LỜI CAM ĐOAN

Tôi xin cam đoan đồ án này là do chính tôi thực hiện.

Tất cả những tham khảo từ các nghiên cứu liên quan đều được nêu nguồn gốc một cách rõ ràng từ danh mục tài liệu tham khảo trong đồ án. Trong đồ án, không có việc sử dụng tài liệu, công trình nghiên cứu của người khác mà không chỉ rõ về tài liệu tham khảo. Các chương trình thực nghiệm đều là do tôi thực hiện mà có được, không sao chép từ bất cứ nguồn nào.

Sinh viên thực hiện

Trần Thị Diệu Ninh

LỜI NÓI ĐẦU

Vài năm trở lại đây, sự bùng nổ mạnh mẽ của các đồng tiền ảo Bitcoin (BTC), đồng Etherium (ETH), đồng Lifecoin (LTC)... và sự tăng giá lên tới hàng trăm lần của nhiều đồng tiền ảo trong năm 2017 đã khiến thị trường tiền ảo trở thành thị trường đầu tư hấp dẫn và sinh lời cao, thu hút mạnh sự quan tâm của giới đầu tư, đầu cơ, quỹ đầu tư lớn, các nhà đầu tư cá nhân.

Điểm qua một vài con số có thể thấy, tính đến cuối năm 2017, Bitcoin tăng giá khoảng gần 20 lần, đồng ETH tăng 120 lần và hàng trăm đồng coins khác cũng tăng giá rất nhiều lần trong một thời gian ngắn.

Thị trường tiền ảo có đặc điểm thu hút các nhà đầu tư, đó là tăng nhanh, tăng nhiều (có thể tăng 50-100 lần/ngày), tăng không có công thức và cũng rất khó dự đoán. Ngược lại, đà giảm sâu thì giảm cũng xuống rất nhanh, có thể giá trị của nhà đầu tư nhìn thấy thị trường này thực sự màu mỡ để đầu tư và sinh lời nhanh, bởi nguyên tắc rủi ro cao thì lợi nhuận cao. Nên nhiều ứng dụng phần mềm đã được lập ra nhằm để đào, mua bán, trao đổi, quản lý nhằm tăng lợi nhuận và tránh rủi ro. Vì vậy, em đã chọn đề tài “Xây dựng ứng dụng Bitcoin Wallet trên nền tảng Android và IOS” với mục đích nghiên cứu, tìm hiểu về ứng dụng phần mềm.

Nội dung báo cáo gồm 3 chương:

**Chương 1:** Giới thiệu tổng quan

**Chương 2**: Phân tích và thiết kế hệ thống

**Chương 3:** Xây dựng ứng dụng

**Kết luận và hướng nghiên cứu trong tương lai**

**LỜI CẢM ƠN**

Để có được thành quả học tập như ngày hôm nay, ngoài sự nỗ lực phấn đấu không ngừng của bản thân thì một phần không nhỏ đóng góp nên thành công ấy là sự sự hướng dẫn, dạy dỗ của các thầy cô trong khoa Công nghệ thông tin nói riêng và trường Đại học Điện Lực nói chung trong suốt gần năm năm em học tập và nghiên cứu tại trường.

Em xin gửi lời cảm ơn chân thành đến thầy Th.S Ngô Ngọc Thành – giảng viên khoa Công nghệ thông tin, trường Đại học Điện Lực đã định hướng, hướng dẫn và chỉ bảo tận tình trong quá trình em làm báo cáo tốt nghiệp này. Em xin chúc thầy và gia đình luôn luôn mạnh khoẻ, hạnh phúc.

Cuối cùng, em xin được cảm ơn đến gia đình, bạn bè đã động viên, đóng góp ý kiến và giúp đỡ em trong quá trình học tập, nghiên cứu và hoàn thành đồ án tốt nghiệp này. Tuy nhiên, do thời gian và trình độ có hạn nên báo cáo không thể tránh khỏi những thiếu sót. Chính vì vậy, em rất mong có được sự góp ý từ các thầy cô để em đạt được kết quả tốt nhất.

Em xin chân thành cảm ơn!

Sinh viên thực hiện

Trần Thị Diệu Ninh

**PHIẾU CHẤM ĐIỂM**

Sinh viên thực hiện:

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **Chữ ký** | **Ghi chú** |
| Trần Thị Diệu Ninh |  |  |

Giảng viên hướng dẫn chấm:

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **Chữ ký** | **Điểm** |
|  |  |  |

**MỤC LỤC**

[CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN 1](#_Toc29207195)

[1.1 Các đồng tiền ảo nổi tiếng 1](#_Toc29207196)

[1.1.1 Đồng Bitcoin (BTC) 1](#_Toc29207197)

[1.1.2 Đồng Ethereum (ETH) 1](#_Toc29207199)

[1.2 Ví Bitcoin 2](#_Toc29207201)

[1.2.1 Tổng quan về công nghệ ví 2](#_Toc29207202)

[1.2.2 Chi tiết về công nghệ ví 3](#_Toc29207204)

[1.2.2.1 Cụm từ mã trợ nhớ (BIP-39) 3](#_Toc29207205)

[1.2.2.2 Passphrase tuỳ chọn trong BIP-39 6](#_Toc29207208)

[1.2.2.3 Tạo ví HD từ hạt giống 7](#_Toc29207209)

[1.2.2.4 Tạo khoá bí mật con 8](#_Toc29207211)

[1.2.2.5 Tạo khoá công khai con 10](#_Toc29207213)

[1.3 Bài toán 11](#_Toc29207215)

[1.4 Nội dung đề tài 12](#_Toc29207217)

[1.5 Kết luận chương 1 13](#_Toc29207219)

[CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 14](#_Toc29207220)

[2.1 Biểu đồ Usecase 14](#_Toc29207221)

[2.1.1 Biểu đồ Usecase tổng quát 14](#_Toc29207222)

[2.1.2 Biểu đồ phân rã Usecase 15](#_Toc29207224)

[2.1.3 Các tác nhân chính tham gia hệ thống 18](#_Toc29207241)

[2.2 Đặc tả và hiện thực hóa Usecase 19](#_Toc29207243)

[2.3 Kết luận chương 2 28](#_Toc29207268)

[CHƯƠNG 3: XÂY DỰNG ỨNG DỤNG 29](#_Toc29207269)

[3.1. Công nghệ, công cụ và môi trường phát triển 29](#_Toc29207270)

[3.2 Giao diện hệ thống 35](#_Toc29207271)

[3.2.1 Màn hình home 35](#_Toc29207272)

[3.2.2 Màn hình Send 36](#_Toc29207274)

[3.2.3. Màn hình Receive 37](#_Toc29207276)

[3.2.4. Màn hình Exchange 39](#_Toc29207278)

[3.2.5 Màn hình Helps 40](#_Toc29207280)

[3.2.6 Màn hình Settings 41](#_Toc29207282)

[3.2.7 Màn hình History 42](#_Toc29207284)

[KẾT LUẬN 43](#_Toc29207286)

**DANH MỤC CÁC BẢNG**

[Bảng 2.1 Mô tả Usecase của hệ thống 18](#_Toc29207242)

**DANH MỤC CÁC HÌNH ẢNH**

[Hình 1.1 Logo Bitcoin 1](#_Toc29207198)

[Hình 1.2 Logo Ethereum 2](#_Toc29207200)

[Hình 1.3 Tạo entropy và mã hoá thành cụm từ trợ nhớ 4](#_Toc29207206)

[Hình 1.4 Từ cụm trợ nhớ đến hạt giống 6](#_Toc29207207)

[Hình 1.5 Tạo khoá chính và mã chuỗi từ một hạt giống gốc 8](#_Toc29207210)

[Hình 1.6 Mở rộng khoá bí mật mẹ để tạo khoá bí mật con 9](#_Toc29207212)

[Hình 1.7 Mở rộng khoá công khai mẹ để tạo ra khoá công khai con 11](#_Toc29207214)

[Hình 2.1 Usecase tổng quát 14](#_Toc29207223)

[Hình 2.2 Biểu đồ phân rã Usecase create wallet 15](#_Toc29207226)

[Hình 2.3 Biểu đồ phân rã Usecase delete wallet 15](#_Toc29207228)

[Hình 2.4 Biểu đồ phân rã Usecase import wallet 15](#_Toc29207230)

[Hình 2.5 Biểu đồ phân rã Usecase deposit 16](#_Toc29207232)

[Hình 2.6 Biểu đồ phân rã Usecase withdraw 16](#_Toc29207234)

[Hình 2.7 Biểu đồ phân rã Usecase exchange 16](#_Toc29207236)

[Hình 2.8 Biểu đồ phân rã Usecase helps 16](#_Toc29207238)

[Hình 2.9 Biểu đồ phân rã Usecase withdraw 17](#_Toc29207240)

[Hình 2.10 Biểu đồ hoạt động create wallet 19](#_Toc29207245)

[Hình 2.11 Biểu đồ trình tự create wallet 20](#_Toc29207247)

[Hình 2.12 Biểu đồ hoạt động import wallet 21](#_Toc29207249)

[Hình 2.13 Biểu đồ trình tự import wallet 21](#_Toc29207250)

[Hình 2.14 Biểu đồ hoạt động delete wallet 22](#_Toc29207252)

[Hình 2.15 Biểu đồ trình tự delete wallet 22](#_Toc29207253)

[Hình 2.16 Biểu đồ hoạt động withdraw 23](#_Toc29207255)

[Hình 2.17 Biểu đồ trình tự withdraw 24](#_Toc29207256)

[Hình 2.18 Biểu đồ hoạt động deposit 24](#_Toc29207258)

[Hình 2.19 Biểu đồ trình tự deposit 25](#_Toc29207259)

[Hình 2.20 Biểu đồ hoạt động exchange 26](#_Toc29207260)

[Hình 2.21 Biểu đồ trình tự exchange 26](#_Toc29207261)

[Hình 2.22 Biểu đồ hoạt động helps 27](#_Toc29207263)

[Hình 2.23 Biểu đồ trình tự helps 27](#_Toc29207264)

[Hình 2.24 Biểu đồ hoạt động Thay đổi thông tin cá nhân 28](#_Toc29207266)

[Hình 2.25 Biểu đồ trình tự Thay đổi thông tin cá nhân 28](#_Toc29207267)

[Hình 3.1 Màn hình home 36](#_Toc29207273)

[Hình 3.2 Màn hình Send 37](#_Toc29207275)

[Hình 3.3 Màn hình Receive 38](#_Toc29207277)

[Hình 3.4 Màn hình Exchange 39](#_Toc29207279)

[Hình 3.5 Màn hình Helps 40](#_Toc29207281)

[Hình 3.6 Màn hình Settings 41](#_Toc29207283)

[Hình 3.7 Màn hình History 42](#_Toc29207285)

1. **DANH MỤC CÁC KÝ HIỆU**

|  |  |  |  |
| --- | --- | --- | --- |
| **PHẦN TỬ MÔ HÌNH** | | **KÝ HIỆU** | **Ý NGHĨA** |
| **Biểu đồ USE CASE** | | | |
| Tác nhân  (Actor) | | a | Một người / nhóm người hoặc một thiết bị hoặc hệ thống tác động hoặc thao tác đến chương trình. |
| Use-case  (“Ca” sử dụng) | | uc | Biểu diễn một chức năng xác định của hệ thống |
| Mối quan hệ giữa các use case | |  | Use case này sử dụng lại chức năng của use case kia |
|  | Use case này mở rộng từ use case kia bằng cách thêm chức năng cụ thể |
|  | Use case này kế thừa các chức năng từ use case kia |
| **Biểu đồ LỚP** | | | |
| Lớp  (Class) | | class1 | Biểu diễn tên lớp, thuộc tính, và phương thức của lớp đó |
| Quan hệ kiểu kết hợp | |  | Biểu diễn quan hệ giữa hai lớp độc lập, có liên quan đến nhau |
| Quan hệ hợp thành | |  | Biểu diễn quan hệ bộ phận – tổng thể |
| Quan hệ phụ thuộc | |  | Các lớp phụ thuộc lẫn nhau trong hoạt động của hệ thống |
| **Biểu đồ TRẠNG THÁI** | | | |
| Trạng thái | |  | Biểu diễn trạng thái của đối tượng trong vòng đời của đối tượng đó |
| Trạng thái khởi đầu | |  | Khởi đầu vòng đời của đối tượng đó |
| Trạng thái kết thúc | |  | Kết thúc vòng đời của đối tượng |
| Chuyển tiếp  (transition) | |  | Chuyển từ trạng thái này sang trạng thái khác |
| **Biểu đồ TUẦN TỰ** | | | |
| Procedure  (Phương thức) | | s | Là một phương thức của B mà đối tượng A gọi thực hiện. |
| Message  (Thông điệp) | | m | Là một thông báo mà B gửi cho A. |
| **Biểu đồ HOẠT ĐỘNG** | | | |
| Hoạt động |  | | Mô tả hoạt động gồm tên hoạt động và đặc tả của nó |
| Trạng thái khởi đầu |  | |  |
| Trạng thái kết thúc |  | |  |
| Thanh đồng bộ ngang |  | | Mô tả thanh đồng bộ ngang |
| Chuyển tiếp |  | |  |
| Quyết định |  | | Mô tả một lựa chọn điều kiện |
| Các luồng | Phân tách các lớp đối tượng khác nhau trong biểu đồ hoạt động | | Phân cách nhau bởi một đường kẻ dọc từ trên xuống dưới biểu đồ |
| **Biểu đồ THÀNH PHẦN** | | | |
| Thành phần | |  | Mô tả một thành phần của biểu đồ, mỗi thành phần có thể chứa nhiều lớp hoặc nhiều chương trình con |
| Mối quan hệ phụ thuộc giữa các thành phần | |  | Mỗi quan hệ giữa các thành phần(nếu có) |
| **Biểu đồ TRIỂN KHAI** | | | |
| Các node  (các thiết bị) |  | | Biểu diễn các thành phần không có bộ vi xử lý |
| Các bộ xử lý |  | | Biểu diễn các thành phần có bộ vi xử lý |
| Liên kết truyền thông TCP/IP |  | | Giao thức truyền thông TCP/IP thông qua kết nói mạng LAN |

**DANH MỤC CÁC THUẬT NGỮ**

*BIP*

Đây là từ viết tắt của cụm Bitcoin Improvement Proposals (các Đề xuất Cải tiến Bitcoin), chỉ một tập hợp đề xuất mà các thành viên của cộng đồng bitcoin đã gửi để cải tiến bitcoin. Ví dụ, BIP-21 là một đề xuất cải tiến cơ chế định dạng tài nguyên thống nhất (URI) của bitcoin.

