

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC VĂN HIẾN



TÌM HIỂU VÀ XÂY DỰNG VÍ ĐIỆN TỬ

BÁO CÁO MÔN THỰC TẬP CƠ SỞ

GVHD: Đỗ Đình Trang

SVTH: Bùi Minh Nhật—201A290002

TP. HỒ CHÍ MINH, 2024

Mục lục

Chương I	Tổng quan về ví điện tử	1
1.1	Giới thiệu ví điện tử	1
1.2	Sơ lược công ty phát hành ví	3
1.3	Những chức năng chính	3
1.3.1	Xem số dư	3
1.3.2	Chuyển tiền nội bộ	3
1.3.3	Rút/Nạp tiền	4
1.3.4	Lịch sử giao dịch	4
1.3.5	Phí giao dịch	5
1.4	Những yêu cầu phi chức năng	5
1.4.1	Khả năng kiểm toán	5
1.4.2	Bảo mật	6
1.4.3	Về mặt hệ thống	7
Chương II	Cơ sở lý thuyết	8
2.1	Phương pháp kế toán ghi sổ kép	8
2.1.1	Credit/Debit	8
2.1.2	Sổ ghi giao dịch (Ledger)	9
2.1.3	Liability	9
2.1.4	Quan hệ Ledger và Liability	10
2.2	Cơ sở dữ liệu quan hệ	10
2.3	Cơ bản về bảo mật	12
2.3.1	Hàm băm	12
2.3.2	Quy trình lưu mật khẩu	13
2.3.3	Giao tiếp mạng - SSL Handshake	14
2.4	Sinh mã giao dịch	15
Chương III	Phân tích và thiết kế hệ thống	18
3.1	Thiết kế cơ sở dữ liệu	18
3.2	Mô phỏng dữ liệu giao dịch	22

3.3	Tính số dư	24
Chương IV	Cài đặt phần mềm	25
4.1	Công cụ	25
4.1.1	Django Framework	25
4.1.2	PostgreSQL	27
4.2	Cấu trúc tổng quát của Ví điện tử	27
4.3	Chạy web app	29
Chương V	Kết luận	33
Tài liệu tham khảo		34

Lời nói đầu

Trong thời đại công nghệ số hiện nay, các dịch vụ thanh toán điện tử đã và đang trở thành một phần không thể thiếu trong cuộc sống hàng ngày của con người. Với sự phát triển mạnh mẽ của Internet và các thiết bị di động thông minh, việc thực hiện các giao dịch tài chính trực tuyến trở nên vô cùng thuận tiện và nhanh chóng. Ví điện tử, một trong những công cụ thanh toán phổ biến nhất trong thời đại số, không chỉ đơn thuần là một phương tiện lưu trữ và quản lý tài sản số, mà còn là cầu nối quan trọng giữa người sử dụng và hệ sinh thái tài chính rộng lớn, bao gồm ngân hàng, tổ chức tín dụng, và các dịch vụ tài chính khác. Ví điện tử giúp người dùng thực hiện các giao dịch mua bán, chuyển khoản, nạp rút tiền một cách đơn giản và an toàn mà không cần phải tiếp xúc trực tiếp với tiền mặt.

Đề tài *Tìm hiểu và Xây dựng Ví điện tử* được sinh viên Bùi Minh Nhật lựa chọn nhằm mục đích nghiên cứu sâu về các nguyên lý cơ bản và các cơ chế hoạt động đằng sau một ví điện tử, cùng với việc lập trình một hệ thống như vậy. Đây là một chủ đề không chỉ thú vị mà còn rất thiết thực, khi mà các hệ thống ví điện tử đang dần trở thành phần không thể thiếu trong các nền tảng tài chính hiện đại. Ngoài ra, đề tài còn đề cập đến những vấn đề về bảo mật, yêu cầu kiểm toán và các quy định liên quan mà một ví điện tử phải tuân thủ, nhằm đảm bảo tính hợp pháp và bảo vệ người dùng.

Lời cảm ơn

Em trân trọng cảm ơn những đóng góp, ý kiến và hỗ trợ từ thầy Đỗ Đình Trang đã đồng hành cùng em trong suốt hành trình thực hiện đề tài này hơn hai tháng qua.

Tổng quan về ví điện tử

1.1 Giới thiệu ví điện tử

Ngân hàng là dịch vụ thiết yếu trong cuộc sống hàng ngày. Hiện nay, nó đang trải qua những thay đổi chưa từng có trước xu hướng của nền kinh tế số. Với sự phổ biến của internet tốc độ cao và các thiết bị điện tử như máy tính bảng và điện thoại thông minh, dịch vụ ngân hàng số (digital banking) ngày càng trở nên phổ biến. Bên cạnh các trang web ngân hàng và ứng dụng ngân hàng di động truyền thống, các dịch vụ thanh toán điện tử và ngân hàng ảo (hoạt động hoàn toàn trực tuyến) gần đây đã nổi lên như những nền tảng chuyển tiền thay thế. Hơn nữa, đại dịch COVID-19 đã đẩy nhanh xu hướng này. Nhiều ngân hàng đã đóng cửa các chi nhánh vật lý và thay thế chúng bằng nhiều nền tảng ngân hàng kỹ thuật số hơn. Một số dịch vụ thanh toán nổi tiếng và ngân hàng ảo có thể kể đến như PayPal, ApplePay, Google Pay, Netbank, Webank,...

Ví điện tử (digital wallet) là ứng dụng di động hoặc web cho phép người dùng lưu trữ, gửi và nhận tiền. Nó loại bỏ nhu cầu sử dụng tiền mặt hoặc thẻ tín dụng bằng cách cho phép người dùng thực hiện giao dịch từ thiết bị di động. Ví điện tử có thể được sử dụng để mua hàng trực tuyến, thanh toán hóa đơn và chuyển tiền cho người dùng khác. Một số ví còn cung cấp các tính năng bổ sung như lưu trữ thẻ khách hàng thân thiết, vé giảm giá, hoàn tiền (cashback),... Ví điện tử (hay các dịch vụ thanh toán trực tuyến) và ngân hàng số là hai dịch vụ tài chính khác biệt, cung cấp cho người dùng nhiều tính năng và lợi ích. Trong khi ngân hàng số chủ yếu là các tổ chức ngân hàng truyền thống cung cấp dịch vụ ngân hàng trực tuyến, ví điện tử là nền tảng độc lập tập trung vào các giao dịch và thanh toán.

