0x3b3454a79b6a322293e1cd62934bc2ade59ab4d6\]](#)

📄 Learn how to verify your contract on multiple blockchains with a single API key [here](#).


Code Reader ?


Compiler Output

Bước 2: Khai báo hồ sơ CIC tại hàm creditScoreCenter

creditScoreCenter

_canCuocCongDan:	001202016939
_ngayCap:	22/01/2022
_tenKhachHang:	Phạm Tuấn Linh
_soDienThoai:	0982827309
thanhToanDungHan:	10
soTienNo:	200
thoiGianMoTKTD:	12
soLuongTDSohuu:	2
mucDichVay:	1

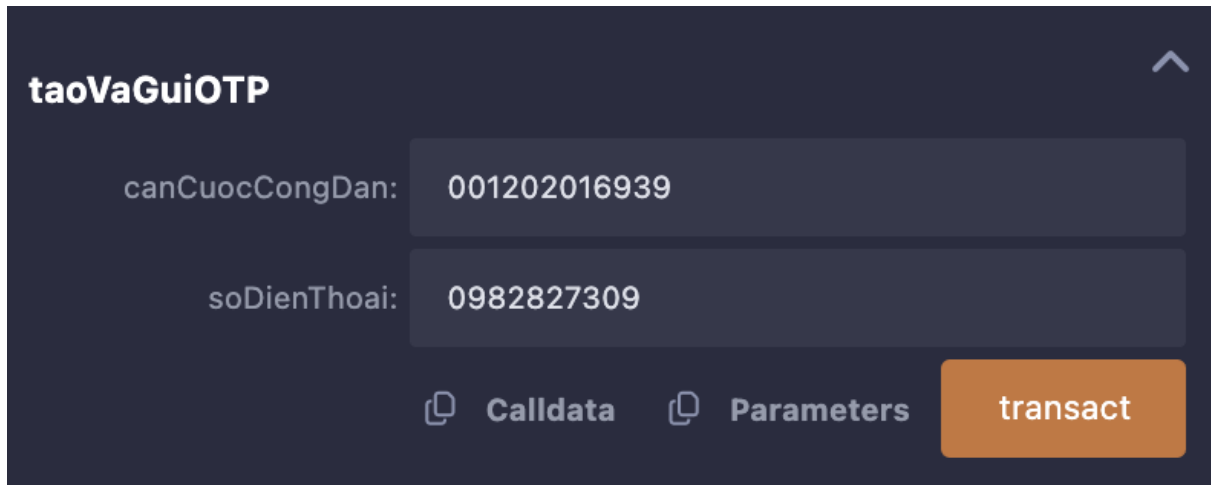
 Calldata

 Parameters

transact

Các thông tin bao gồm (1) Số căn cước công dân, (2) Ngày cấp, (3) Tên khách hàng, (4) Số điện thoại khách hàng, (5) Số lần thanh toán đúng hạn (tính trên 12 tháng gần nhất), (6) Số tiền đang nợ, (7) Thời gian mở tài khoản tín dụng, (8) Số lượng tín dụng đang sở hữu, (9) Mục đích vay nợ.

Bước 3: Gửi mã OTP đến số điện thoại khách hàng để xác nhận. Điều này đảm bảo rằng chỉ khi có sự đồng ý của khách hàng thì cán bộ ngân hàng mới có thể tra cứu thông tin và điểm CIC của khách hàng.