*URI (định dạng tài nguyên thống nhất – Uniform Resource Indentifier)*

Trong lĩnh vực công nghệ thông tin, một URI là một chuỗi các ký tự dùng để xác định một tài nguyên (hay còn gọi là tài nguyên hệ thống, tức các hợp phần vật lý hoặc ảo với số lượng hữu hạn trong một hệ thống máy tính, chẳng hạn như CPU, bộ nhớ, thiết bị ngoại vi…)

*bitcoin*

Tên gọi của một đơn vị tiền tệ (coin), một mạng lưới và một phần mềm.

*block (khối dữ liệu)*

Là một nhóm các giao dịch được đánh dấu bằng một nhãn thời gian và một đặc điểm nhận dạng [vân tay số] của block trước đó. Phần tiêu đề block được băm để tạo bằng chứng xử lý, qua đó xác thực các giao dịch. Các block hợp lệ được thêm vào blockchain chính thông qua cơ chế đồng thuận mạng lưới.

*blockchain (chuỗi khối dữ liệu)*

Là một danh sách các block đã được xác thực, mỗi block liên kết với block liền trước nó và cứ như thế truy ngược về block gốc (genesis block).

*genesis block (block gốc)*

Là block đầu tiên của một blockchain, được dùng để khởi tạo tiền mã hoá

*băm (hashing)*

Là qúa trình sử dụng một hàm toán học để tạo ra một hoặc nhiều giá trị từ một chuỗi ký tự. Đây là một kỹ thuật giúp bảo đảm an ninh trong quá trình truyền tin. Khi người dùng gửi đi một dữ liệu, một hàm băm của dữ liệu sẽ được tạo ra và mã hoá rồi được gửi đi cùng với dữ liệu. Người nhận dữ liệu sẽ giải mã mã băm cùng với dữ liệu rồi tạo một mã băm khác từ dữ liệu nhận được. Nếu hai mã băm này giống nhau, có nghĩa dữ liệu đã được truyền đi an toàn. Quá trình băm này giúp bảo đảm rằng dữ liệu được truyền không bị thay đổi bởi người dùng cuối nào không được uỷ quyền.

*hash (mã băm)*

Là mã số có chiều dài cố định của một chuỗi dữ liệ có độ dài bất kỳ.

*HD wallet (ví HD [ví tất định phân cấp])*

Là ví sử dụng giao thức tất định phân cấp.

*giao thức tất định phân cấp*

Là giao thức có hành vi được quyết định hoàn toàn bởi trạng thái ban đầu và các dữ liệu đầu vào, loại trừ yếu tố ngẫu nhiên. Các giao thức, thuật toán, mô hình, quá trình… chỉ có một kết quả đẩu ra duy nhất được gọi là tất định, xét theo nghĩa đầu ra của chúng đã được “quyết định từ trước”.

*ví tất định (deterministic wallet)*

Là một hệ thống tạo các khoá từ một điểm khởi đầu duy nhất gọi là hạt giống (seed). Hạt giống này cho phép người dùng có thể dễ dàng sao lưu ví mà không cần thêm bất kỳ thông tin nào khác, và trong một số trường hợp còn có thể cho phép tạo địa chỉ công khai mà không cần biết về khoá bí mật.

*satoshi*

Là mệnh giá nhỏ nhất của bitcoin có thể được ghi lại trên blockchain. 1 satoshi tương đương với 0,00000001 bitcoin; đây là tên được đặt theo tên của người sáng tạo ra bitcoin, Satoshi Nakamoto.

*Satoshi Nakamoto*

Là danh xưng của thực tể (có thể là một hoặc nhiều người) đã thiết kế ra bitcoin và bản thực hiện tham chiếu gốc, Bitcoin Core, trong đó Nakamoto cũng để lại cơ sở dữ liệu blockchain đầu tiên. Trong quá trình này, Nakamoto trở thành người đầu tiên giải quyết được bài toán lặp chi cho tiền số. Danh tính thật của Nakamoto cho đến nay vẫn còn là một bí ẩn.

*secreat key (hay còn gọi là private key - khoá bí mật)*

Là một dãy số bí mật có thể mở khoá lượng bitcoin được gửi đến một địa chỉ tương ứng.

*SHA (Secure Hash Algorithm - thuật toán băm an toàn)*

Thuật toán băm an toàn là một họ các hàm băm mã hoá do Viện Tiêu chuẩn và Kỹ thuật Quốc gia Mỹ (NIST) công bố.

*transaction – giao dịch*

Nói một cách đơn giản, giao dịch là một sự chuyển giao bitcoin từ địa chỉ này đến địa chỉ khác. Chính xác hơn, một giao dịch là một cấu trúc dữ liệu đã được ký, thể hiện một sự chuyển giao giá trị. Các giao dịch được truyền qua mạng bitcoin, do các thợ đào thu thập lại, và được thêm vào các block, trở thành vĩnh viễn trên blockchain.

# 

# CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN

* 1. **Các đồng tiền ảo nổi tiếng**

Hiện nay trên thế giới có hơn 2000 đồng tiền ảo khác nhau. Và Bitcoin, Ethereum vẫn là hai đồng tiền điện tử dẫn đầu thị trường tiền ảo.

* + 1. **Đồng Bitcoin (BTC)**

Bitcoin (ký hiệu BTC, XBT) là một loại tiền mã hoá, được phát minh bởi 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.



*Hình 1.1 Logo Bitcoin*

* + 1. **Đồng Ethereum (ETH)**

Ethereum (ETH) hay còn gọi là Bitcoin 2.0 là một nền tảng điện toán phân tán khối chuỗi chạy trên công nghệ blockchain, thông qua việc sử dụng chức năng hợp đồng thông minh (smart contract). Tiền ảo Ethereum có thể thực hiện các giao dịch, hợp đồng mạng ngang hàng thông qua đơn vị tiền ảo là Ether. Không những thế ETH còn được đánh giá là một nền tảng ứng dụng hữu ịch và tự tạo ra hệ tài chính phân tán của riêng mình. Ethereum được giới thiệu vào cuối năm 2013 bởi một chuyên viên nghiên cứu về lập trình tiền ảo có tên Vitalik Buterin.



*Hình 1.2 Logo Ethereum*

* 1. **Ví Bitcoin**

Từ “ví” dùng để nói chỉ một số yếu tố khác nhau trong bitcoin. Ở mức khái quát, ví là ứng dụng đóng vai trò giao diện người dùng chính. Ví kiểm soát quyền truy cập tiền của người dùng, quản lý các khoá và địa chỉ, theo dõi số dư, tạo và ký các giao dịch. Ở góc độ hẹp hơn, từ góc nhìn của một lập trình viên, từ “ví” chỉ cấu trúc dữ liệu dùng để lưu trữ và quản lý các khoá của người dùng.

* + 1. **Tổng quan về công nghệ ví**

Trong phần này, chúng ta sẽ nêu tóm lược về các công nghệ khá nhau dùng để tạo ra các ví bitcoin thân thiện với người dùng, an toàn và linh hoạt.

Một sự nhầm lẫn khái niệm thường gặp về bitcoin là ví bitcoin chứa bitcoin. Trên thực tế, ví chỉ chứa khoá. Các “coin” (tiền) được ghi lại vào blockchain trên mạng bitcoin. Người dùng kiểm soát tiền của mình trên mạng lưới bằng cách dùng khoá trong ví để ký các giao dịch. Ở một góc độ nào đó, có thể hiểu ví bitcoin là một chùm chìa khoá.

Có hai loại ví chính, được phân biệt bằng mối liên hệ giữa các khoá chứa trong ví.

Loại đầu tiên là ví bất định, trong đó mỗi khoá được tạo ra độc lập từ một số ngẫu nhiên. Các khoá này không liên quan đến nhau. Loại ví này còn được gọi là ví JBOK, viết tắt của cụm từ “Just a Bunch Of Keys” (chỉ là một đống khoá).

Loại thứ hai là ví tất định, trong đó tất cả các khoá đều được tạo ra từ một khoá chính duy nhất (master key), hay còn gọi là hạt giống (seed). Tất cả các khoá trong loại ví này đều liên quan đến nhau và có thể được tạo lại nếu có hạt giống gốc. Có một số cách tạo khoá khác nhau được dùng trong các ví tất định, trong đó cách phổ biến nhất là sử dụng một cấu trúc dạng cây gọi là ví tất định phân cấp (hierarchical deterministic), hay ví HD. Các ví tất định được khởi tạo từ một hạt giống. Để chúng dễ sử dụng hơn, các hạt giống được mã hoá thành từ tiếng Anh, còn được gọi là các từ mật mã trợ nhớ.

* + 1. **Chi tiết về công nghệ ví**

Bây giờ, chúng ta sẽ khám phá chi tiết từng chuẩn quan trọng của ngành đang được nhiều ví bitcoin áp dụng.

* + - 1. **Cụm từ mã trợ nhớ (BIP-39)**

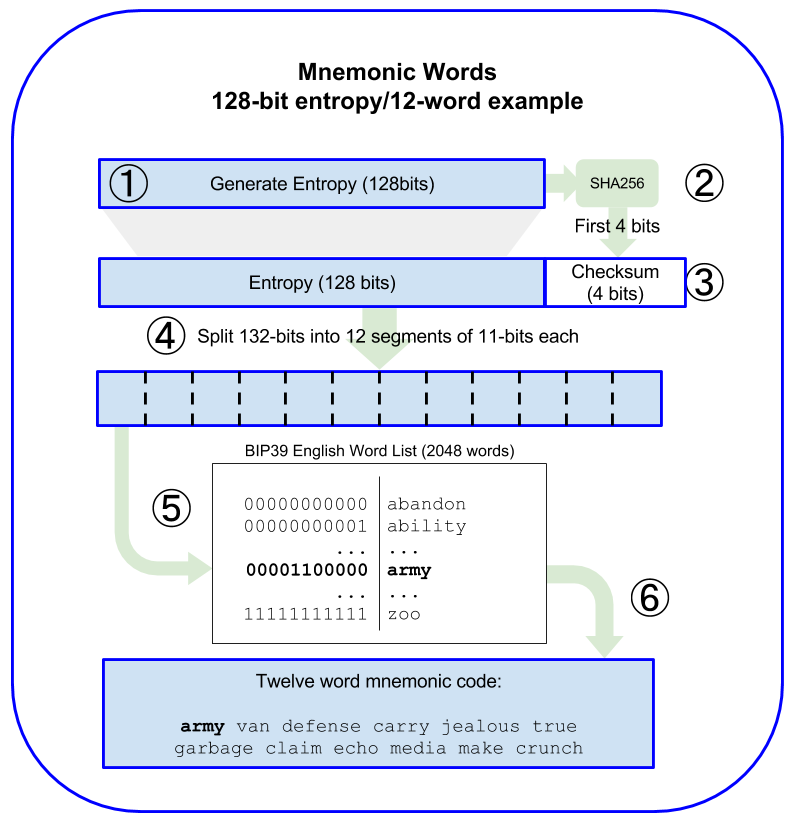
Cụm từ mã trợ nhớ là chuỗi từ tượng trưng (mã hoá) cho một số ngẫu nhiên được dùng làm hạt giống để tạo ví tất định. Chỉ cần chuỗi từ này là có thể tạo lại hạt giống đó, rồi từ hạt giống đó tạo lại ví với tất cả các khoá đã được tạo ra. Khi lần đầu tạo ví, ứng dụng ví cài đặt các ví tất định với cụm từ trợ nhớ sẽ hiển thị cho người dùng một chuỗi 12 đến 24 từ. Chuỗi từ này chính là bản sao lưu của ví và có thể dùng nó để khôi phục cũng như tạo lại tất cả các khoá trong cùng ứng dụng ví hoặc trong bất cứ ứng dụng ví tương thích nào. Các từ trợ nhớ giúp người dùng sao lưu ví dễ dàng hơn bởi chúng dễ đọc và dễ sao chép chính xác hơn so với một chuỗi ngẫu nhiên các con số. Các mã trợ nhớ được định nghĩa trong BIP-39 (<https://github.com/bitcoin/bips)>. Lưu ý, BIP-39 là một bản thực thi của một chuẩn mã trợ nhớ. Ngoài ra còn có một chuẩn khác trước BIP-39 với một tập hợp từ khác, được ví Electrum sử dụng. BIP-39 do công ty phát triển ví phần cứng Trezor đề xuất và không tương thích với bản thực thi của Electrum. Tuy nhiên, hiện nay BIP-39 đã nhận được sự ủng hộ rộng lớn trong ngành thông qua hàng chục bản thực thi tương thích lẫn nhau, do đó nên coi nó là chuẩn mặc định của ngành.