Ví điện tử có thể được truy cập từ bất cứ đâu, bất cứ lúc nào, miễn là người dùng có kết nối internet. Điều này đặc biệt hữu ích cho những người thường xuyên đi du lịch hoặc không muốn mang theo tiền mặt. Hơn nữa, ví điện tử thường nhanh chóng và hiệu quả hơn so với ngân hàng số, cho phép người dùng thực hiện giao dịch

gần như ngay lập tức. Một ưu điểm khác của ví điện tử là phí giao dịch thường phải chăng hơn so với ngân hàng truyền thống. Ví điện tử thường có phí thấp hơn và cung cấp tỷ giá hối đoái tốt hơn cho các giao dịch quốc tế, do không phải tuân theo các quy định và chi phí chung như các ngân hàng truyền thống.

Tuy nhiên, việc sử dụng ví điện tử cũng có một số nhược điểm. Một trong những mối quan tâm chính là an ninh. Mặc dù ví điện tử sử dụng mã hóa và các biện pháp bảo mật khác để bảo vệ thông tin tài chính và giao dịch của người dùng, chúng không chịu sự giám sát theo quy định giống như các ngân hàng truyền thống. Điều này có nghĩa là người dùng có thể gặp rủi ro gian lận và đánh cắp danh tính cao hơn. Ngoài ra, ví điện tử có thể không cung cấp mức hỗ trợ khách hàng tương đương với ngân hàng truyền thống. Mặc dù nhiều ví điện tử có sẵn các nhóm dịch vụ khách hàng để hỗ trợ người dùng, họ có thể không đáp ứng hoặc hữu ích như các đại diện ngân hàng truyền thống. Đây có thể là mối lo ngại đối với những người dùng cần trợ giúp giải quyết các vấn đề hoặc tranh chấp liên quan đến tài khoản của họ.

Nhìn chung, ví điện tử cung cấp tùy chọn thanh toán tiện lợi, tương đối an toàn và giàu tính năng cho người dùng hiện đại. Khi ngày càng nhiều người chuyển sang thanh toán kỹ thuật số, ví điện tử có khả năng sẽ trở nên phổ biến hơn trong những năm tới. Xu hướng này đặc biệt rõ ràng ở các thế hệ trẻ, những người có nhiều khả năng sử dụng các phương thức thanh toán kỹ thuật số hơn các thế hệ trước. Khi công nghệ đằng sau ví điện tử tiếp tục phát triển, chúng ta có thể thấy nhiều tính năng và dịch vụ được thêm vào các nền tảng này, khiến chúng trở nên thuận tiện và thân thiện hơn với người dùng.

Sự phát triển của ví điện tử được thúc đẩy bởi những tiến bộ trong công nghệ di động, cũng như những thay đổi trong hành vi và sở thích của người tiêu dùng. Khi ngày càng nhiều người chuyển sang thanh toán kỹ thuật số, các công ty đã đầu tư mạnh vào việc phát triển và cải tiến công nghệ ví điện tử. Bằng chứng là đã có hàng chục nền tảng ví điện tử khác nhau, mỗi nền tảng có một bộ tính năng và dịch vụ riêng biệt. Không chỉ dừng lại ở việc thanh toán hàng hóa và dịch vụ, ví điện tử còn tích hợp nhiều tính năng khác như quản lý tài chính cá nhân, chuyển tiền quốc tế, và thậm chí là đầu tư. Sự tiện lợi và an toàn mà ví điện tử mang lại đã làm thay đổi hoàn toàn cách chúng ta giao dịch, giúp tiết kiệm thời gian và nâng cao trải nghiệm người dùng. Với sự hỗ trợ mạnh mẽ từ các chính phủ và cơ quan quản lý, ví điện tử đang dần trở thành một phần không thể thiếu trong cuộc sống hiện đại.

1.2 Sơ lược công ty phát hành ví

Công ty phát hành ví điện tử vận hành với mạng lưới chi nhánh phân bố ở từng huyện nhằm đảm bảo thuận tiện cho người dùng trong việc nạp và rút tiền. Mỗi chi nhánh chịu trách nhiệm quản lý tài sản tài chính của riêng mình, bao gồm tiền mặt tại quầy và tiền trong tài khoản ngân hàng liên kết với chi nhánh đó. Hệ thống này cho phép người dùng thực hiện giao dịch trực tiếp tại chi nhánh gần nhất mà không cần phụ thuộc hoàn toàn vào các dịch vụ trực tuyến.

Các chi nhánh hoạt động như một đơn vị tài chính độc lập trong mạng lưới của công ty, đảm bảo rằng lượng tiền mặt và tài sản số luôn được cân đối. Khi người dùng thực hiện giao dịch nạp tiền, chi nhánh sẽ xử lý bằng cách tăng số dư ví của người dùng đồng thời giảm lượng tiền mặt tại chi nhánh. Ngược lại, khi rút tiền, hệ thống điều chỉnh số dư ví và xuất quỹ tại chi nhánh, đảm bảo tính minh bạch và chính xác.

Ngoài ra, công ty áp dụng các cơ chế quản lý tài chính chặt chẽ để giám sát tình hình tài sản tại từng chi nhánh, từ đó tối ưu hóa dòng tiền và phòng ngừa gian lận. Các báo cáo định kỳ về tiền mặt và tài khoản số giúp ban quản trị nắm bắt được hiệu quả hoạt động của từng chi nhánh, đảm bảo an toàn tài chính cho toàn hệ thống.

1.3 Những chức năng chính

1.3.1 Xem số dư

Chức năng xem số dư cho phép người dùng kiểm tra số tiền hiện có trong ví điện tử một cách nhanh chóng và chính xác. Hệ thống sẽ lấy dữ liệu từ bảng giao dịch (transaction) để tính toán số dư, bao gồm các khoản nạp, rút, chuyển khoản và các giao dịch đang chờ xử lý.

Tính năng này phải đảm bảo cập nhật theo thời gian thực, đồng thời hiển thị chi tiết từng loại số dư như số dư khả dụng và số dư bị phong tỏa (nếu có). Bên cạnh đó, giao diện cần rõ ràng và dễ sử dụng, giúp người dùng nắm bắt tài chính của mình mọi lúc, mọi nơi.

1.3.2 Chuyển tiền nội bộ

Chuyển tiền (money transfer) là chức năng quan trọng nhất, và được sử dụng nhiều nhất của ví điện tử. Việc gửi tiền từ một người hoặc tổ chức (gọi chung là thực thể) này sang người hoặc tổ chức khác sẽ phát sinh giao dịch (transaction). Một số ví dụ sử dụng chức năng chuyển tiền bao gồm: anh A gửi tiền cho chị B; anh C thanh

toán hóa đơn điện cho EVN; chị D muốn nạp tiền vào ví hoặc rút tiền ra; ví điện tử thu phí của người dùng.

Một yêu cầu phải có của chức năng chuyển tiền là sự chính xác và sự tin cậy (reliability). Bất kỳ hệ thống nào cũng có nguy cơ ngưng hoạt động bất chợt ở giữa giao dịch, khi tiền đã rời khỏi ví của người gửi nhưng chưa đến ví của người nhận. Đây là sự cố đến từ phần cứng, hoặc phần mềm, hoặc ngay cả người dùng. Thứ hai, số tiền giao dịch phải đảm bảo toàn vẹn, tức số tiền chuyển đi cũng bằng số tiền nhận về.