taoVaGuiOTP

canCuocCongDan: 001202016939

soDienThoai: 0982827309

Calldata Parameters **transact**

Bước 4: Nhận mã OTP tại cửa sổ Terminal trong giao diện của Remix. Đây chỉ là giải pháp tạm thời khi phát triển hợp đồng, trên thực tế blockchain không thể gửi OTP. Do đó ta cần sử dụng một bên thứ 3 cung cấp dịch vụ SMS gateway để gửi OTP về số điện thoại liên lạc đã đăng ký của khách hàng. Sau đó tích hợp thông tin này vào DApps để đối chiếu ngược lên blockchain.



```
input 0x15a...00000
decoded input {
  "string canCuocCongDan": "001202016939",
  "string soDienThoai": "0982827309"
}
decoded output -
logs [
  {
    "from": "0x3b3454a79b6a322293e1cd62934bc2ade59ab4d6",
    "topic": "0x465364d1a2fcb9c0b4750e419a357763742c4d178c61569982314a084119445",
    "event": "guiOTP",
    "args": {
      "0": {
        "_isIndexed": true,
        "hash": "0x37ebbe7920764cdccfd6d0afacc03f94a59d08e84e2872555737bbb91ddd1"
      },
      "1": "428504"
    }
  }
]
```

Bước 5: Tại hàm tra cứu thông tin tín dụng, nhập số căn cước nhân dân và OTP vừa nhận được để kiểm tra thông tin CIC cá nhân.

traCuuThongTinTD

canCuocCongDan:

001202016939

otp:

428504

Calldata

Parameters

call

0: string: _canCuocCongDan 001202016939

1: string: _ngayCap 22/01/2022

2: string: _tenKhachHang Phạm Tuấn Linh

3: string: _phanLoaiCIC CIC nhóm 2

4: string: _soDienThoai 0982827309

5: bool: hopLe true

Hàm traCuuThongTinTD đã được tích hợp với hàm tính và phân loại tín dụng để kết quả trả về phân loại nhóm CIC của khách hàng.

Lưu ý: Mã OTP này chỉ có hiệu lực trong 5 phút kể từ thời điểm gửi mã, nếu quá 5 phút thì khách hàng phải gửi yêu cầu tạo OTP lại. Mỗi hồ sơ tại một thời điểm chỉ có một mã duy nhất, không trùng lặp với nhau.

traCuuThongTinTD

canCuocCongDan:

001202016939

otp:

428503

Calldata

Parameters

call

Ví dụ như hình trên, mã OTP của ta là 428504, nếu ta nhập nhầm số khác thì kết quả trả về tại cửa sổ Terminal như sau:

```
from: 0x0E86BBb5476Ffc807C84Fd120Ea07222DaFC854c to: QuanLyDiemTinDung.
x56d...00000
DiemTinDung.traCuuThongTinTD
DiemTinDung.traCuuThongTinTD errored: execution reverted: OTP khong hop le
DiemTinDung.traCuuThongTinTD
```

Nếu quá 5 phút mà OTP chưa được nhập vào, mã OTP sẽ bị reset. Lúc này khách hàng cần yêu cầu tạo và gửi lại một mã khác.

```
[call] from: 0x0E86BBb5476Ffc807C84Fd120Ea07222DaFC854c
to: QuanLyDiemTinDung.traCuuThongTinTD(string,uint256) data: 0x56d...00000
call to QuanLyDiemTinDung.traCuuThongTinTD
call to QuanLyDiemTinDung.traCuuThongTinTD errored: execution reverted: OTP da het han
```

Sau khi thực hiện giao dịch, ta có thể kiểm tra lại trên block như sau:

The screenshot shows the Etherscan website interface. At the top, there's a navigation bar with 'Home', 'Blockchain', 'Tokens', 'NFTs', and 'More'. Below this, the main header shows 'Contract' with the address '0x3b3454A79B6a322293E1cd62934BC2ADe59ab4d6'. The 'Source Code' tab is selected. The 'Overview' section shows 'ETH BALANCE' as '0 ETH'. The 'More Info' section shows 'CONTRACT CREATOR' as '0x0E86BBb5...2DaFC854c' at transaction '0xd49dc...'. The 'Multichain Info' section shows 'N/A'. Below this, there are tabs for 'Transactions', 'Token Transfers (ERC-20)', 'Contract', and 'Events'. The 'Transactions' tab is selected, showing 'Latest 2 from a total of 2 transactions'. A 'Download Page Data' button is visible. The transaction list has columns: Transaction Hash, Method, Block, Age, From, To, and Amount. Two transactions are listed: one with hash '0x1a8628ad81...' and method 'Tao Va Gui OTP', and another with hash '0x598bd3cf191...' and method 'Credit Score C...'. Both transactions are from '0x0E86BBb5...2DaFC854c' to '0x3b3454A7...De59ab4d6' and have an amount of '0 ETH'.