BIP-39 quy định quá trình tạo mã trợ nhớ và hạt giống (được miêu tả thành 9 bước ở đây). Để rõ ràng hơn, quá trình này được chia thành 2 phần: các bước từ 1 đến 6 được liệt kê trong mục “Tạo các từ trợ nhớ” và các bước từ 7 đến 9 được liệt kê trong mục “Từ cụm trợ nhớ đến hạt giống”.

**Tạo các từ trợ nhớ**

Ví tự động tạo ra các từ trợ nhớ thông qua quá trình chuẩn hoá được định nghĩa trong BIP-39. Từ điểm khởi đầu là một nguồn entropy, ví bổ sung một mã checksum, sau đó ánh xạ entropy này đến một danh sách các từ:

1. Tạo một chuỗi ngẫu nhiên (một entropy) 128 đến 256 bit.
2. Tạo một checksum của chuỗi ngẫu nhiênn trên bằng cách lấy các bit đầu tiên (độ dài entropy chia cho 32) của mã băm SHA256 của nó.
3. Thêm checksum này vào cuối chuỗi ngẫu nhiên.
4. Chia chuỗi này thành các đoạn 11 bit.
5. Ánh xạ từng giá trị 11 bit này thành 1 từ trong một từ điển cho trước gồm 2048 từ.
6. Mã trợ nhớ chính là chuỗi từ này.



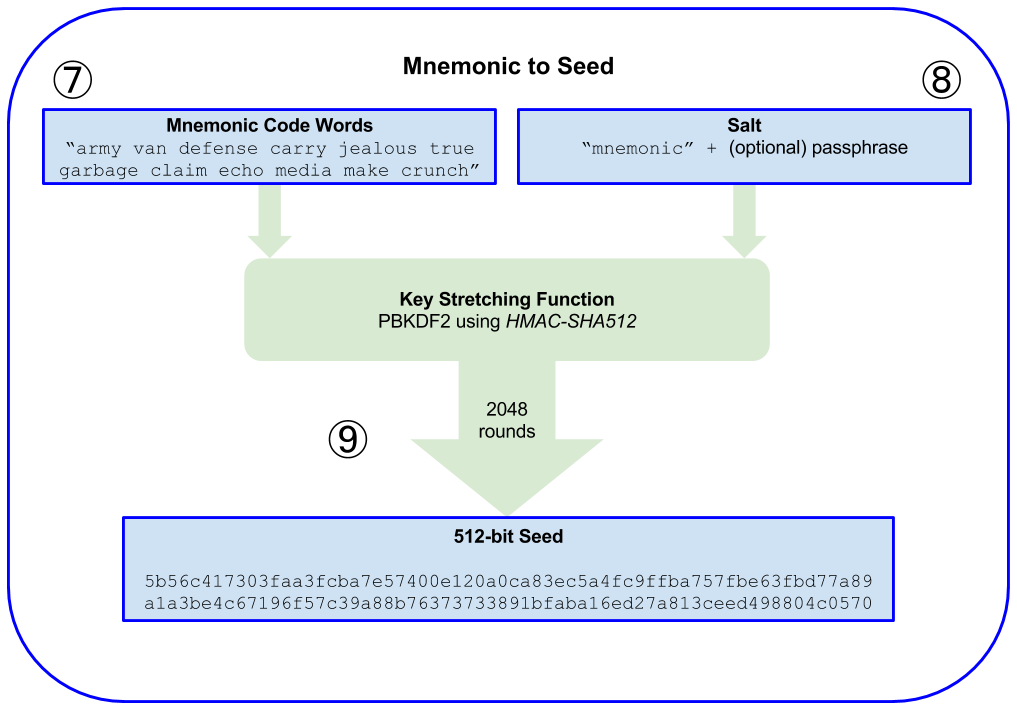
*Hình 1.3 Tạo entropy và mã hoá thành cụm từ trợ nhớ*

**Từ cụm trợ nhớ đến hạt giống**

Các từ trợ nhớ này đại diện cho entropy có độ dài từ 128 đến 256 bit. Sau đó, dùng entropy này để tạo một hạt giống dài hơn (512 bit) bằng hàm kéo dài khoá PBKDF2. Tiếp đến, dùng hạt giống tạo được để xây dựng một ví tất định rồi tạo các khoá của nó.

Hàm kéo dài nhận hai tham số: cụm từ trợ nhớ và một hạt muối (salt). Mục đích của hạt muối trong hàm khéo dài khoá là nhằm gây khó khăn cho việc xây dựng một bảng tìm kiếm để thực hiện một cuộc tấn công vét cạn. Trong chuẩn BIP-39, hạt muối còn có một mục đích khác – nó cho phép tạo ra một passphrase đóng vai trò một lớp bảo vệ tăng cường để bảo vệ hạt giống. Quá trình được mô tả trong các bước từ 7 đến 9 tiếp tục quá trình đã được mô tả trước đó ở phần “Tạo các từ trợ nhớ”.

1. Tham số đầu tiên cho hàm kéo dài khoá PBKDF2 là cụm trợ nhớ tạo ra ở bước 6.
2. Tham số thứ hai cho hàm kéo dài khoá PBKDF2 là một hạt muối. Hạt muối bao gồm một hằng số dạng chuỗi “mnemonic” được nối với một chuỗi passphrase tuỳ chọn do người dùng cung cấp.
3. PBKDF2 kéo dài các tham số cụm trợ nhớ và hạt muối bằng việc băm 2048 lần với thuật toán HMAC-SHA512, tạo ra kết quả cuối cùng là một giá trị 512 bit. Giá trị 512 bit này chính là hạt giống.



*Hình 1.4 Từ cụm trợ nhớ đến hạt giống*

# 1.2.2.2 Passphrase tuỳ chọn trong BIP-39

Chuẩn BIP-39 cho phép sử dụng một passphrase tuỳ chọn trong quá trình tạo hạt giống. Nếu không dùng passphrase, cụm trợ nhớ sẽ được kéo dài bằng một hạt muối chứa chuỗi không đổi “mnemonic”, tạo ra một hạt giống 512 bit từ bất kỳ cụm trợ nhớ nào cho trước. Nếu dùng passphrase, hàm kéo dài khoá sẽ tạo ra một hạt giống khác từ cùng một cụm trợ nhớ trên. Trên thực tế, với một cụm trợ nhớ cho trước, mỗi passphrase tiềm năng sẽ cho ra một hạt giống khác nhau. Về cơ bản, không có passphrase “sai”. Tất cả các passphrase đều hợp lệ và tạo ra các hạt giống khác nhau, từ đó hình thành nên một tập khổng lồ các ví chưa khởi tạo khả thi. Tập các ví khả thi này lớn (2^512) đến lỗi khả năng thực hiện một cuộc tấn công giải mã vét cạn hay vô tình đoán đúng ví đang được sử dụng là bất khả thi.

Passphrase tuỳ chọn này tạo ra 2 tính năng quan trọng:

* Một lớp bảo mật thứ hai (một thứ được nhớ) khiến cho một cụm trợ nhớ trở nên vô dụng nếu nó chỉ đi một mình. Lớp bảo mật này đóng vai trò bảo vệ các bản sao cụm trợ nhớ khỏi nguy cơ bị trộm.
* Một hình thức từ chối hợp lý hay “ví cưỡng ép”, trong đó sử dụng một passphrase dẫn đến một ví chứa ít tiền hơn nhằm đánh lạc hướng kẻ tấn công khỏi ví thật chứa phần lớn lượng tiền.

Tuy nhiên, cần phải lưu ý rằng việc dùng passphrase cũng mang lại rủi ro mất mát:

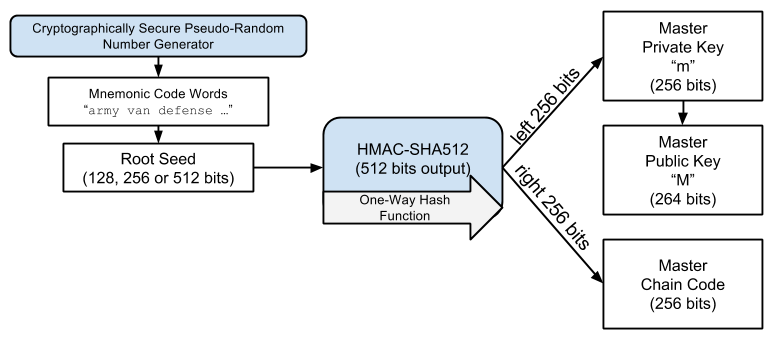
* Nếu chủ sở hữu ví bị mất khả năng (do thương tật, ốm đau…) hoặc qua đời và không ai biết passphrase, hạt giống sẽ trở nên vô dụng và toàn bộ tiền lưu trong ví đó sẽ bị mất vĩnh viễn.
* Ngược lại, nếu chủ sở hữu cất bản sao lưu passphrase ở cùng chỗ với hạt giống, nó sẽ vô hiệu hoá mục đích của lớp bảo vệ thứ hai.

Tuy passphrase rất hữu dụng, song chỉ nên dùng chúng kết hợp với một quy trình sao lưu và hồi phục được lên kế hoạch cẩn thận, trong đó tính đến cả khả năng chủ sở hữu qua đời và cho phép người thân trong gia đình có thể khôi phục lại tài sản tiền mã hoá.

* + - 1. **Tạo ví HD từ hạt giống**

Các ví HD được tạo ra từ một hạt giống gốc đơn lẻ là một số ngẫu nhiên 128, 256 hoặc 512 bit. Thường gặp nhất là các hạt giống được tạo ra từ một cụm trợ nhớ như đã trình bày chi tiết ở phần trước.

Mọi khoá trong ví HD đều được tạo ra từ hạt giống gốc này, vì vậy có thể tạo lại toàn bộ ví HD từ hạt giống đó trong bất kỳ ví HD tương thích nào. Vì thế có thể dễ dàng sao lưu, khôi phục và xuất nhập các ví HD vốn chứa hàng ngàn, thậm chí hàng triệu khoá với một thao tác đơn giản di chuyển cụm trợ nhớ đã tạo ra các hạt giống gốc.



*Hình 1.5 Tạo khoá chính và mã chuỗi từ một hạt giống gốc*

Hạt giống gốc là đầu vào cho thuật toán HMAC-SHA512 và mã băm kết quả được dùng để tạo ra một khoá bí mật chính (m) và một mã chuỗi chính (c).

Sau đó, khoá bí mật chính (m) tạo ra một khoá công khai chính tương ứng (M) thông qua phép nhân đường công elliptic m\*G.

Mã chuỗi (c) được dùng để tạo entropy trong hàm tạo khoá con từ khoá mẹ.

* + - 1. **Tạo khoá bí mật con**

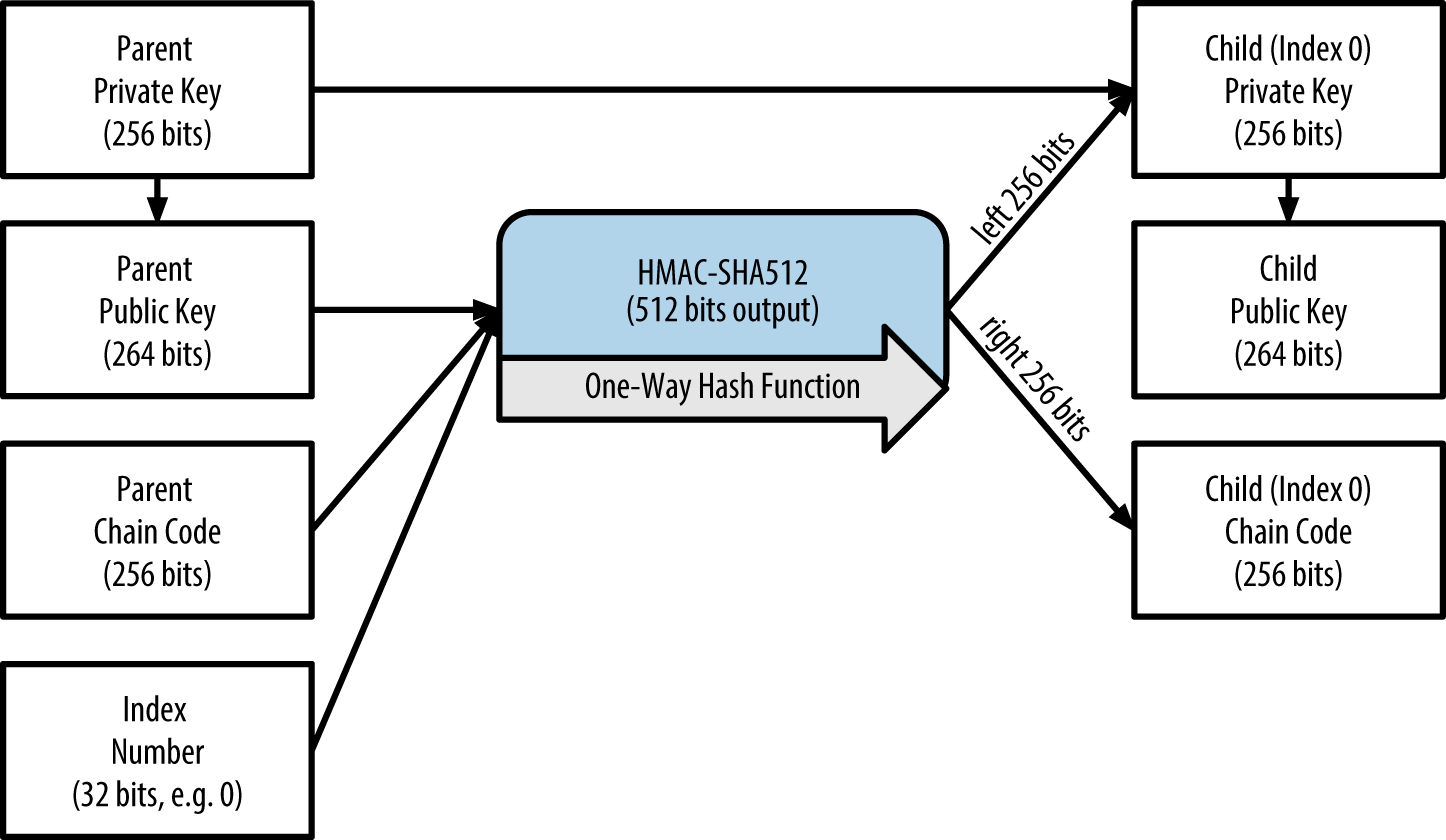
Ví HD sử dụng hàm tạo khoá con (child key derivation – CKD) để tạo ra các khoá con từ các khoá mẹ. Các hàm tạo khoá con dựa trên một hàm băm một chiều kết hợp với:

* Một khoá bí mật hoặc công khai mẹ (khoá ECDSA không nén)
* Một hạt giống gọi là một mã chuỗi (256 bit)
* Một số thứ tự (32 bit)

Mã chuỗi được dùng để đưa dữ liệu ngẫu nhiên tất định vào quá trình này, do đó nếu chỉ biết số thứ tự và một khoá con là chưa đủ để suy ra các khoá con khác. Việc biết một khoá con không làm cho việc tìm kiếm các khoá ngang hàng với nó trở nên khả thi, nếu như không biết mã chuỗi. Hạt giống mã chuỗi đầu tiên (tại gốc của cây) được tạo ra từ hạt giống, trong khi các mã chuỗi con về sau lại được tạo ra từ mã chuỗi mẹ.