1.3.3 Rút/Nạp tiền

Ví điện tử sẽ không thể hoạt động khi không có tiền. Để rút/nạp tiền, người dùng cần đến những địa điểm hỗ trợ như các chi nhánh của công ty tạo ra ví hoặc đối tác của công ty. Người dùng thực hiện thủ tục cần thiết với giao dịch viên (teller/cashier) để quy đổi tiền mặt và tiền số.

Yêu cầu này cho biết công ty đứng sau ví điện tử phải có một lượng tiền mặt bằng với số lượng tiền số. Bản thân tiền số chỉ là một dạng thể hiện của tiền pháp định trong lưu thông, chứ không phải loại tiền khác. Khi tổng lượng tiền số của ví tăng lên hoặc giảm xuống, thì lượng tiền mặt trong kho cất trữ của công ty phải tăng lên hoặc giảm xuống tương đương.

Quy trình nạp tiền (deposit) như sau:

- Người dùng đưa tiền cho giao dịch viên.
- Giao dịch viên cập nhật tiền cho tài khoản.
- Hệ thống chuyển tiền số sang tài khoản người dùng.

Quy trình rút tiền (withdraw) như sau:

- Giao dịch viên xem chứng minh thư.
- Giao dịch viên sử dụng chức năng rút tiền, nhập số tài khoản và lượng tiền cần rút.
- Hệ thống trừ lượng tiền mặt của chi nhánh đó.
- Giao dịch viên đưa tiền cho người dùng.

1.3.4 Lịch sử giao dịch

Lịch sử giao dịch là một trong những tính năng quan trọng của ví điện tử, và nó đóng một vai trò then chốt trong việc quản lý tài chính của người dùng. Người dùng

theo dõi các giao dịch đã thực hiện và kiểm tra tính chính xác của các giao dịch trong phần lịch sử giao dịch. Nó giúp người dùng theo dõi các giao dịch đã thực hiện, giải quyết các mập mờ liên quan đến các giao dịch, đối chiếu thông tin với các thông báo giao dịch được gửi từ người khác, đảm bảo tính chính xác và minh bạch cho các giao dịch. Vì vậy, khi sử dụng ví điện tử, người dùng luôn chú ý đến tính năng lịch sử giao dịch và đảm bảo rằng các thông tin liên quan đến giao dịch của mình được lưu trữ một cách chính xác và đầy đủ.

Thông thường, các ứng dụng ngân hàng chỉ cho phép xem lịch sử giao dịch trong khoảng thời gian 30 ngày. Nguyên do xuất phát từ hiệu năng của hệ thống. Chi tiết này sẽ được phân tích trong chương Phân tích thiết kế hệ thống của báo cáo.

1.3.5 Phí giao dịch

Nguồn doanh thu của các ví điện tử hầu hết đến từ phí giao dịch. Phí giao dịch có thể được tính bằng phần trăm trên tiền gửi, hoặc phí là một con số cụ thể. Người gửi tiền được tùy chọn bên gửi hoặc bên nhận sẽ trả phí này. Để đơn giản, báo cáo này sẽ tính phí 0.1% cho tất cả giao dịch (ngoại trừ rút/ nạp tiền) theo công thức:

$$\text{phí} = 0.1\% \times \text{số tiền chuyển.}$$

1.4 Những yêu cầu phi chức năng

1.4.1 Khả năng kiểm toán

Khả năng kiểm toán (auditability) là một yếu tố quan trọng trong mọi hệ thống tài chính, đặc biệt là đối với ứng dụng ví điện tử. Trong bối cảnh giao dịch trực tuyến ngày càng phổ biến, việc đảm bảo tính minh bạch, chính xác và khả năng truy xuất của các giao dịch tài chính không chỉ giúp nâng cao độ tin cậy của hệ thống mà còn đáp ứng các yêu cầu pháp lý và kỳ vọng của người dùng.

Một trong những mục tiêu cốt lõi của ứng dụng ví điện tử là tạo sự tin tưởng cho người dùng. Mỗi khi người dùng nạp tiền, rút tiền hoặc chuyển khoản, họ mong muốn rằng giao dịch của mình được xử lý chính xác và có thể kiểm tra lại khi cần thiết. Khả năng kiểm toán cho phép hệ thống ghi lại mọi giao dịch một cách chi tiết, bao gồm các thông tin như số tiền, thời gian, tài khoản liên quan và các hành động thực hiện. Ví dụ Khi một giao dịch bị thất bại hoặc có sự chậm trễ, người dùng có thể yêu cầu kiểm tra nhật ký giao dịch (audit logs) để xác minh nguyên nhân. Điều này giúp

giảm thiểu xung đột, khiếu nại và củng cố lòng tin rằng hệ thống hoạt động công bằng và minh bạch.

Trong lĩnh vực tài chính, các tổ chức và ứng dụng ví điện tử thường phải tuân thủ nghiêm ngặt các quy định pháp luật liên quan đến giao dịch, bảo mật dữ liệu và quản lý rủi ro. Khả năng kiểm toán là một phần không thể thiếu để đáp ứng các yêu cầu này. Quy định pháp lý: Nhiều quốc gia yêu cầu các hệ thống tài chính phải lưu trữ lịch sử giao dịch trong một khoảng thời gian nhất định (thường là 5-10 năm). Khả năng kiểm toán đảm bảo rằng mọi giao dịch đều được ghi lại và có thể truy xuất khi cần. Kiểm toán nội bộ: Khả năng kiểm toán giúp các công ty tiến hành kiểm tra nội bộ định kỳ, đảm bảo rằng không có sai sót hoặc gian lận xảy ra trong hệ thống.

Khả năng kiểm toán là một công cụ quan trọng trong việc phát hiện và phòng ngừa các hành vi gian lận. Một hệ thống kiểm toán tốt cho phép quản trị viên theo dõi các giao dịch bất thường: Như giao dịch số tiền lớn hoặc nhiều giao dịch nhỏ diễn ra trong một thời gian ngắn. Truy cập không hợp lệ: nếu có ai đó cố gắng truy cập trái phép hoặc sửa đổi dữ liệu giao dịch. Hành vi đáng ngờ: Ví dụ, một nhân viên lạm dụng quyền hạn để chỉnh sửa số dư của tài khoản. Nhờ vào khả năng kiểm toán, hệ thống có thể cảnh báo sớm và ngăn chặn các hành vi gian lận trước khi chúng gây ra thiệt hại lớn.