Transaction Hash	Method	Block	Age	From	To	Amount
0x1a8628ad81...	Tao Va Gui OTP	7465067	15 hrs ago	0x0E86BBb5...2DaFC854c	0x3b3454A7...De59ab4d6	0 ETH
0x598bd3cf191...	Credit Score C...	7465061	15 hrs ago	0x0E86BBb5...2DaFC854c	0x3b3454A7...De59ab4d6	0 ETH

Ta có thể kiểm tra trực tiếp OTP của hàm taoVaGuiOTP tại block.

Source Code

Overview

ETH BALANCE

0 ETH

More Info

CONTRACT CREATOR

0x0E86BBb5...2DaFC854c at txn 0xd49dc...

Multichain Info

N/A

Transactions Token Transfers (ERC-20) Contract Events

Latest 2 Contract Events

Tip: Logs are used by developers/external UI providers for keeping track of contract actions and for auditing

Transaction Hash	Block	Age	Method	Logs
0x1a8628ad81...	7465067	15 hrs ago	0x15aa0356 taoVaGuiOTP(string,st...	guiOTP (index_topic_1 string canCuocCongDan, uint256 otp) [topic0] 0x465364d1a2fcba9c0b4750e419a357763742c4d178c [topic1] 0x37ebbe7920764cdcccf6d0afacc03f94a59d08e84e Num → 428504
0x598bd3cf191...	7465061	15 hrs ago	0x252d72da creditScoreCenter(stri...	capNhatDiemTD (index_topic_1 string canCuocCongDan, string ngayCap. [topic0] 0xeb67b77818cdc0af18440a66a4582fec9f81e2f4384 [topic1] 0x37ebbe7920764cdcccf6d0afacc03f94a59d08e84e Hex → 00 Hex → 00

2.2.3. Vấn đề kỹ thuật đã gặp phải

Khi thực hiện hợp đồng thông minh này, tác giả đã gặp phải một số vấn đề kỹ thuật:

- Solidity là một ngôn ngữ lập trình được thiết kế để sử dụng trên các blockchain. Do đó, phần code của Solidity không hoàn toàn giống với ngôn ngữ C. Tác giả phải điều chỉnh lại một số hàm và lệnh để hợp đồng có thể triển khai hợp lý.
- Khi triển khai hợp đồng quản lý tín dụng trên Remix vào lúc 10h tối, tốc độ băng thông rất chậm thì phí giao dịch yêu cầu từ MetaMask có thể lên tới \$500, tuy nhiên vẫn hợp đồng đó khi triển khai vào 10h sáng chỉ tốn \$15.

CHƯƠNG 3: ĐỊNH HƯỚNG VÀ GIẢI PHÁP HOÀN THIỆN HỢP ĐỒNG THÔNG MINH QUẢN LÝ ĐIỂM TÍN DỤNG CIC

3.1. Kết luận và hướng phát triển

3.1.1. Tổng kết các kết quả đạt được

Thứ nhất, hợp đồng được triển khai đã đáp ứng đủ 3 tiêu chí được đề ra khi bắt đầu xây dựng: tính bảo mật, tính chính xác và tính minh bạch. Các thuật toán tính điểm được mã hóa rõ ràng trong hợp đồng. Mọi thao tác cập nhật điểm tín dụng đều được ghi lại thông qua events, có thể tra cứu và kiểm chứng. Dữ liệu được lưu trữ phi tập trung trên blockchain, không thể bị sửa đổi hay xóa bỏ, chỉ CIC admin được phép cập nhật thông tin điểm tín dụng.