Ba thành phần này (khoá mẹ, mã chuỗi và số thứ tự) được kết hợp lại với nhau rồi băm để tạo ra các khoá con như sau.

Kết hợp khoá công khai mẹ, mã chuỗi và số thứ tự với nhau rồi băm bằng thuật toán HMAC-SHA512 để tạo ra một mã băm 512 bit. Chia mã băm 512 bit này thành hai nửa 256 bit. Nửa 256 bit bên phải của mã băm trở thành mã chuỗi của khoá con. Nửa 256 bit bên trái được thêm vào khoá bí mật mẹ để tạo khoá bí mật con.



*Hình 1.6 Mở rộng khoá bí mật mẹ để tạo khoá bí mật con*

Việc thay đổi số thứ tự cho phép ta mở rộng khoá mẹ và tạo ra các khoá con khác thành chuỗi. Ví dụ, Con 0, Con 1, Con 2,… Mỗi khoá mẹ có thể có tới 2.147.483 647 (2^31) khoá con (2^31 là một nửa của toàn dải khả dụng 2^32).

Lặp lại quá trình này ở một tầng thấp hơn của cây, mỗi khoá con khi đó lại trở thành một khoá mẹ và tạo ra các con của nó, cứ thế tiếp diễn thành vô số thế hệ.

**Sử dụng các khoá con được tạo ra**

Khoá bí mật con và khoá bất định (ngẫu nhiên) không có gì khác nhau. Bởi hàm tạo khoá là hàm một chiều, nên không thể dùng khoá con để tìm khoá mẹ, cũng không thể dùng khoá con để tìm các anh em (khoá ngang hàng) của nó. Nếu có khoá con thứ n, bạn không thể tìm anh em của nó như khoá con thứ n-1 hay n+1, hay bất cứ khoá con nào trong chuỗi. Chỉ khoá mẹ và mã chuỗi mới có thể tạo ra tất cả các khoá con. Nếu không có mã chuỗi của khoá con, cũng không thể dùng khoá con để tạo ra bất kỳ khoá cháu nào. Bạn cần phải có cả khoá bí mật con và mã chuỗi của khoá con đó mới có thể bắt đầu một nhánh mới và tạo ra các khoá cháu.

Vậy chỉ mình khoá bí mật con thì có thể làm được gì? Có thể dùng nó để tạo ra một khoá công khai và một địa chỉ bitcoin. Sau đó có thể dùng nó để ký các giao dịch và chi tiêu các khoản tiền được gửi đến địa chỉ đó.

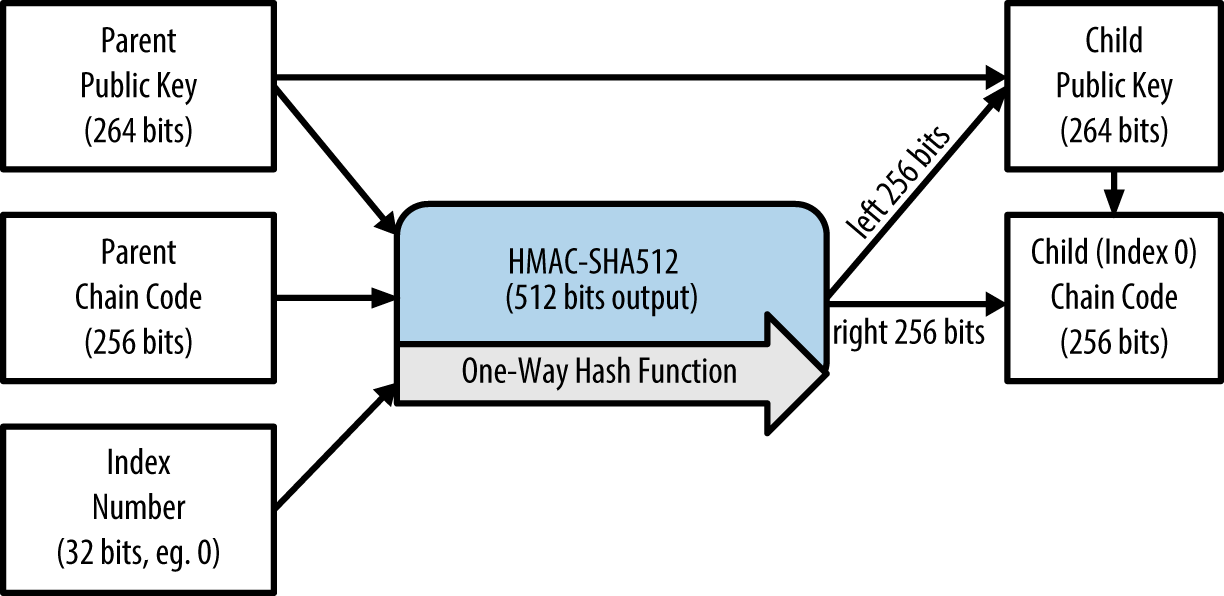
* + - 1. **Tạo khoá công khai con**

Như đã đề cập ở trên, một đặc tính rất hữu dụng của ví HD là khả năng tạo khoá công khai con từ khoá công khai mẹ mà không cần phải có khoá bí mật. Đặc tính này mang đến hai cách tạo khoá công khai con: hoặc là tạo từ khoá bí mật con, hoặc tạo trực tiếp từ khoá công khai mẹ.

Vì thế, có thể dùng khoá công khai mở rộng để tạo ra tất cả các khoá công khai (và chỉ khoá công khai mà thôi) trong nhánh đó của cấu trúc ví HD.

Có thể dùng lối tắt này để tạo ra các dạng triển khai chỉ dùng khoá công khai rất an toàn, trong đó một máy chủ hay ứng dụng có một bản sao của một khoá công khai mở rộng nhưng không hề có khoá bí mật nào. Dạng triển khai này có thể tạo ra vô số khoá công khai và địa chỉ bitcoin, nhưng không thể chi tiêu bất kì khoản tiền nào được gửi đến những địa chỉ ấy. Trong khi đó, trên một máy chủ khác an toàn hơn, khoá bí mật mở rộng có thể tạo ra tất cả các khoá bí mật tương ứng để ký các giao dịch và chi tiêu các khoản tiền đó.

Một ứng dụng phổ biến của giải pháp này là cài đặt khoá công khai mở rộng trên một máy chủ nền web đóng vai trò một ứng dụng thương mại điện tử. Máy chủ nền web này có thể dùng hàm tạo khoá công khai để tạo ra một địa chỉ bitcoin mới cho mỗi giao dịch (ví dụ cho một giỏ hàng của khách). Máy chủ nền web này không có bất kỳ khoá bí mật nào nên không sợ trộm cắp. Nếu không có ví HD, cách duy nhất để thực hiện điều này là tạo sẵn hàng nghìn địa chỉ bitcoin trên một máy chủ an toàn riêng biệt và nạp sẵn chúng vào máy chủ thương mại điện tử. Đây là một phương án rất rắc rối và đòi hỏi phải bảo trì thường xuyên để đảm bảo rằng máy chủ thương mại điện tử không bị “hết” khoá.



*Hình 1.7 Mở rộng khoá công khai mẹ để tạo ra khoá công khai con*

Một ứng dụng phổ biến khác của giải pháp này là lưu trữ lạnh và các ví phần cứng. Trong trường hợp này, có thể lưu khoá bí mật mở rộng vào ví giấy hay thiết bị phần cứng (thí dụ như ví phần cứng Trezor), trong khi có thể lưu khoá công khai mở rộng trên mạng trực tuyến. Người dùng có thể tạo ra các địa chỉ “nhận” tuỳ ý, trong khi các khoá bí mật vẫn được cất giữ an toàn ngoại tuyến. Để chi tiêu tiền, người dùng có thể sử dụng khoá bí mật mở rộng trên một phần mềm ký giao dịch bitcoin ngoại tuyến hoặc ký giao dịch trên một thiết bị vi phần cứng (ví dụ như Trezor).

* 1. **Bài toán**

Bitcoin, không giống như hầu hết các tiền tệ truyền thống. Vì vậy, cách tiếp cận loại tiền tệ này là hoàn toàn khác biệt, đặc biệt là cách mua và lưu trữ. Vì bitcoin không tồn tại trong bất kỳ hình dạng vật lý nào nên nó không thể được lưu trữ ở bất kỳ nơi nào. Thay vì đó, đồng tiền này sử dụng private key để kết nối với địa chỉ bitcoin và ký nhận giao dịch. Khi thực hiện một giao dịch bitcoin, bạn cần phải sử dụng cả public key của người nhận và private key của bạn. Có nhiều loại ví bitcoin, cung cấp nhiều lựa chọn về tính bảo mật, tiện lợi cho người dùng như ví giấy, ví web, ví di động… Ví giấy là một dạng tài liệu văn bản có chứa địa chỉ public dùng để nhận bitcoin là một private key, cho phép sử dụng hoặc chuyển bitcoin tại một địa chỉ. Những key này được in dưới dạng các mã QR để bạn dễ dàng sử dụng và thêm vào phần mềm khi cần tiến hành giao dịch. Lợi ích của ví giấy là các key không được lưu trữ theo hình thức kỹ thuật số, làm cho người dùng không thể bị hacker tấn công hay malware xâm nhập. Tuy nhiên ví giấy cũng có nhược điểm: mực in trên ví có thể phai mờ và chất liệu giấy mỏng khiến việc bảo quản cực kỳ khó. Bên cạnh đó, nếu như bạn làm mất ví giấy, bạn sẽ không thể truy cập và khôi phục ví được. Do vậy hầu hết người dùng ưu tiên sử dụng ví web. Ví web lưu trữ các private key của bạn trên máy chủ của một công ty thứ ba cung cấp dịch vụ. Máy chủ sẽ liên tục trực tuyến và được một chủ thể kiểm soát. Ví web cho phép bạn sử dụng tiền điện tử thuận tiện hơn từ bất kỳ thiết bị nào có kết nối internet. Trừ khi được vận hành đúng, các tổ chức cung cấp dịch vụ sẽ nắm giữ toàn bộ thông tin private key của bạn và chắc chắn sẽ nắm quyền kiểm soát quỹ của bạn nếu có sự cố xảy ra. Ngoài ra, một số ví web được vận hành trên nền tảng của các sàn giao dịch. Còn đối với những người thường xuyên sử dụng bitcoin thường ngày, chi trả hàng hoá trong cửa hàng hoặc giao dịch trực tiếp mặt đối mặt, ví bitcoin trền điện thoại di động là một công cụ hoàn hảo. Sử dụng một ứng dụng trên điện thoại thông minh của bạn, loại ví này lưu trữ private key và cho phép bạn thanh toán trực tiếp từ điện thoại của mình. Chính bởi tính di động, tiện lợi này mà ví di động được ưu tiên sử dụng nhiều nhất trong các loại ví bitcoin. Và bài toán đặt ra ở đây là làm thế nào để xây dựng được một ứng dụng ví Bitcoin theo những thông tin có được ở trên.