Một ứng dụng ví điện tử phải đảm bảo khả năng truy xuất dữ liệu khi xảy ra sự cố. Trong trường hợp hệ thống gặp sự cố, chẳng hạn như mất dữ liệu hoặc lỗi phần mềm, khả năng kiểm toán giúp khôi phục và đối chiếu dữ liệu nhanh chóng. Điều này cũng đặc biệt hữu ích trong việc giải quyết tranh chấp giữa các bên. Người gửi tiền có thể xác nhận rằng giao dịch đã được thực hiện đúng số tiền và thời gian. Bên nhận tiền có thể kiểm tra nguồn gốc của giao dịch để tránh nhầm lẫn.

1.4.2 Bảo mật

Ứng dụng ví điện tử phải bảo vệ chặt chẽ dữ liệu người dùng, đặc biệt là thông tin nhạy cảm như mật khẩu, số tài khoản, và giao dịch. Dữ liệu cần được mã hóa ở cả phần lưu trữ (AES-256) và truyền tải (TLS/HTTPS), mật khẩu phải được băm bằng bcrypt kèm theo salt để tăng cường bảo mật. Hệ thống cần giới hạn quyền truy cập và ghi lại nhật ký các hoạt động để đảm bảo minh bạch và phát hiện truy cập trái phép.

Đảm bảo tính toàn vẹn của giao dịch là yếu tố sống còn. Hệ thống cần xác thực người dùng mạnh mẽ với các biện pháp như xác thực hai yếu tố (2FA), kiểm tra giao dịch qua nhiều lớp bảo vệ, và phát hiện các hành vi đáng ngờ như giao dịch bất thường

hoặc từ thiết bị không quen thuộc. Mọi giao dịch phải được ghi lại chi tiết trong nhật ký để dễ dàng kiểm tra và giải quyết tranh chấp khi cần. Trong phạm vi đề tài này, sinh viên xin không thêm vào, vì khả năng và công cụ có giới hạn.

Ngoài ra, việc phòng ngừa các cuộc tấn công mạng là bắt buộc. Hệ thống phải bảo vệ chống lại SQL Injection, XSS, và các cuộc tấn công DDoS thông qua chuẩn hóa truy vấn, mã hóa đầu vào, và sử dụng các công cụ bảo vệ như Cloudflare. Kiểm tra bảo mật định kỳ thông qua thử nghiệm thâm nhập và đánh giá lỗ hổng sẽ đảm bảo rằng hệ thống luôn an toàn và đáp ứng các tiêu chuẩn pháp lý như PCI DSS hoặc GDPR.

1.4.3 Về mặt hệ thống

Hai mối quan tâm lớn nhất khi triển khai một hệ thống giao dịch là 1) Độ tin cậy, 2) Khả năng mở rộng. Trong chương tiếp theo, chúng ta sẽ xem xét các kỹ thuật, kiến trúc, và thuật toán khác nhau được sử dụng để giải quyết các mối quan tâm này.

Yêu cầu phải có của ví điện tử là sự tin cậy (reliability) và tính chính xác. Bất kỳ hệ thống nào cũng có nguy cơ ngưng hoạt động bất chợt ở giữa giao dịch, khi tiền đã rời khỏi ví của người gửi nhưng chưa đến ví của người nhận. Đây là sự cố đến từ phần cứng, hoặc phần mềm, hoặc ngay cả người dùng. Thứ hai, số tiền giao dịch phải đảm bảo toàn vẹn, tức số tiền chuyển đi cũng bằng số tiền nhận về. Đây

Khả năng mở rộng (scalability) là một thuật ngữ chỉ khả năng chịu tải tăng (increased load) của hệ thống. Nó đặt ra một nghi vấn: nếu hệ thống có thể chạy tốt với 100 giao dịch cùng lúc thì liệu nó có còn ổn định khi số lượng giao dịch tăng lên 10.000 hay 1 triệu? Đây không đơn thuần là một câu hỏi rạch ròi có/không. Khi nói đến khả năng mở rộng, ta phải biết hệ thống sẽ lớn lên theo hướng nào và đưa ra các lựa chọn phát triển phù hợp, hoặc biết cách thêm tài nguyên tính toán để chịu nhiều tải.

Cơ sở lý thuyết

2.1 Phương pháp kế toán ghi sổ kép

Tất cả các hệ thống kế toán thực tế sử dụng phương pháp kế toán ghi sổ kép (DEA, Double-entry accounting/bookkeeping) cho tất cả các giao dịch. Mục đích và lợi ích của việc sử dụng DEA:

- Loại bỏ khả năng mất tiền do lỗi đột ngột giữa giao dịch (bên gửi hoặc nhận). Tiền không chỉ được tìm thấy dễ dàng, mà còn biết được chính xác những gì đã xảy ra với nó, và hiện tại nó đang ở đâu. Ta có thể xác định tất cả một cách nhanh chóng.
- Thỏa mãn đầy đủ yêu cầu kiểm toán (audit). Ở hệ thống ví điện tử, lưu trữ tốt tài khoản là chưa đủ, điều bắt buộc đối với một doanh nghiệp quản lý tiền của người khác là phải dễ dàng kiểm toán được. Bất kỳ kế toán viên hoặc kiểm toán viên nào cũng phải có khả năng kiểm tra sổ sách mà không bị cản trở.
- Các tác vụ hàng ngày hoặc cuối tháng, chẳng hạn như số dư khi chốt sổ, có thể được tính toán dễ dàng và nhanh chóng. Tất cả các báo cáo, bảng cân đối,... có thể được lấy dễ dàng (với một lệnh SELECT trong cơ sở dữ liệu quan hệ).

2.1.1 Credit/Debit

Nguyên lý đầu tiên của DEA là cặp Credit/Debit. Chúng ta phải hiểu rằng tiền (money) trong lưu thông không tự sinh ra và cũng không tự mất đi, nó chỉ có thể di chuyển. Tiền di chuyển từ ví này sang ví khác. Ví gửi tiền, hay bên gửi, gọi là Debit; ví nhận tiền, hay bên nhận, gọi là Credit. Mọi giao dịch phải bao gồm cặp Credit và Debit mang cùng một số tiền (dĩ nhiên không giới hạn chỉ có hai bên). Một giao dịch bao gồm hai hoạt động: chuyển từ Debit và nhận cho Credit, hai hoạt động này được ghi trên hai dòng trong một cuốn sổ kế toán. Mỗi dòng ghi gọi là một entry, mỗi giao dịch sẽ có 2 entry, tạo nên cái tên double-entry accouting. Ví dụ Alice (tài khoản: 02468)

chuyển \$100 cho Bob (tài khoản: 13579) sẽ phát sinh 2 entry sau:

Debit | \$100 | 02468 | Alice

Credit | \$100 | 13579 | Bob

Nhiều giao dịch (đặc biệt là trong ngân hàng) chỉ bao gồm hai phía Credit và Debit (bằng nhau và ngược lại). Nhưng nói chung, giao dịch có thể có nhiều phía hơn—thường là với thuế và/hoặc phí, nhiều lần chuyển từ tài khoản này sang nhiều tài khoản khác.