Thứ hai và cũng là điểm sáng lớn nhất của hợp đồng là triển khai cơ chế xác thực OTP để bảo vệ việc truy cập thông tin. Mã hóa và kiểm tra khớp số điện thoại khi tạo OTP. Thêm vào đó, cơ chế timeout được thêm cho OTP để tăng tính bảo mật.

Cuối cùng là hợp đồng xây dựng được 5 tiêu chí đánh giá rõ ràng với trọng số phù hợp. Những tiêu chí này được căn cứ theo pháp luật hiện hành, thực hiện phân loại khách hàng thành 5 nhóm tín dụng dựa trên điểm số. Các tiêu chí đánh giá được mã hóa thành các hàm riêng biệt, dễ bảo trì.

Bên cạnh đó, tác giả cũng đúc kết được một số bài học kinh nghiệm như sau:

Về thiết kế hợp đồng, cần phân tách logic thành các thư viện và hợp đồng riêng biệt để dễ quản lý.

Về bảo mật, cần triển khai nhiều lớp bảo mật (admin, OTP, timeout).

Về hiệu năng, code đã được tối ưu để tiết kiệm phí gas. Số lượng events được hạn chế để không làm tăng kích thước blockchain.

3.1.2. Đề xuất hướng phát triển trong tương lai

Để hợp đồng này có thể được đầu tư và đi vào triển khai trong thực tiễn, cần hoàn thiện một số tiêu chí sau:

- Cải thiện code để phát key modifier hợp đồng cho nhiều bên (tức nhiều ngân hàng), cho phép có nhiều CIC admin tại một thời điểm;
- Kết hợp với bên thứ 3 cung cấp dịch vụ SMS gateway để gửi OTP, tích hợp mã OTP này vào blockchain, đảm bảo phải khớp với thông tin off-chain;
- Thêm cơ chế khôi phục quyền admin trong trường hợp mất khóa;

- Bổ sung chức năng khôi phục quyền truy cập;
- Tăng cường bảo mật bằng cách mã hóa dữ liệu nhạy cảm.

3.2. Tài liệu tham khảo và phụ lục

3.2.1. Danh sách các nguồn tài liệu tham khảo đã sử dụng

1. <https://tapchinganhang.gov.vn/an-toan-bao-mat-thong-tin-tai-khoan-cua-khach-hang-nhiem-vu-quan-trong-cua-ngan-hang-trong-cung-cap-dich-vu-9475.html>
2. <https://tapchinganhang.gov.vn/bao-mat-thong-tin-khach-hang-khi-su-dung-dich-vu-ngan-hang-so-thuc-trang-va-mot-so-kien-nghi-9421.html>
3. <https://dantri.com.vn/kinh-doanh/can-bo-bi-to-lam-lo-thong-tin-cic-cua-2-ca-nhan-ngan-hang-van-im-lang-20240725104434135.htm>
4. <https://dantri.com.vn/kinh-doanh/2-nguoi-tai-tphcm-to-bi-can-bo-mot-ngan-hang-lam-lo-thong-tin-tin-dung-20240626093851216.htm>
5. <https://luatvietnam.vn/linh-vuc-khac/cic-la-gi-883-98155-article.html>
6. <https://thuvienphapluat.vn/van-ban/Thong-tu-11-2021-TT-NHNN-su-dung-du-phong-de-xu-ly-rui-ro-hoat-dong-cua-to-chuc-tin-dung-483459.aspx>