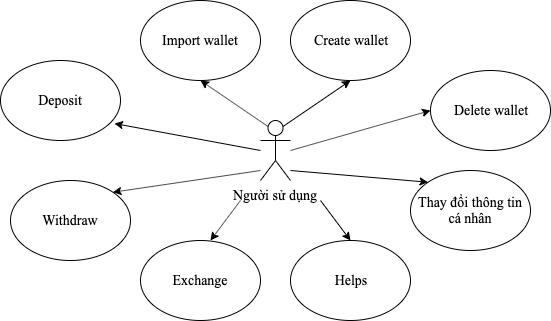
* 1. **Nội dung đề tài**
* **Tên đề tài:** Xây dựng ứng dụng Bitcoin Wallet trên nền tảng Android và IOS.
* **Lý thuyết:** Tìm hiểu về các đồng tiền ảo (BTC, ETH…), ví bitcoin, ngôn ngữ Javascript và React-native framework
* **Thực hành:** Phân tích thiết kế hệ thống theo hướng đối tượng cơ bản và xây dựng các chức năng của ứng dụng, cụ thể như sau:
* Create new wallet: tạo mới ví với các thông số cần thiết như tên ví, mật khẩu
* Import wallet: thêm ví bằng private key hoặc chuỗi mnemonic
* Delete wallet: xoá ví
* Exchange wallet: đổi coin
* Deposit: nạp coin
* Withdraw: gửi coin
* Settings: cài đặt với các chứng năng như rename wallet (đổi tên ví), change wallet password (đổi mật khẩu), backup wallet (sao lưu ví), passcode/touchId, change lanuage (đổi ngôn ngữ ứng dụng)…
  1. **Kết luận chương 1**

Chương 1 đã nêu được các thông tin cơ bản về một số coin nổi tiếng, ví bitcoin và các thông tin liên quan. Từ đó, chúng ta có thể hiểu rõ hơn về mục đích xây dựng ứng dụng ví bitcoin và kiến thức nền để phân tích chi tiết hơn về cách hoạt động trong chương 2.

# CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

# 2.1 Biểu đồ Usecase

# 2.1.1 Biểu đồ Usecase tổng quát



Hình 2.1 Usecase tổng quát

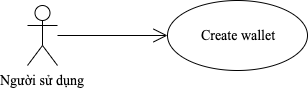
Qua biểu đồ Usecase tổng quát, ta có thể nhìn thấy được khái quát các chức năng chính của hệ thống như:

* create wallet: tạo ví
* delete wallet: xoá ví,
* import wallet: thêm ví,
* deposit: nạp coin vào ví,
* withdraw: gửi coin đi
* exchange: đổi coin,
* helps: trợ giúp,
* thay đổi thông tin cá nhân

dưới sự tác động của tác nhân tham gia hệ thống là người sử dụng.

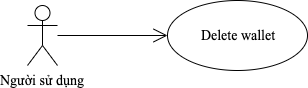
# 2.1.2 Biểu đồ phân rã Usecase

# Chức năng “Create wallet”



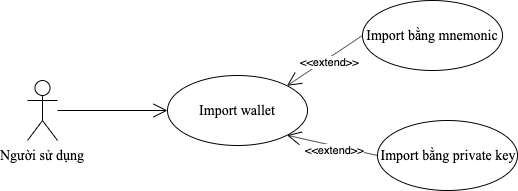
Hình 2.2 Biểu đồ phân rã Usecase create wallet

# Chức năng “Delete wallet”



Hình 2.3 Biểu đồ phân rã Usecase delete wallet

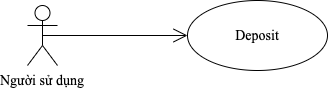
# Chức năng “Import wallet”



Hình 2.4 Biểu đồ phân rã Usecase import wallet

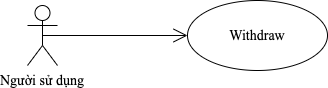
# 

# Chức năng “Deposit”



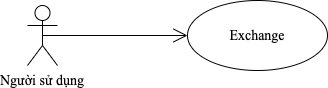
Hình 2.5 Biểu đồ phân rã Usecase deposit

# Chức năng “Withdraw”



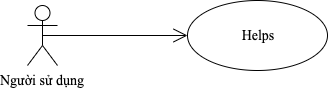
Hình 2.6 Biểu đồ phân rã Usecase withdraw

# Chức năng “Exchange”



Hình 2.7 Biểu đồ phân rã Usecase exchange

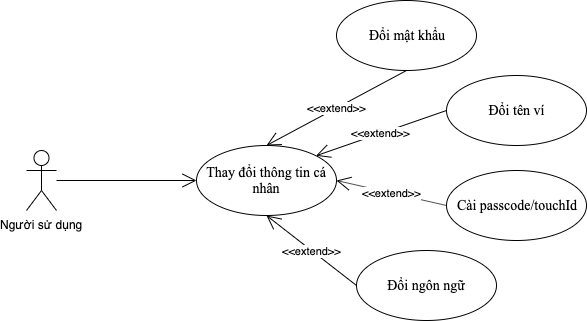
# Chức năng “Helps”



Hình 2.8 Biểu đồ phân rã Usecase helps

# 

# Chức năng “Thay đổi thông tin cá nhân”



Hình 2.9 Biểu đồ phân rã Usecase withdraw

# 2.1.3 Các tác nhân chính tham gia hệ thống

Ví Bitcoin có một tác nhân chính: Người sử dụng.

Dựa vào tác nhân đã xác định, bảng mô tả Usecase của hệ thống được xây dựng như sau:

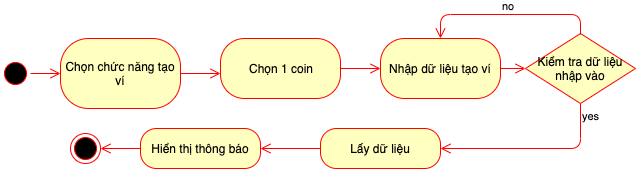
Bảng 2.1 Mô tả Usecase của hệ thống

|  |  |  |
| --- | --- | --- |
| **Tác nhân** | **Ca sử dụng** | **Mô tả** |
| Người sử dụng | Create wallet | Tạo ví với tên và mật khẩu, mật khẩu ví sẽ dùng để sinh ra Private Key |
| Delete wallet | Người dùng phải backup thông tin ví trước khi xóa ví hoàn toàn. |
| Import wallet | Thêm ví bằng Private Key hoặc chuỗi Mnemonic |
| Withdraw | Người sử dụng gửi một lượng coin đến một địa chỉ ví khác |
| Deposit | Người sử dụng nhận coin về ví thông qua địa chỉ của ví coin |
| Exchange | Là chức năng đổi tiền tại ví từ đồng coin này sang đồng coin khác với mức phí quy định. |
| Helps | Người dùng có thể gửi thắc mắc hoặc ý kiến đóng góp cho ví |
| Thay đổi thông tin cá nhân | Thay đổi các thông tin về tên ví, mật khẩu, passcode, ngôn ngữ |

* 1. **Đặc tả và hiện thực hóa Usecase**

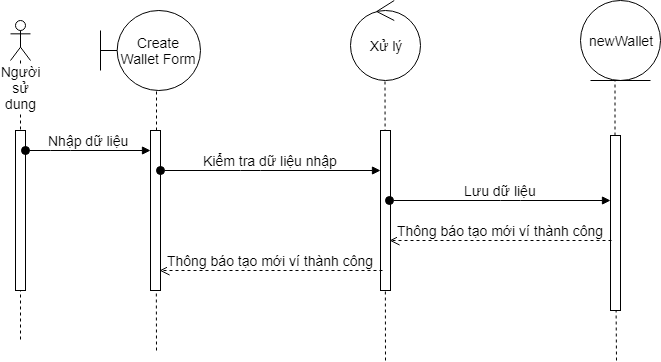
**Usecase “Create wallet”**

* Tác nhân chính: người sử dụng
* Mô tả: Usecase cho phép người sử dụng tạo ví
* Điều kiện: không có
* Luồng sự kiện
  + - 1. Người dùng chọn chức năng create wallet trên app
      2. App hiển thị màn hình danh sách coin
      3. Người dùng chọn một coin
      4. App hiển thị màn hình create wallet
      5. Người dùng nhập dữ liệu yêu cầu như wallet name, password, confirm password và nhấn nút create wallet
      6. App tiến hành nhận và kiểm tra dữ liệu hợp lệ. Nếu người dùng nhập sai thông tin, app sẽ thông báo yêu cầu nhập lại cho tới khi người dùng nhập đúng thông tin. Nếu việc tạo ví thành công thì app hiển thị thông báo thành công cho người dùng và chuyển sang màn hình chính. Ngược lại, nếu việc tạo ví thất bại, app sẽ thông báo tạo ví thất bại cho người dùng, app vẫn ở lại màn hình tạo ví.

****

Hình 2.10 Biểu đồ hoạt động create wallet

.



Hình 2.11 Biểu đồ trình tự create wallet

**Usecase “Import wallet”**

* Tác nhân chính: người sử dụng
* Mô tả: Usecase cho phép người sử dụng thêm ví
* Điều kiện: không có
* Luồng sự kiện:

Người dùng chọn chức năng import wallet trên app

App hiển thị màn hình danh sách coin

Người dùng chọn một coin

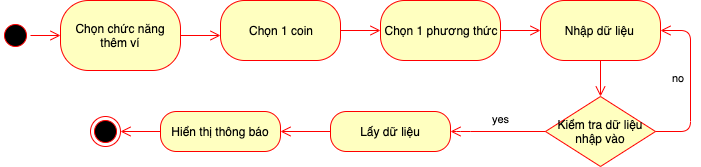
App hiển thị màn hình danh sách phương thức thêm ví

Người dùng chọn một phương thức

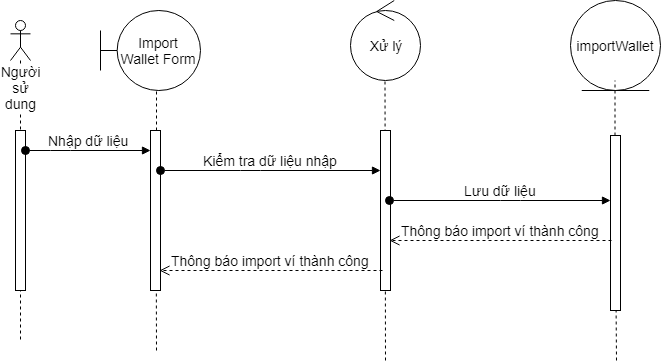
App hiển thị màn hình thêm ví

Người dùng nhập dữ liệu yêu cầu và nhấn nút import wallet

App nhận và kiểm tra dữ liệu hợp lệ. Nếu người dùng nhập sai thông tin, app sẽ thông báo yêu cầu nhập lại cho tới khi người dùng nhập đúng thông tin. Nếu việc thêm ví thành công thì app hiển thị thông báo thành công cho người dùng và chuyển sang màn hình chính. Ngược lại, nếu việc thêm ví thất bại, app sẽ thông báo thêm ví thất bại cho người dùng, app vẫn ở lại màn hình thêm ví.



Hình 2.12 Biểu đồ hoạt động import wallet



Hình 2.13 Biểu đồ trình tự import wallet

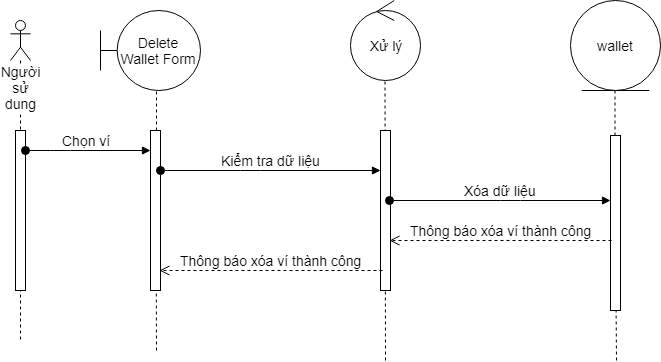
**Usecase “Delete wallet”**

* Tác nhân chính: người sử dụng
* Mô tả: Usecase cho phép người sử dụng xoá ví
* Điều kiện: Người dùng phải backup dữ liệu trước đó
* Luồng sự kiện:

1. Người dùng chọn chức năng xoá ví
2. App tiến hành xoá ví. Nếu việc xoá ví thành công thì app sẽ hiển thị thông báo xoá thành công cho người dùng. Ngược lại, xoá ví thất bại, app sẽ thông báo xoá ví thất bại.

****

Hình 2.14 Biểu đồ hoạt động delete wallet

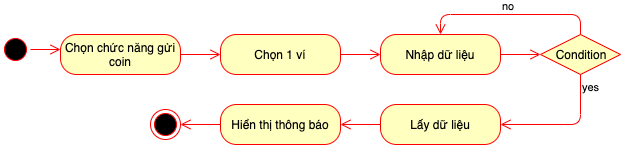


Hình 2.15 Biểu đồ trình tự delete wallet

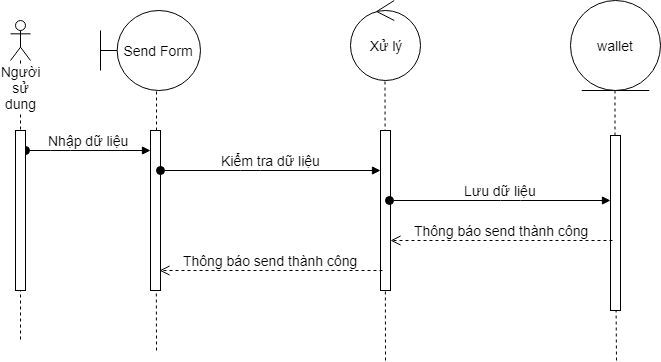
**Usecase “Withdraw”**

* Tác nhân chính: người sử dụng
* Mô tả: Usecase cho phép người sử dụng gửi coin
* Điều kiện: số dư ví lớn hơn 0
* Luồng sự kiện

1. Người sử dụng chọn chức năng withdraw
2. App hiển thị màn hình withdraw
3. Người dùng chọn ví muốn withdraw
4. Người dùng nhập dữ liệu yêu cầu như địa chỉ nhận, số lượng coin gửi đến địa chỉ tương ứng và nhấn nút “send”
5. App nhận và kiểm tra dữ liệu hợp lệ. Nếu gửi coin thành công, app sẽ hiển thị thông báo gửi thành công cho người dùng. Ngược lại, nếu người dùng nhập sai thông tin yêu cầu, app sẽ thông báo yêu cầu người dùng nhập lại đến khi nhập đúng thông tin yêu cầu. Nếu việc gửi coin thất bại, app cũng sẽ thông báo lỗi đến người dùng.