2.1.2 Sổ ghi giao dịch (Ledger)

Sổ cái (general ledger) đại diện cho hệ thống lưu giữ hồ sơ về từng giao dịch tài chính diễn ra trong vòng đời của một công ty đang hoạt động và lưu giữ thông tin tài khoản cần thiết để lập báo cáo tài chính của công ty. Thực tế, dữ liệu giao dịch được chia thành các tài khoản như tài sản, vốn chủ sở hữu, doanh thu, chi phí,... Từ ledger, ta có thể theo vết dòng tiền, biết tiền di chuyển do hoạt động nào. Ở hệ thống ví điện tử, chúng ta không cần cầu kì như trên nhưng vẫn phải có những sổ sau đây:

- Sổ ghi hoạt động chuyển tiền (cho ví dụ trên).
- Sổ ghi hoạt động thu phí giao dịch.
- Các sổ ghi chép lượng tiền mặt và lượng tiền số ở mỗi chi nhánh (gọi chung là tài sản của chi nhánh), phục vụ khi khách rút/nạp tiền.

Bản thân hệ thống vẫn có những giao dịch giữa các ledger với nhau, ví dụ khi người dùng nạp tiền bằng tiền mặt thì ledger trong giao dịch sẽ chuyển tiếp cho ledger-tiền-mặt trong chi nhánh để phản ánh lượng tiền mặt đang có, còn nếu nạp bằng ngân hàng thì ledger-số sẽ thay đổi.

2.1.3 Liability

Trong hệ thống ngân hàng nói chung hay ví điện tử nói riêng, liability (nợ phải trả) là khái niệm quan trọng để phản ánh số tiền công ty ví điện tử phải chịu trách nhiệm quản lý thay cho người dùng. Về cơ bản, mỗi khi người dùng nạp tiền vào ví, số tiền này trở thành tài sản của công ty, nhưng đồng thời cũng là một khoản nợ phải trả (liability) của công ty đối với người dùng. Tương tự, khi người dùng thực hiện rút tiền, liability giảm đi tương ứng với số tiền đã được hoàn tất giao dịch.

Trong một giao dịch, liability đại diện cho trách nhiệm tài chính của công ty đối với các số dư tài khoản của người dùng. Ví dụ:

- Nạp tiền: Khi người dùng nạp tiền mặt tại chi nhánh, công ty tăng liability để phản ánh số dư mới của người dùng trong ví. Số tiền nạp này chuyển từ "branch cash ledger"(tiền mặt tại chi nhánh) sang liability của người dùng.
- Rút tiền: Ngược lại, khi rút tiền, liability giảm đi vì số tiền này đã được chuyển từ ví người dùng sang chi nhánh (branch cash ledger).
- Chuyển tiền nội bộ: Trong trường hợp người dùng chuyển tiền cho người khác (intra-transfer), liability không thay đổi về tổng thể. Tuy nhiên, liability được tái phân bổ, hay chuyển đổi giữa hai tài khoản của người dùng trong hệ thống.

2.1.4 Quan hệ Ledger và Liability

Liability được ghi nhận và quản lý thông qua các ledger trong hệ thống ví điện tử. Các ledger khác nhau được thiết kế để phản ánh các tài sản và trách nhiệm tài chính của từng chi nhánh cũng như toàn hệ thống:

Branch cash ledger: Ghi nhận lượng tiền mặt tại mỗi chi nhánh, bao gồm các giao dịch nạp, rút tiền.

Branch digital ledger: Quản lý số tiền kỹ thuật số của chi nhánh trong tài khoản ngân hàng liên kết.

Central liability ledger: Theo dõi tổng liability của công ty, bao gồm tất cả số dư của người dùng trên toàn hệ thống. Đây là ledger tham gia vào các giao dịch nội bộ.

Trong kiểm toán, ledger giúp đảm bảo rằng các giao dịch được cân đối. Khi có giao dịch phát sinh, các ledger phải được cập nhật để duy trì trạng thái cân bằng giữa tài sản và liability. Ví dụ, nếu người dùng nạp tiền tại chi nhánh, ledger của "branch cash" giảm, "central liability" tăng, và cả hai thay đổi này phải khớp về mặt giá trị. Điều này đảm bảo tính minh bạch và toàn vẹn tài chính của hệ thống.

2.2 Cơ sở dữ liệu quan hệ

Các mô hình dữ liệu là một trong những thành phần quan trọng nhất khi phát triển hệ thống. Bởi vì chúng đại diện không chỉ cách hệ thống được xây dựng, mà còn là cách chúng ta nghĩ về vấn đề đang cần giải quyết. Mô hình quan hệ từng là một đề xuất về mặt lý thuyết và nhiều người vào thời điểm đó đã nghi ngờ liệu nó có thể được triển khai hiệu quả hay không. Tuy nhiên, vào giữa những năm 1980, các hệ thống quản

lý cơ sở dữ liệu (CSDL) quan hệ (RDBMS, relational database management system) và SQL đã trở thành công cụ được lựa chọn cho hầu hết những người cần lưu trữ và truy vấn dữ liệu với một số loại cấu trúc thông thường. Sự thống trị của CSDL quan hệ đã kéo dài khoảng 30-35 năm cho đến bây giờ.

Cơ sở dữ liệu quan hệ (Relational Database) là một hệ thống quản lý dữ liệu tổ chức thông tin dưới dạng bảng (table) có mối quan hệ với nhau. Dữ liệu trong cơ sở dữ liệu được lưu trữ dưới dạng các bảng bao gồm các hàng (rows) và cột (columns), tương ứng với các bản ghi và thuộc tính của dữ liệu. Hệ quản trị cơ sở dữ liệu quan hệ (RDBMS - Relational Database Management System) như MySQL, PostgreSQL, và Oracle là những công cụ phổ biến được sử dụng để quản lý loại cơ sở dữ liệu này.