3.2.2. Phụ lục: Toàn bộ code của hợp đồng thông minh

```
// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

// Thư viện tính điểm tín dụng
library CreditScoreLib {
    uint constant MAX_SCORE = 750;
    // (1) Tính điểm lịch sử thanh toán
    function      tinhDiemLSTT(uint      thanhToanDungHan)
internal pure returns (uint) {
        require(thanhToanDungHan <= 12, "Khong hop
le.");

        if (thanhToanDungHan > 10) {
            return (35 * MAX_SCORE) / 100;
        } else if (thanhToanDungHan == 10) {
            return (30 * MAX_SCORE) / 100;
        } else if (thanhToanDungHan == 9) {
            return (25 * MAX_SCORE) / 100;
        } else if (thanhToanDungHan == 8) {
            return (20 * MAX_SCORE) / 100;
        } else if (thanhToanDungHan == 7) {
            return (15 * MAX_SCORE) / 100;
        } else if (thanhToanDungHan == 6) {
            return (10 * MAX_SCORE) / 100;
        } else if (thanhToanDungHan == 5) {
            return (5 * MAX_SCORE) / 100;
        } else {
            return 0;
        }
    }
}

// (2) Tính điểm lượng dư nợ
```



```

function tinhDiemLuongDuNo(uint soTienNo) internal
pure returns (uint) {
    if (soTienNo > 1000) {
        return 0;
    } else if (soTienNo > 800) {
        return (5 * MAX_SCORE) / 100;
    } else if (soTienNo > 600) {
        return (10 * MAX_SCORE) / 100;
    } else if (soTienNo > 400) {
        return (15 * MAX_SCORE) / 100;
    } else if (soTienNo > 200) {
        return (20 * MAX_SCORE) / 100;
    } else if (soTienNo > 100) {
        return (25 * MAX_SCORE) / 100;
    } else {
        return (30 * MAX_SCORE) / 100;
    }
}

// (3) Tính điểm độ dài lịch sử tín dụng
function tinhDiemDoDaiLSTD(uint thoiGianMoTKTD)
internal pure returns (uint) {
    if (thoiGianMoTKTD > 15) {
        return (15 * MAX_SCORE) / 100;
    } else if (thoiGianMoTKTD > 10) {
        return (10 * MAX_SCORE) / 100;
    } else if (thoiGianMoTKTD > 5) {
        return (5 * MAX_SCORE) / 100;
    } else {
        return 0;
    }
}

```

```

// (4) Tính điểm số lượng khoản vay
function          tinhDiemSoLuongKhoanVay(uint
soLuongTDSoHuu) internal pure returns (uint) {
    if (soLuongTDSoHuu == 0) {
        return (10 * MAX_SCORE) / 100;
    } else if (soLuongTDSoHuu < 3) {
        return (5 * MAX_SCORE) / 100;
    } else {
        return 0;
    }
}

// (5) Tính điểm loại hình khoản vay
function tinhDiemLoaiHinhKhoanVay(uint mucDichVay)
internal pure returns (uint) {
    if (mucDichVay == 1) {
        return (10 * MAX_SCORE) / 100;
    } else if (mucDichVay == 2) {
        return (5 * MAX_SCORE) / 100;
    } else {
        return 0;
    }
}
}

contract QuanLyDiemTinDung {
    address private cicAdmin;
    mapping(string => uint) private activeOTP;
    mapping(string => uint) private otpTimestamp;
    mapping(string => uint) private otpNonce;
    uint constant private OTP_VALIDITY_PERIOD = 5
minutes;

```

```

    constructor() {
        cicAdmin = msg.sender;
    }

    modifier onlyCICAdmin() {
        require(msg.sender == cicAdmin, "Only admin
can perform this action.");
        _;
    }

    struct CIC {
        string canCuocCongDan;
        string ngayCap;
        string tenKhachHang;
        string phanLoaiCIC;
        string soDienThoai;
        bool hopLe;
    }

    mapping(string => CIC) private thongTinTD;

    event capNhatDiemTD(
        string indexed canCuocCongDan,
        string ngayCap,
        string tenKhachHang,
        string phanLoaiCIC,
        string soDienThoai
    );

    event guiOTP(string indexed canCuocCongDan, uint
otp);

    using CreditScoreLib for uint;