****

Hình 2.16 Biểu đồ hoạt động withdraw



Hình 2.17 Biểu đồ trình tự withdraw

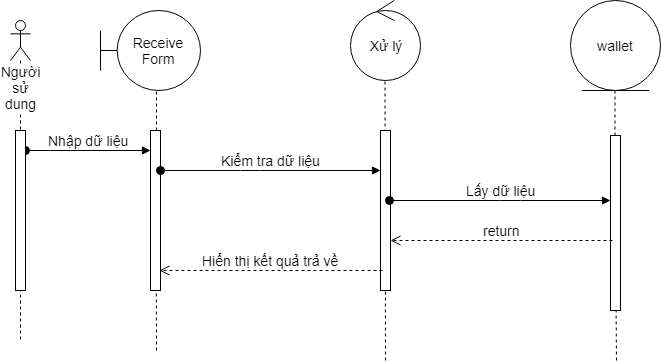
**Usecase “Deposit”**

* Tác nhân chính: người sử dụng
* Mô tả: Usecase cho phép người sử dụng nạp coin
* Điều kiện: sau khi tạo ví hoặc import ví thành công
* Luồng sự kiện:

1. Người dùng chọn chức năng deposit
2. App hiển thị màn hình deposit
3. Người dùng chọn ví cần nạp coin
4. App hiển thị địa chỉ ví, mã QR của địa chỉ ví
5. Người dùng nhấn nút coppy address để sao chép địa chỉ ví
6. App hiển thị thông báo sao chép thành công hoặc thất bại

****

Hình 2.18 Biểu đồ hoạt động deposit

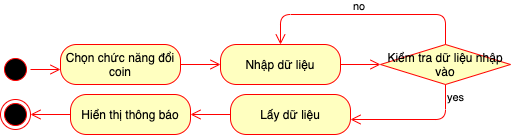


Hình 2.19 Biểu đồ trình tự deposit

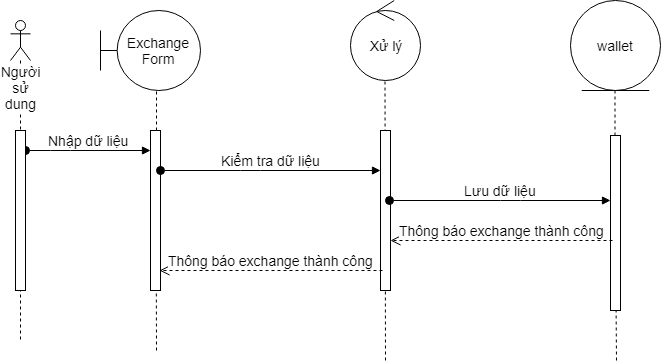
**Usecase “Exchange”**

* Tác nhân chính: người sử dụng
* Mô tả: Usecase cho phép người sử dụng đổi coin
* Điều kiện: số dư của coin đổi lớn hơn 0
* Luồng sự kiện

1. Người dùng chọn chức năng exchange
2. App hiển thị màn hình exchange
3. Người dùng nhập dữ liệu yêu cầu như coin đổi, số lượng
4. App nhận và kiểm tra dữ liệu. Nếu dữ liệu nhập vào không hợp lệ thì app sẽ thông báo lỗi cho người dùng và yêu cầu nhập lại đến khi nhập đúng. Nếu việc đổi coin thành công, app sẽ thông báo thành công cho người dùng. Ngược lại, việc đổi coin thất bại, app cũng thông báo thất bại tới người dùng.

****

Hình 2.20 Biểu đồ hoạt động exchange

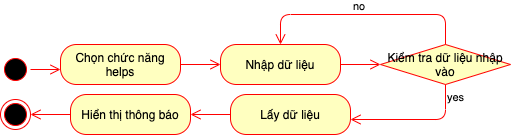


Hình 2.21 Biểu đồ trình tự exchange

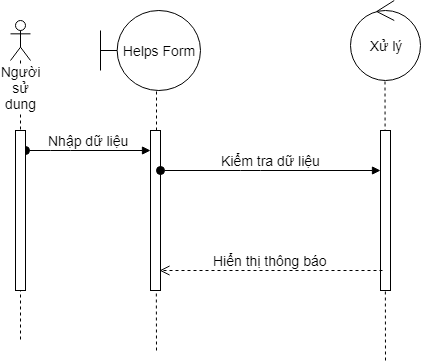
**Usecase “Helps”**

* Tác nhân chính: người sử dụng
* Mô tả: Usecase cho phép người sử dụng yêu cầu trợ giúp
* Điều kiện: không có
* Luồng sự kiện

1. Người dùng chọn chức năng trợ giúp
2. App hiển thị màn hình trợ giúp
3. Người dùng nhập dữ liệu yêu cầu như tiêu đề, địa chỉ email, nội dung mail và nhấn nút “send”
4. App nhận và kiểm tra dữ liệu hợp lệ. Nếu gửi mail trợ giúp thành công thì app sẽ hiển thị thông báo cho người dùng. Ngược lại nếu có lỗi xảy ra, app cũng sẽ thông báo lỗi cho người dùng.

****

Hình 2.22 Biểu đồ hoạt động helps

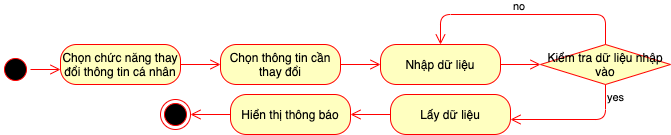


Hình 2.23 Biểu đồ trình tự helps

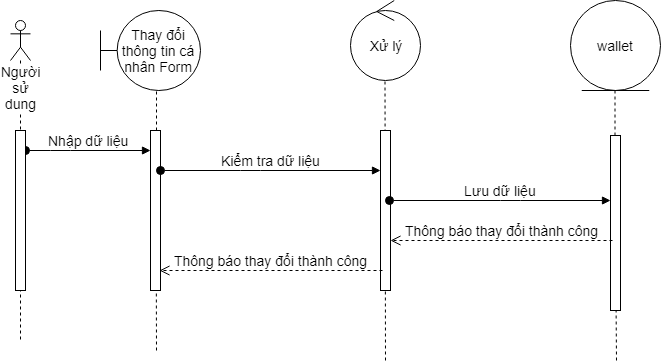
**Usecase “Thay đổi thông tin cá nhân”**

* Tác nhân chính: người sử dụng
* Mô tả: Usecase cho phép người sử dụng thay đổi thông tin cá nhân
* Điều kiện: không có
* Luồng sự kiện

1. Người dùng chọn chức năng thay đổi thông tin cá nhân
2. App hiển thị màn hình thay đổi thông tin cá nhân
3. Người dùng chọn thông tin cần thay đổi như tên ví, mật khẩu…
4. Người dùng nhập dữ liệu mới
5. App nhận, kiểm tra dữ liệu hợp lệ và hiển thị thông báo cho người dùng nếu thành công hoặc thất bại.

****

Hình 2.24 Biểu đồ hoạt động Thay đổi thông tin cá nhân



Hình 2.25 Biểu đồ trình tự Thay đổi thông tin cá nhân

* 1. **Kết luận chương 2**

Chương 2 đã phân tích rõ yêu cầu và chức năng của ứng dụng ví bitcoin. Đồng thời đưa ra các biểu đồ thể hiện cụ thể từng chức năng sẽ hoạt động ra sao để người dùng nắm rõ.

# CHƯƠNG 3: XÂY DỰNG ỨNG DỤNG

# 3.1. Công nghệ, công cụ và môi trường phát triển

Ứng dụng Bitcoin Wallet lựa chọn xây dựng trên nền tảng React-native, ngôn ngữ lập trình JavaScript cho cả hai hệ điều hành Android và IOS, công cụ sử dụng Visual Studio Code, Android Studio, Xcode, môi trường phát triển di động.

**3.1.1 React-native Framework**

React native là framework xây dựng ứng dụng di động native sử dụng Javascript được tạo bởi Facebook. React native cho phép các nhà phát triển sử dụng Javascript để xây dựng ứng dụng IOS và Android.

Để hiểu rõ về React native là gì, chúng ta cần phân biệt sự khác nhau giữa ứng dụng Native và Hybrid.

**Ứng dụng Hybrid**

Hybrid App là sự kết hợp giữa ứng dụng Web và ứng dụng mobile. Là chương trình phần mềm trên điện thoại di động được viết dựa trên nền tảng web (html5, css3, javascript), bản chất hoàn toàn là ứng dụng web nhưng có thêm được các tính năng thao tác phần hệ điều hành như tập tin, truy cập máy ảnh, GPS hoặc các cảm biến như con quay hồi chuyển, gia tốc kế… Nhược điểm của Hybrid App đó chính là vấn đề hiệu năng sẽ bị ảnh hưởng đáng kể cũng như không tương tác được hết những tài nguyên hệ thống phần mềm cũng như phần cứng. Vì vậy, mà Native App là sự lựa chọn hàng đầu của giới lập trình trong việc phát triển phần mềm di động bởi vì nó mang lại hiệu năng cao, trải nghiệm người dùng tốt.

**Ứng dụng Native**

Native App là các ứng dụng được phát triển và build trên chính những công cụ hỗ trợ mà nhà phát triển iOS và Android cung cấp cho lập trình viên ở đây là Xcode và Android. Ví dụ với iOS là Objective-C, Swift và Android là Java, Window phone là C++ hoặc C#. Các ứng dụng Native được build dựa trên ngôn ngữ của chính hệ điều hành. Và sử dụng những tính năng sẵn có của hệ điều hành mà không phải thông qua bất cứ ứng dụng bên thứ ba. Hay engine nào khác để vận hành nên tốc độ là nhanh nhất. Tốc độ ở đây bao gồm tốc độ hiển thị (độ mượt) và tốc độ xử lý. Nên để ý kỹ bạn sẽ thất các thao tác của ứng dụng Native như chuyển trang, điều hướng, cuộn trang đều rất mượt. Với việc hơn hẳn về tốc độ xử lý nên ứng dụng Native có thể làm được ứng dụng nặng đồ hoạ như game. Còn hybrid vẫn có thể làm được nhưng không mượt mà. Hầu như hiếm ai dùng hybrid cho lập trình game cho điện thoại di động. Ngoại trừ một số game đơn giản như xếp hình, giải đố, 2048…)

**Cách hoạt động của React native**

Ứng dụng viết bằng React native được chia làm hai phần: phần hiển thị (view) và phần xử lý.

* Phần hiển thị được biên dịch từ javascript sẽ map với những component của hệ thống. Ví dụ như điều hướng, tab, touch… Phần view này được lấy cảm hứng từ Virtual DOM của React JS. Mọi xử lý view được thực hiện trên một cây DOM ảo, sau đó được React native render lại bằng native view.
* Phần xử lý vẫn được thực hiện trực tiếp bằng ngôn ngữ javascript. Ví dụ “1+1=2” biểu thức này được xử lý dưới bộ core thực thi javascript. Không phải thông dịch qua Java hay Swift/Objective-C rồi mới làm phép tính.

**Ưu và nhược điểm của React native**

Sẽ không có cái gì gọi là hoàn hảo cả React native cũng vậy. React native cũng giống như các Framework khác cũng có những ưu nhược điểm riêng

* **Ưu điểm của react native**
  + Các thao tác của ứng dụng native như chuyển trang, điều hướng, cuộn trang đều rất mượt.
  + Hiệu quả về mặt thời gian khi mà bạn muốn phát triển một ứng dụng nhanh chóng.
  + Hiệu năng tương đối ổn định.
  + Cộng đồng phát triển mạnh.
  + Tiết kiệm tiền.
  + Ứng dụng tin cậy và ổn định
  + Xây dựng cho nhiều hệ điều hành khác nhau với ít native code nhất.
  + Trải nghiệm người dùng tốt hơn là hybrid app.
  + Giúp cho lập trình viên web có thể viết ứng dụng native để khắc phục các điểm yếu của ứng dụng web và hybrid.
* **Nhược điểm của React native**

React native là một giải pháp tuyệt vời cho phát triển ứng dụng trên điện thoại di động. Tuy nhiên đến thời điểm hiện tại, vẫn còn tồn tại một số khuyết điểm:

* Vẫn còn thiếu các component view cho Android: Map, Modal, Slider, Spinner hoặc các module như Camera Roll, Media, PushNotificationIOS.
* Không hỗ trợ Window Phone: đây sẽ là điểm trừ lớn nếu so với ứng dụng hybrid. Tuy nhiên, nếu bạn chỉ tập trung cho iOS và Android thì tất nhiên đây không phải vấn đề lớn.
* Không build được ứng dụng iOS trên Window và Linux: tất nhiên đây không phải điểm yếu kém của bản thân React native. Mà đó là do yêu cầu từ Apple.
* React native không thể build được ứng dụng quá phức tạp. Không nên dùng để viết game có tính đồ hoạ và cách chơi phức tạp.