Các thành phần chính trong cơ sở dữ liệu quan hệ

- **Table (Bảng):** Là nơi lưu trữ dữ liệu. Ví dụ, bảng *People* lưu thông tin người dùng, bảng *Transactions* lưu dữ liệu giao dịch, và bảng *Ledgers* lưu trữ tài sản của từng chi nhánh.
- **Row (Hàng):** Đại diện cho một đối tượng cụ thể trong bảng. Ví dụ, một hàng trong bảng *People* chứa thông tin của một người dùng duy nhất.
- **Column (Cột):** Là thuộc tính hoặc đặc điểm của đối tượng. Ví dụ, bảng *People* có các cột như *NationalID*, *Name*, *PhoneNumber*.
- **Primary Key (Khóa chính):** Là một cột hoặc tập hợp cột dùng để nhận diện duy nhất một bản ghi. Ví dụ, *NationalID* trong bảng *People* là khóa chính.
- **Foreign Key (Khóa ngoại):** Là cột liên kết với khóa chính của bảng khác, tạo ra mối quan hệ giữa các bảng. Ví dụ, cột *AccountNumber* trong bảng *Transactions* liên kết đến *AccountNumber* trong bảng *People*.
- **Relationship (Quan hệ):**
 - *One-to-One (1:1)*: Một người dùng trong bảng *People* có một tài khoản và một mật khẩu *Passwords*.
 - *One-to-Many (1:N)*: Một tài khoản có thể thực hiện nhiều giao dịch, do đó một hàng trong bảng *Accounts* liên kết với nhiều hàng trong bảng *Transactions*.
 - *Many-to-Many (M:N)*: Một nhân viên có thể làm ở nhiều chi nhánh, một chi nhánh có nhiều nhân viên *StaffBranchesHistory*.
- **Schema (Lược đồ):** Là cấu trúc logic của cơ sở dữ liệu, định nghĩa các bảng, cột, kiểu dữ liệu, và các mối quan hệ giữa chúng.

Đặc điểm nổi bật của cơ sở dữ liệu quan hệ

- **Tính toàn vẹn dữ liệu:** Các ràng buộc như khóa chính và khóa ngoại đảm bảo dữ liệu chính xác và hợp lệ.
- **Truy vấn mạnh mẽ:** Hỗ trợ ngôn ngữ truy vấn SQL (*Structured Query Language*) để thao tác và truy xuất dữ liệu.
- **Tính nhất quán:** Dữ liệu được tổ chức tránh dư thừa, đảm bảo tính nhất quán qua các quy tắc chuẩn hóa.

2.3 Cơ bản về bảo mật

2.3.1 Hàm băm

Bất kỳ hệ thống nào cũng bắt buộc phải lưu mật khẩu đúng cách. Cách tệ nhất để lưu mật khẩu của người dùng trong database là giữ nguyên ở dạng ban đầu người dùng nhập. Nếu database bị hack, hoặc đơn giản là SQL-injection, mật khẩu sẽ lộ ra theo cách không thể dễ hơn để đánh cắp. Giải pháp là biến đổi mật khẩu thành một chuỗi khác trước khi lưu trữ. Việc này được thực hiện bằng hàm băm.

Hàm băm, trong bảo mật, sẽ biến dữ liệu có kích thước tùy ý thành những chuỗi có kích thước cố định. Điều này đồng nghĩa là nó sẽ xử lý vô hạn các trường hợp dữ liệu và đưa chúng về một tập hợp hữu hạn kết quả băm. Gọi tập hợp (set) mật khẩu là \mathcal{U} ; số lượng mật khẩu kí hiệu là u (giá trị u theo lý thuyết là vô hạn, vì người dùng có thể nhập bất kỳ thứ gì); gọi \mathcal{T} là tập hợp chứa chuỗi sau băm; gọi t là số lượng phần tử trong \mathcal{T} . Thông thường, t sẽ nhỏ hơn u rất nhiều. Hàm băm h là một hàm:

$$h : \mathcal{U} \rightarrow \{s_0, s_1, s_2, \dots, s_t\}.$$

biến đổi mỗi phần tử trong \mathcal{U} thành một chuỗi s_i tương ứng trong \mathcal{T} . Ví dụ, $h(\text{'xinhayquenanh'})$ sẽ cho kết quả yNaQaazbBkcY và $h(\text{'boyphoco'})$ sẽ cho ra 3XpzQeNV2n3v. Hai chuỗi vào có kích thước khác nhau nhưng kết quả băm có độ dài bằng nhau. Kết quả băm gọi là digest hay hash code.

Với x khác y , hiện tượng $h(x) = h(y)$ gọi là collision (trùng kết quả băm). Ở trường hợp lưu mật khẩu chúng ta đang xét, khả năng xảy ra collision là cực kỳ thấp. Vì trong thực tế, mật khẩu thông dụng chỉ bao gồm mười mấy ký tự, trong khi hàm băm lại trả về digest dài hơn (32, 64,...), tức t lớn hơn u .

Định nghĩa về hàm băm trong cấu trúc dữ liệu bảng băm và hàm băm trong

mật mã có một số điểm khác nhau. Song việc bám vào định nghĩa thực sự không quan trọng bằng các tính chất của hàm băm phổ biến hiện nay. Hàm băm/thuật toán băm cần có các tính chất sau:

- Băm các input giống nhau phải cho ra các output giống nhau. Băm các input khác nhau phải cho ra các output khác nhau (khả năng xảy ra hiện tượng trùng nhau cực kỳ thấp).
- Cho hai input a và b , việc biết giá trị băm của a không liên quan gì đến việc biết giá trị băm của b .
- Không thể suy ngược input từ giá trị băm. Một số thuật toán băm nổi tiếng có thể được kể đến như: MD5, SHA-1, SHA-2, CRC32, RipeMD160, bcrypt,...

2.3.2 Quy trình lưu mật khẩu

Băm mật khẩu bằng hàm có tốc độ nhanh (như MD5) là hành động cực kỳ tai hại. Hacker có thể tra ra mật khẩu từ digest dễ dàng bằng brute-force ¹. Mật khẩu phải được băm bằng hàm có tính chất memory-hard. Memory-hard function tức là những hàm cần nhiều bộ nhớ để tính digest nhanh, và nếu sử dụng ít bộ nhớ sẽ phải đợi lâu. Tóm lại, hàm băm phải là hàm chạy chậm, tốn nhiều chi phí tính toán ở mức nào đó. Việc tra ngược mật khẩu sẽ chậm hơn hàng (chục) nghìn lần. Như vậy ở hệ thống ví điện tử, ta có thể sử dụng các hàm băm như Argon 2, bcrypt, hoặc scrypt,...

Nhưng băm mật khẩu thôi là chưa đủ. Người dùng hay sử dụng những mật khẩu rất đơn điệu và phổ biến như 123456789 hay 0000, dễ nhớ nhưng cũng dễ bị tra ngược. Thứ hai, cũng có những mật khẩu vô tình trùng nhau, chỉ cần giải được một digest là biết được tài khoản của nhiều người. Giải pháp cho vấn đề này là nối thêm một chuỗi ký tự khác vào mật khẩu, sau đó đem đi hash. Chuỗi thêm vào gọi là salt. Thay vì $h(\text{password})$ ta sẽ dùng $h(\text{password}+\text{salt})$. Nếu ta sinh salt ngẫu nhiên cho từng mật khẩu, hacker sẽ phải tính toán toàn bộ trường hợp cộng với riêng từng salt. Chi phí cho 2 phép tính này là vô cùng lớn và tốn rất nhiều thời gian để thực hiện. Chúng ta hoàn toàn có thể lưu salt chung với digest. ²

Cuối cùng ta có quy trình chứng thực như sau:

¹Xem vụ VNG rò rỉ mật khẩu hash bằng MD5 vào năm 2018

²Để đẩy bảo mật lên một mức cao hơn, ta có thể băm mật khẩu rồi mã hóa digest. Khóa (key) của bước mã hóa được lưu ở một server khác. Phương pháp này không phải ý tưởng mới, cũng được sử dụng ở Facebook (<https://bristolcrypto.blogspot.com/2015/01/password-hashing-according-to-facebook.html>).