```

```

function tinhVaPhanLoaiTinDung(
    uint thanhToanDungHan,
    uint soTienNo,
    uint thoiGianMoTKTD,
    uint soLuongTDSoHuu,
    uint mucDichVay
) private pure returns (string memory phanLoaiTD)
{
    uint diem1 = thanhToanDungHan.tinhDiemLSTT();
    uint diem2 = soTienNo.tinhDiemLuongDuNo();
    uint diem3 =
thoiGianMoTKTD.tinhDiemDoDaiLSTD();
    uint diem4 =
soLuongTDSoHuu.tinhDiemSoLuongKhoanVay();
    uint diem5 =
mucDichVay.tinhDiemLoaiHinhKhoanVay();

    uint diemTinDung = diem1 + diem2 + diem3 +
diem4 + diem5;

    if (diemTinDung >= 680) {
        return "CIC nhom 1";
    } else if (diemTinDung >= 570) {
        return "CIC nhom 2";
    } else if (diemTinDung >= 431) {
        return "CIC nhom 3";
    } else if (diemTinDung >= 322) {
        return "CIC nhom 4";
    } else {
        return "CIC nhom 5";
    }
}

```

```

    }

    function creditScoreCenter(
        string memory _canCuocCongDan,
        string memory _ngayCap,
        string memory _tenKhachHang,
        string memory _soDienThoai,
        uint thanhToanDungHan,
        uint soTienNo,
        uint thoiGianMoTKTD,
        uint soLuongTDSoHuu,
        uint mucDichVay
    ) public onlyCICAdmin {
        string          memory          phanLoaiCIC          =
    tinhVaPhanLoaiTinDung(
        thanhToanDungHan,
        soTienNo,
        thoiGianMoTKTD,
        soLuongTDSoHuu,
        mucDichVay
    );
    thôngTinTD[_canCuocCongDan] = CIC({
        canCuocCongDan: _canCuocCongDan,
        ngayCap: _ngayCap,
        tenKhachHang: _tenKhachHang,
        phanLoaiCIC: phanLoaiCIC,
        soDienThoai: _soDienThoai,
        hopLe: true
    });
    emit capNhatDiemTD(_canCuocCongDan, _ngayCap,
    _tenKhachHang, phanLoaiCIC, _soDienThoai);
    }

```

```

        function taoVaGuiOTP(string memory canCuocCongDan,
string memory soDienThoai)
        public returns (uint) {
            require(thongTinTD[canCuocCongDan].hopLe,
"Thong tin khong hop le.");
            require(

keccak256(abi.encodePacked(thongTinTD[canCuocCongDan].so
DienThoai)) ==
                keccak256(abi.encodePacked(soDienThoai)),
                "So dien thoai khong khop"
            );
            // Tăng nonce cho mỗi lần tạo OTP
            otpNonce[canCuocCongDan]++;
            // Tạo OTP mới
            uint newOTP = uint(keccak256(abi.encodePacked(
                block.timestamp,
                msg.sender,
                otpNonce[canCuocCongDan],
                canCuocCongDan,
                block.prevrandao
            ))) % 900000 + 100000;
            // Lưu OTP mới và thời gian tạo
            activeOTP[canCuocCongDan] = newOTP;
            otpTimestamp[canCuocCongDan] =
block.timestamp;
            emit guiOTP(canCuocCongDan, newOTP);
            return newOTP;
        }

        Function        traCuuThongTinTD(string        memory
canCuocCongDan, uint otp)

```

```

public view returns (
    string memory _canCuocCongDan,
    string memory _ngayCap,
    string memory _tenKhachHang,
    string memory _phanLoaiCIC,
    string memory _soDienThoai,
    bool hopLe)
{
    CIC          memory          thôngTin          =
thôngTinTD[canCuocCongDan];
    require(thôngTin.hopLe, "Thông tin không
hop le");
    // Kiểm tra OTP có tồn tại và còn hiệu
lực
    require(activeOTP[canCuocCongDan] == otp,
"OTP không hop le");
    require(block.timestamp          -
otpTimestamp[canCuocCongDan]
<= OTP_VALIDITY_PERIOD, "OTP đã hết hạn");
    return (
        thôngTin.canCuocCongDan,
        thôngTin.ngayCap,
        thôngTin.tenKhachHang,
        thôngTin.phanLoaiCIC,
        thôngTin.soDienThoai,
        thôngTin.hopLe);
}
}

```