**3.1.2 Ngôn ngữ lập trình Javascript**

Javascript chính là một ngôn ngữ lập trình với khả năng đem tới sự sinh động khi thiết kế website. Đây là dạng ngôn ngữ theo kịch bản, dựa trên chính đối tượng phát triển có sẵn hoặc tự định nghĩa ra. Chính vì tính tiện lợi, hiệu quả ứng dụng cao mà ngôn ngữ lập trình này được tin dùng ngày càng nhiều, ứng dụng rộng rãi trong các website hiệu quả. Việc sử dụng ngôn ngữ Javascript có thể ứng dụng cho mọi trình duyệt khác nhau, hiện được sử dụng phổ biến như Chrome, hay Firefox… Hơn nữa, đây còn là ngôn ngữ lập trình hoạt động hiệu quả, được hỗ trợ đầy đủ các trình duyệt của thiết bị di động. Bởi thế mà việc sử dụng đa dạng, có thể đáp ứng tốt cho nhiều nhu cầu, những đòi hỏi khác nhau của người dùng.

Nhìn chung, ngôn ngữ lập trình Javascript khi sử dụng được đánh giá dễ học, được phát triển bởi Netscape và sử dụng trong hơn 92% các website ngày nay. Việc sử dụng Javascript có thể gắn trong một element trong website, hoặc có thể trong một sử kiện của trang web tiện lợi và hiệu quả. Chính việc có thể hoạt động nhanh, nhẹ và mượt mà… nên ngôn ngữ lập trình này càng được tin tưởng và ứng dụng nhiều hơn cho nhu cầu của người dùng. Chúng ta có thể thêm Javascript trực tiếp với HTML, cũng có thể lưu lại trên một files hoàn toàn riêng biệt và sử dụng khi cần thiết.

**Những ưu điểm làm nên chất lượng của ngôn ngữ Javascript**

* Sử dụng Javascript được thực hiện từ phía Client

Điều này là các mã được tiến hành thực hiện thông qua bộ vi xử lý của chính người dùng, thay vì sử dụng trên máy chủ web. Chính đặc điểm này giúp việc tiết kiệm năng thông, đồng thời giảm những căng thẳng, hoạt động quá nhiều trên máy chủ web dễ dàng hơn.

* Là ngôn ngữ lập trình dễ dàng sử dụng
* Đánh giá cao ở độ nhanh chóng với người dùng cuối
* Mang tới các tính năng bổ sung cho các website
* Sử dụng đa dạng trên nhiều trình duyệt
* Dễ dàng khi sử dụng với ngôn ngữ Javascript
* Sở hữu giao diện thân thiện, nhiều tính năng

**Hạn chế còn tồn tại của ngôn ngữ Javascript**

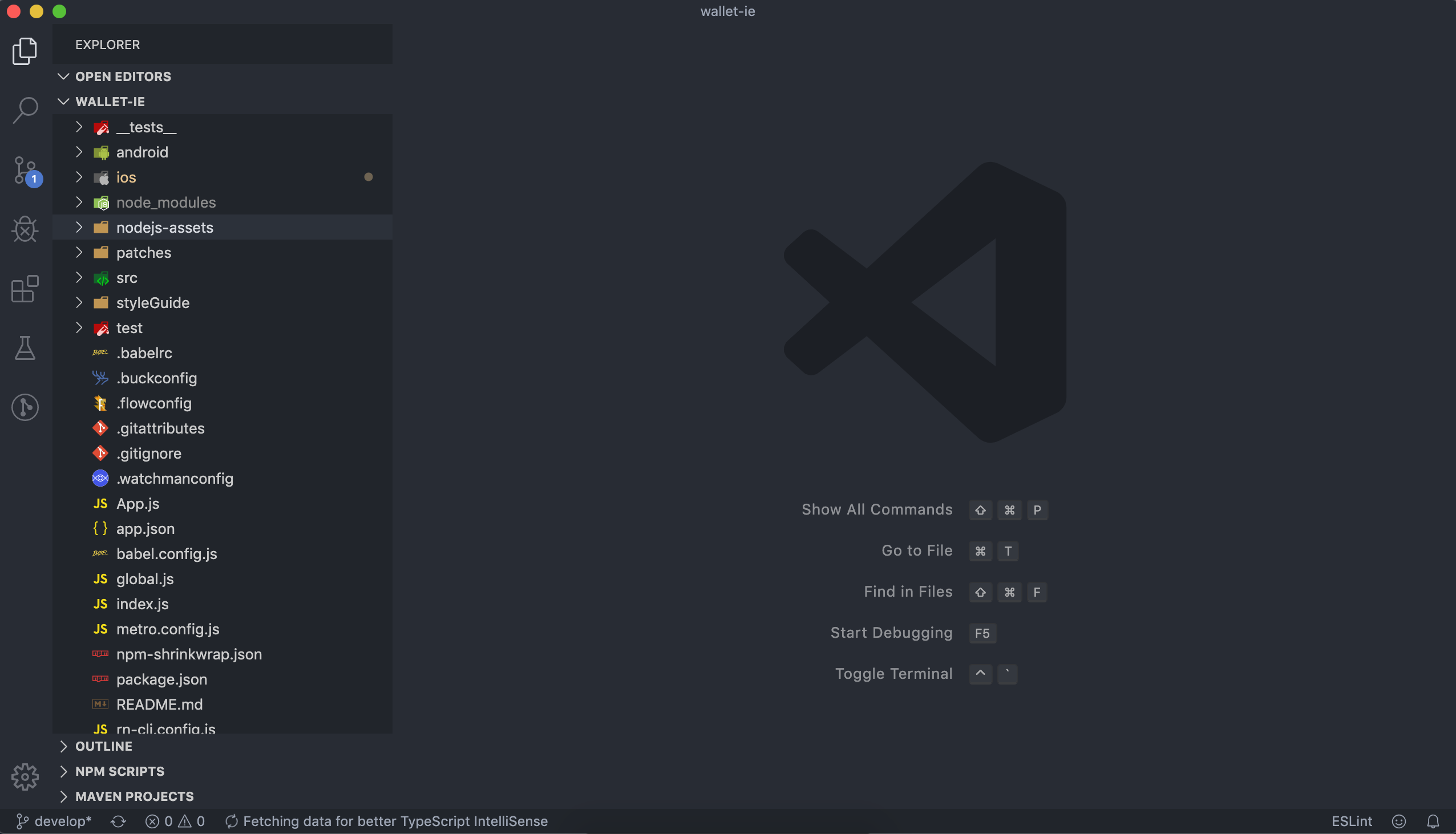
Tồn tại song song với những ưu điểm là hạn chế mà ngôn ngữ lập trình Javascript đang phải đối mặt. Một hạn chế của ngôn ngữ Javascript chính là việc chúng ta có thể sử dụng để thực thi mã độc trên máy tính của người dùng. Tình trạng này vô tình gây ra những thiệt hại, ảnh hưởng không hề nhỏ tới công việc mà chúng ta đang thực hiện. Trong một số trường hợp, trong một vài trình duyệt cụ thể việc sử dụng ngôn ngữ lập trình Javascript không thể thực hiện được. Nó gây ảnh hưởng tới việc ứng dụng, đáp ứng cho nhu cầu của từng người. Sử dụng ngôn ngữ lập trình này có một hạn chế khác khi có Javascript code snippets khá lớn. Khi có khả năng triển khai khác nhau tuỳ từng thiết bị, nó vừa là ưu điểm song cũng tồn tại những hạn chế nhất định. Chính điều này khiến nguy cơ không đồng nhất có thể xảy ra, gây ảnh hưởng tới quá trình thực hiện mục tiêu cuối cùng.

**3.1.3 Công cụ**

**Visual Studio Code**

Visual Studio Code là sản phẩm của Microsoft, ra mắt vào tháng 4 năm 2015 ở hội nghị Build. Đặc điểm nổi bật là đơn giản, gọn nhẹ, dễ dàng cài đặt. Visual Studio Code có thể cài đặt trên cả Windows, Linux và Mac OS và hỗ trợ nhiều ngôn ngữ. Nó hỗ trợ chức năng debug, đi kèm với Git, có syntax highlighting, tự hoàn thành mã thông minh, snippets và cải tiến mã nguồn.

Cài đặt <https://code.visualstudio.com/>

Giao diện ban đầu

**Xcode**

Xcode là bộ phát triển phần mềm tích hợp được Apple phát triển chạy trên hệ điều hành Mac để lập trình viên có thể phát triển phần mềm chạy trên hệ điều hành Mac và iOS.

Cài đặt: download Xcode từ App Store.

**Android Studio**

Android Studio (ide android) là môi trường phát triển tích hợp chuyên nghiệp hỗ trợ việc thiết kế và xây dựng các ứng dụng di động trên nền tảng Android được Google và JetBrains hợp tác xây dựng nhằm thay thế các phiên bản plugin android dành cho Eclipse ngày xưa.

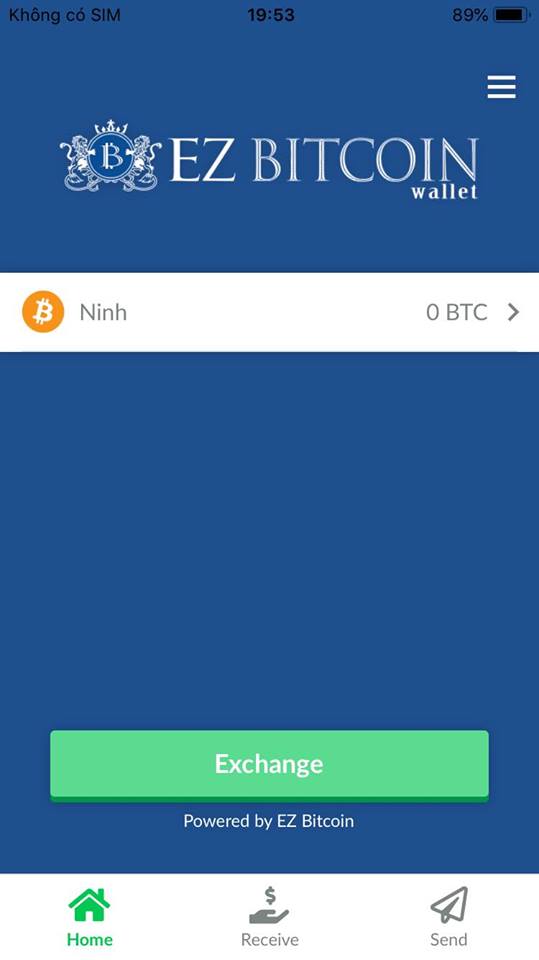
Cài đặt: <https://developer.android.com/studio/install>. Khi tải Android Studio, ngoài Android Studio, bạn sẽ nhận được Android SDK. SDK manager và nhiều công cụ đi kèm khác. Công cụ duy nhất bạn cần là Java Development Kit. (Lưu ý: Android Studio và SDK khá nặng, do vậy đảm bảo bạn có đủ không gian trống trước khi tải).

# 3.2 Giao diện hệ thống

Dưới đây là một số màn hình của ứng dụng đã xây dựng.

# 3.2.1 Màn hình home

Khi mới truy cập ứng dụng, màn hình sẽ hiển thị danh sách ví với coin tương ứng hiện có của người dùng. Người dùng có thể vào xem chi tiết ví khi nhấn chọn một ví coin trong danh sách ví. Người dùng có thể sử dụng chức năng exchange khi nhấn chọn bút “Exchange” phía dưới màn hình. Người dùng có thể thực hiện deposit khi nhấn chọn tab “Receive” hay thực hiện withdraw khi nhấn chọn tab “Send” trên màn hình. Ngoài ra, người dùng có thể di chuyển đến màn hình cài đặt khi nhấn chọn icon menu phía trên góc phải của màn hình.

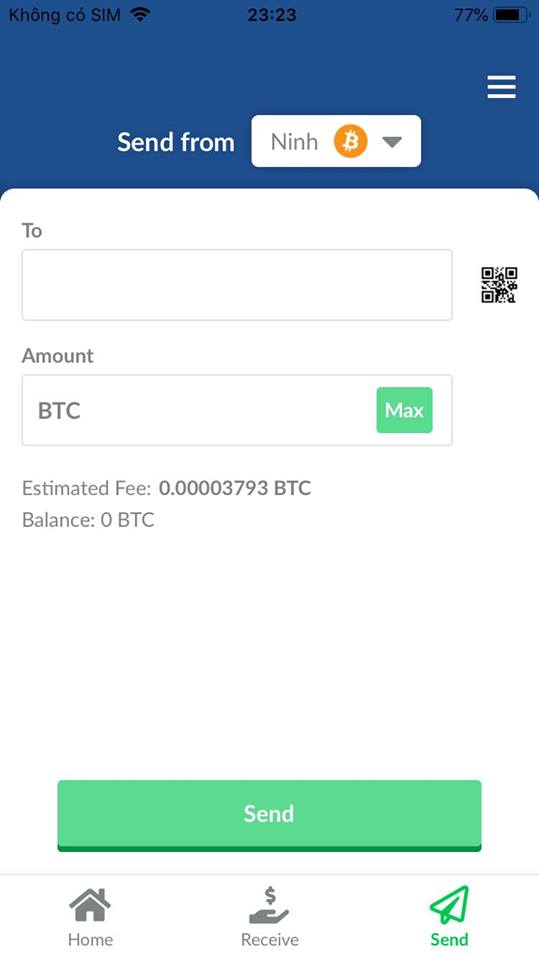


Hình 3.1 Màn hình home

# 3.2.2 Màn hình Send

Khi chọn tab “Send”, người dùng sẽ di chuyển đến màn withdraw. Để withdraw thành công, người dùng chọn ví coin withdraw tại combobox “Send from”, nhập địa chỉ gửi tại ô nhập “To” hoặc quét qr địa chỉ gửi khi nhấn chọn icon qrcode bên cạnh ô nhập, nhập số lượng coin tuỳ ý hoặc nhấn button “MAX” để nhập tối đa lượng coin trong ví. Sau khi nhập đầy đủ thông tin, mức fee phải chịu khi thực hiện withdraw sẽ được ước tính và số dư còn lại của ví. Người dùng nhấn nút “Send” để hoàn tất chức năng.