1. Người dùng gửi password bản rõ lên server.
2. Server băm password+salt, sau đó giải mã.
3. Server kiểm tra trùng khớp với digest đã lưu.

2.3.3 Giao tiếp mạng - SSL Handshake

Trong thế giới kỹ thuật số ngày nay, an toàn thông tin là một yếu tố cực kỳ quan trọng. Một trong những cơ chế bảo mật then chốt mà chúng ta sử dụng hàng ngày mà có thể không nhận ra, chính là quá trình SSL handshake. Đây là quy trình mà các thiết bị và máy chủ sử dụng để thiết lập một kết nối an toàn và được mã hóa trên internet.

1. Client Hello: Quy trình SSL handshake bắt đầu khi một khách hàng (client), chẳng hạn như trình duyệt web gửi một thông điệp “Hello” đến máy chủ. Thông điệp này bao gồm các thông tin về phiên bản SSL/TLS mà client hỗ trợ, danh sách các cipher suites (bộ mã hóa) mà nó có thể sử dụng và một chuỗi byte ngẫu nhiên gọi là “client random.”

Server Hello: Sau khi nhận được thông điệp từ client, máy chủ sẽ phản hồi bằng một thông điệp “Hello” của riêng nó. Thông điệp này chứa chứng chỉ số của máy chủ (SSL certificate), cipher suite mà máy chủ chọn để sử dụng, và một chuỗi byte ngẫu nhiên khác gọi là “server random”.

Xác Thực Chứng Chỉ: Client sẽ xác thực chứng chỉ số của máy chủ bằng cách kiểm tra nó với cơ quan cấp chứng chỉ (CA). Điều này đảm bảo rằng máy chủ thực sự là máy chủ mà nó tuyên bố và không phải là một kẻ giả mạo.

Trao đổi khóa Key Exchange Trong bước này, client và máy chủ trao đổi các khóa để thiết lập một kênh truyền thông được mã hóa. Quá trình này bao gồm việc tạo ra các khóa phiên (session keys) cho việc mã hóa đối xứng, đảm bảo rằng dữ liệu được truyền giữa client và máy chủ là an toàn và không thể bị giải mã bởi bên thứ ba.

Hoàn tất Cuối cùng, cả client và máy chủ gửi một thông điệp “finished” để xác nhận rằng quá trình handshake đã hoàn tất và kết nối bảo mật đã được thiết lập. Từ đây, tất cả các dữ liệu truyền giữa client và máy chủ sẽ được mã hóa, đảm bảo tính bảo mật và toàn vẹn của dữ liệu.

Quy trình SSL handshake không chỉ là một bước kỹ thuật trong kết nối mạng mà còn là một bước bảo mật vô cùng quan trọng. Nó đảm bảo rằng dữ liệu được trao

đổi giữa client và máy chủ là an toàn, bảo vệ người dùng khỏi các cuộc tấn công man-in-the-middle (MiTM). Bằng cách xác thực máy chủ và mã hóa dữ liệu truyền tải, SSL handshake tạo ra một nền tảng bảo mật vững chắc cho các giao dịch trực tuyến.

2.4 Sinh mã giao dịch

Hệ thống chuyển tiền không thể thiếu được mã giao dịch. Mã giao dịch dùng để tra cứu và giải quyết các vấn đề phát sinh sau này. Chúng ta cần phân biệt hai loại transaction: transaction trong database và transaction ngoài thực tế.

Transaction trong database tượng trưng cho một đơn vị công việc, được thực hiện trong HQTCSDL, được xử lý theo cách mạch lạc và đáng tin cậy, độc lập với các transaction hệ thống khác. Một transaction thường đại diện cho bất kỳ thay đổi nào trong database. Còn transaction bên ngoài là dòng tiền chuyển từ bên này sang bên khác, có thể có nhiều bên tham gia. Một transaction thực tế được biểu diễn bằng một hoặc nhiều transaction trong database. Ví dụ Alice chuyển tiền cho Bob sẽ bao gồm 3 transaction sau trong database: Alice \rightarrow Ledger; Ledger \rightarrow Bob; Ledger \rightarrow FeePool.

Mặc dù về mặt hệ thống là 3 transaction riêng biệt nhưng bản chất ngoài đời chúng vẫn là một. Để đại diện cho transaction người ta sử dụng Reference code (ghi tắt là ref code). Ref code là một dãy số duy nhất phân biệt các transaction với nhau. 3 transaction trong hệ thống đã nói trên sẽ sử dụng chung 1 ref code. Theo yêu cầu của hệ thống giao dịch, ref code được sử dụng rất thường xuyên nên cần suy tính kỹ phương pháp sinh ra nó, sao cho đảm bảo không trùng nhau và tốc độ sinh phải nhanh.

Bối cảnh

Hệ thống giao dịch là một trong những hệ thống có lượng tải traffic nhiều nhất. Với dữ liệu càng ngày càng to ra thì hệ thống chỉ có 1 database duy nhất có thể sẽ không thể đáp ứng được, tốc độ truy vấn sẽ trở nên ì ạch. Để giải quyết vấn đề đó, người ta tách database ra thành nhiều database khác nhau, và mỗi database đó sẽ chứa 1 phần dữ liệu. Ví dụ DB-1 chứa thông tin của tài khoản 0001 đến 1000, và DB-2 chứa tài khoản 1001 đến 2000,... Và khi query sẽ tìm xem tài khoản thuộc database nào và thực hiện truy vấn. Mỗi database tách biệt gọi là shard. Tuy nhiên tại thời điểm này có 1 bài toán được đặt ra: Làm thế nào có thể sinh ra ref code duy nhất trên từng shard mà không sợ bị trùng lặp.

Hai cách sinh phổ biến mà ta có thể nghĩ tới là sử dụng auto increment sẵn có trong database và dùng chuẩn sinh ngẫu nhiên UUID.

Auto increment: Cách dùng rất đơn giản, lúc tạo table chỉ cần khai báo auto increment là xong. Tuy nhiên, nó chỉ sử dụng trong 1 database và không thể phân chia sang database khác được. Nó cũng không thể đảm bảo rằng việc sinh ref code ở nhiều database là không bị trùng nhau.