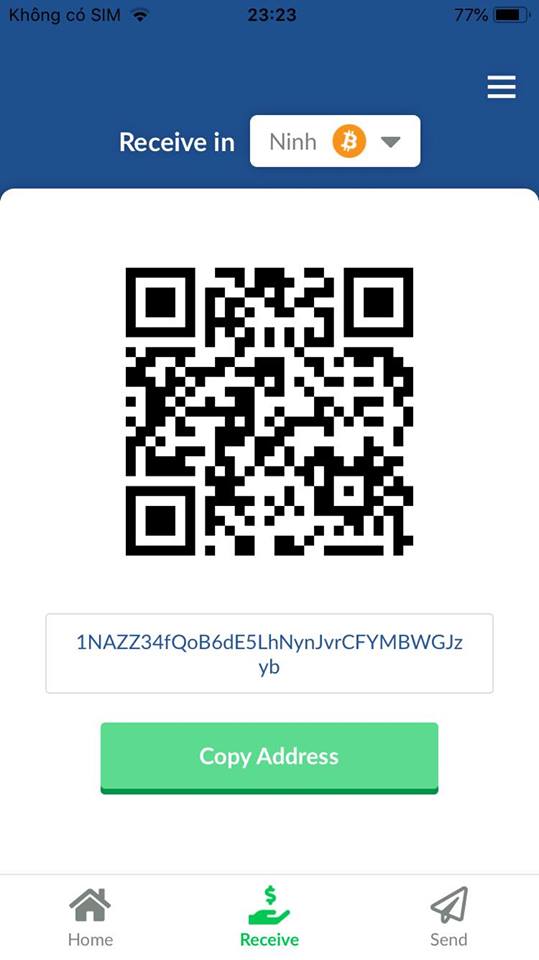
Tại màn hình withdraw, người dùng cũng có thể di chuyển đến màn hình cài đặt khi nhấn icon menu phía trên góc phải.



Hình 3.2 Màn hình Send

# 3.2.3. Màn hình Receive

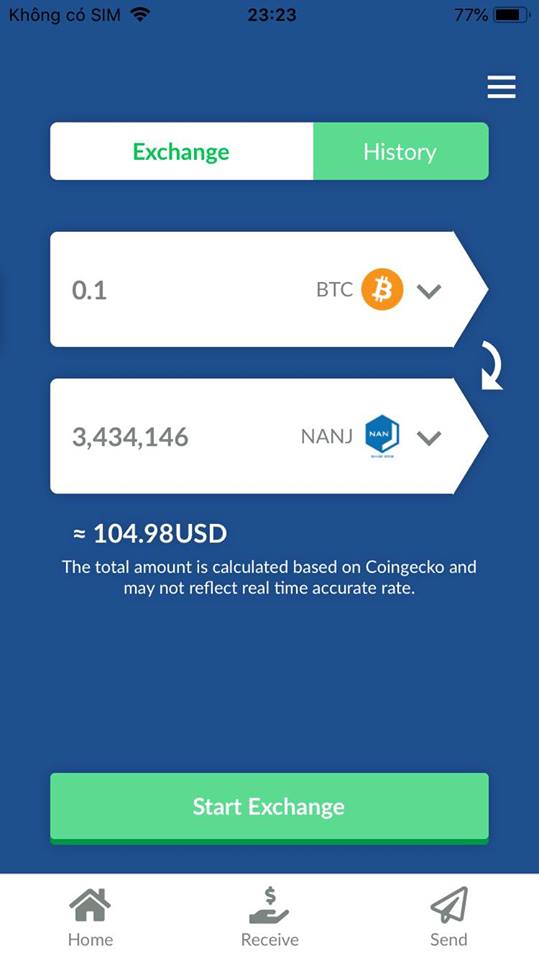
Khi chọn tab “Receive”, người dùng sẽ di chuyển đến màn deposit. Người dùng có thể chọn ví muốn deposit tại combobox “Receive in”. Trên màn sẽ hiển thị chuỗi địa chỉ public của ví, mã qrcode của địa chỉ và nút “Coppy Address” để người dùng sao chép địa chỉ ví. Tại màn hình deposit, người dùng cũng có thể di chuyển đến màn hình cài đặt khi nhấn icon menu phía trên góc phải.



Hình 3.3 Màn hình Receive

# 3.2.4. Màn hình Exchange

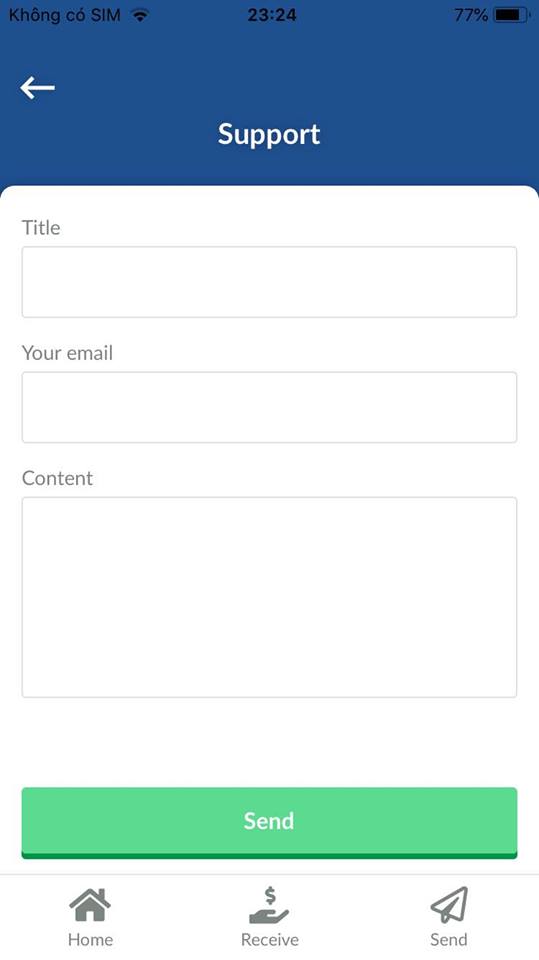
Tại màn hình exchange, người dùng cũng có thể xem thông tin lịch sử exchange khi nhấn chọn tab “History”. Tại tab “Exchange”, để thực hiện exchange thành công, yêu cầu người dùng chọn ví cần exchange, nhập số lượng exchange mong muốn. Khi nhập đầy đủ thông tin, ứng dụng sẽ tính toán quy đổi lượng exchange ra đơn vị USD tương ứng, người dùng nhấn nút “Exchange” để hoàn tất chức năng. Tại màn hình exchange, người dùng cũng có thể di chuyển đến màn hình cài đặt khi nhấn icon menu phía trên góc phải.



Hình 3.4 Màn hình Exchange

# 3.2.5 Màn hình Helps

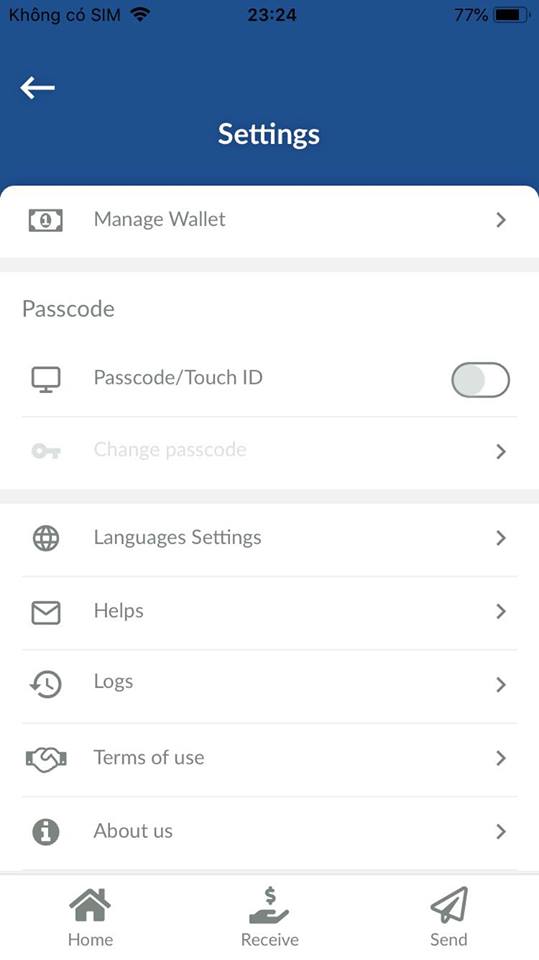
Trong quá trình sử dụng ứng dụng, người dùng có bất cứ thắc mắc về ứng dụng hay góp ý để phát triển ứng dụng có thể gửi mail trực tiếp cho nhà phát triển thông qua chức năng “Helps” trong màn hình cài đặt. Người dùng nhập tiêu đề, địa chỉ mail của người dùng, nội dung muốn gửi vào các ô nhập tương ứng và nhấn “Send” để hoàn thành chức năng.



Hình 3.5 Màn hình Helps

# 3.2.6 Màn hình Settings

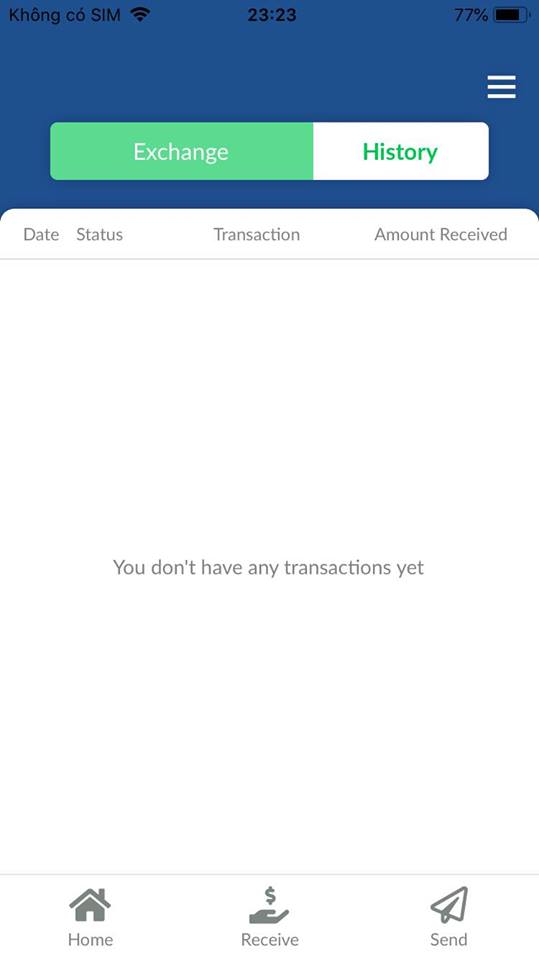
Tại màn hình cài đặt, người dùng có thể cài đặt bảo vệ passcode/touchid, thay đổi ngôn ngữ ứng dụng, xem logs, term of use, about us, thay đổi thông tin ví.



Hình 3.6 Màn hình Settings

# 3.2.7 Màn hình History

Lịch sử các lần exchange của người dùng sẽ được lưu lại với các thông tin cụ thể: ngày tháng giao dịch, trạng thái giao dịch, mã hash giao dịch và số lượng coin trong giao dịch.



Hình 3.7 Màn hình History

**3.3 Kết luận chương 3**

Chương 3 đã nêu được các công cụ để xây dựng nên ứng dụng ví, đồng thời cũng đưa ra được các hình ảnh trực quan của từng chức năng.

# KẾT LUẬN

Ứng dụng ví Bitcoin đã đáp ứng đầy đủ tính năng cần có của một ví tiền điện tử: deposit, withdraw, transaction history… Ngoài ra, ví còn tối ưu được các thao tác người dùng giúp trải nghiệm đơn giản, tiện lợi và an toàn hơn các ứng dụng ví khác.

**Kết luận và hướng nghiên cứu trong tương lai**

Sau quá trình nghiên cứu và thực hiện, cơ bản em đã hiểu biết nhiều hơn về các kỹ thuật trong lập trình React-native; phân thích thiết kế hướng đối tượng theo hướng chuyên nghiệp hơn; hiểu biết hơn về nghiệp vụ ví Bitcoin; nâng cao tinh thần tự học, tự nghiên cứu; Xây dựng đượ ứng dụng có giao diện dễ dùng, thẩm mỹ, đáp ứng đầy đủ tính năng của một ví bitcoin, có tính bảo mật cao, cơ chế rõ ràng.

Với mục đích hoàn thiện hơn nữa ứng dụng của mình để áp ứng tốt nhất cho nhu cầu thực tế, trong tương lại em sẽ cố gắng thêm nhiều ngôn ngữ hỗ trợ hơn, thêm nhiều coin nữa và tăng cường các hình thức bảo mật mới.

Vì thời gian triển khai có hạn và việc tìm hiểu công nghệ mới còn gặp nhiều khó khăn do không có nhiều tài liệu nên khôgn tránh được những sai sót. Em mong nhận được sự góp ý kiến và hướng dẫn của thầy cô để đồ án thêm hoàn thiện.

**TÀI LIỆU THAM KHẢO**

**Tiếng Việt:**

[1] *Andreas M.Antonopoulos, Mastering Bitcoin, Bitcoin thực hành: Những khái niệm cơ bản và cách sử dụng đúng đồng tiền mã hoá*, NXB Đại học kinh tế quốc dân.

[2] <https://vi.wikipedia.org/wiki/Bitcoin>

**Tiếng Anh:**

[1] <https://facebook.github.io/react-native/>

[2] <https://stackoverflow.com/>