UUID (Universally Unique Identifier): là 1 chuẩn chung nhằm sinh ra chuỗi random không trùng nhau (xác suất gần như bằng 0). Ví dụ: ab2cd220-0b54-4d59-acf3-828ff9bc5af2. dù chạy trên nhiều shard cùng thời điểm đi chăng nữa thì xác suất các string đó trùng nhau dường như gần bằng 0. Tuy vậy nó chứa tận 128 bit, khá to và thừa thãi (2^{128} là con số cực lớn). Thực tế chỉ cần 64 bit là đủ.

Phương pháp sinh mã giao dịch

Chúng ta sẽ dùng một thuật toán đơn giản để sinh ra 1 chuỗi ref code ngẫu nhiên duy nhất từ một vài input. Và từ ref code có thể giải mã ngược lại để lấy ra được input. Thuật toán này chính là thuật toán Instagram dùng để sinh id cho các ảnh do người dùng đăng. Ref code có độ dài 64 bit, bao gồm những thành phần sau:

- 41 bits để lưu thời gian (đơn vị miliseconds). Khoảng thời gian này sẽ được tính từ ngày 1970-01-01 lúc 00:00 hay còn gọi là UNIX epoch.
- 13 bits để lưu shard ID. (tối đa có thể tạo ra được $2^{13} = 8192$ shard)
- 10 bits để lưu auto-incrementing sequence, sau đó mod cho 1024 ($2^{10} = 1024$). Điều đó nghĩa là trong 1 milisecond có thể tạo ra 1024 ID trên 1 shard.

Giả sử: Database chia nhỏ dựa trên số tài khoản thành 2000 shards. Chúng ta có một giao dịch vào ngày lúc 2023-09-01 00:00, phát sinh bởi tài khoản số 1019475427. Kể từ UNIX epoch đã trải qua 1693501200000 miliseconds. Vì thời gian cần 41 bits để lưu, chúng ta dịch giá trị này sang trái 23 bits (left shift, “điền” thêm 23 số 0 sau con số trên) để dãy bit có đủ 64 bits:

$$\text{ref_code} = 1693501200000 \ll (64 - 41) = 14206117714329600000$$

Shard_ID được tính bằng số tài khoản mod cho số lượng shards:

$1019475427 \bmod 2000 = 1427$. Chúng ta sẽ điền 13 bit tiếp theo (vào phía sau) bằng giá trị 1427 sử dụng thêm toán tử OR:

$$\begin{aligned} \text{ref_code} &= 14206117714329600000 \mid 1427 \ll (64 - 41 - 13) \\ &= 14206117714331061248. \end{aligned}$$

Cuối cùng, chúng ta sẽ thêm auto-incrementing sequence (dãy tăng này là khác nhau ở mỗi table) và điền nốt 10 bit còn lại. Giả sử ta đã sinh 5000 ref code, giá trị tiếp là 5001; ta sẽ mod cho 1024 để vừa đủ 10 bit rồi thêm vào dãy ref_code:

$$\text{ref_code} = 14206117714331061248 \mid (5001 \bmod 1024).$$

Cuối cùng chúng ta tính ra được ref_code cho giao dịch trên là:

$$\begin{aligned} \text{ref_code} &= 1693501200000 \ll (64 - 41) \\ &\mid 1427 \ll (64 - 41 - 13) \\ &\mid (5001 \bmod 1024) \\ &= 14206117714331062153. \end{aligned}$$

Để tra ngược lấy thời gian, ta chỉ dùng những phép toán ngược lại: dịch ref_code sang phải $(64 - 41)$ bit rồi AND với chuỗi bit gồm 41 số bit 1. Shard_ID và sequence cũng được làm theo cách tương tự.

$$\text{time} = 14206117714331062153 \gg 23 \ \& \ 2199023255551 = 1693501200000.$$

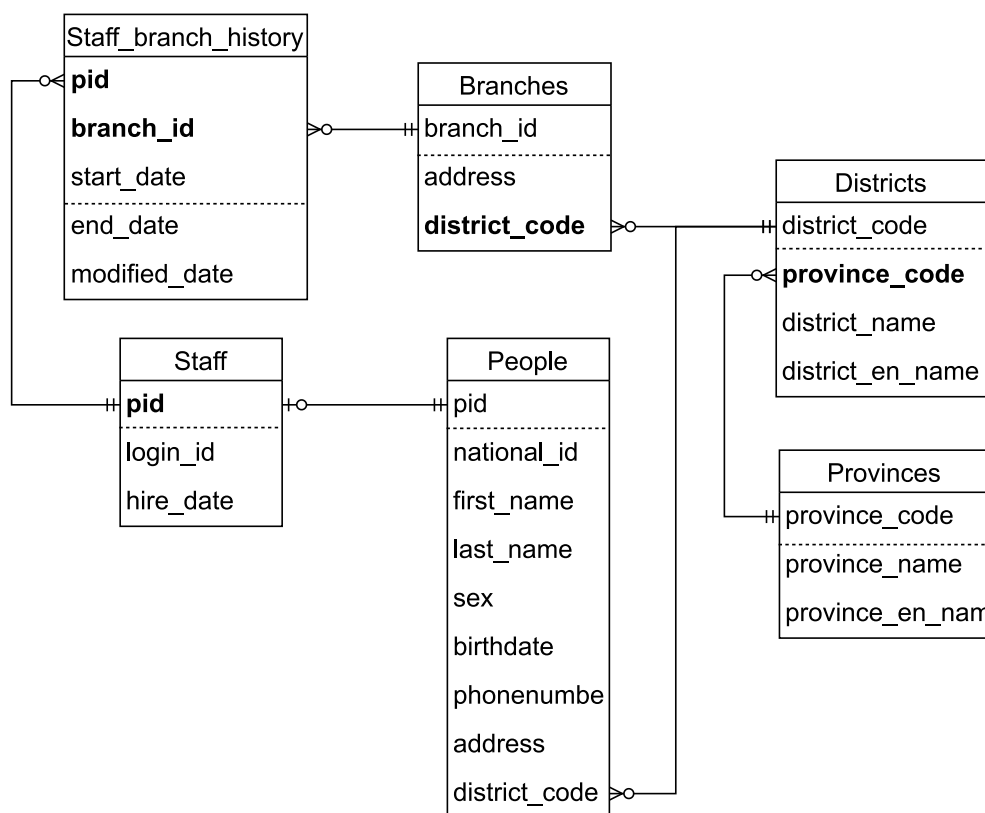
(Con số 2199023255551 chính là chuỗi bit 41 số 1 được chuyển sang hệ thập phân.)

Phân tích và thiết kế hệ thống

3.1 Thiết kế cơ sở dữ liệu

Tổ chức công ty

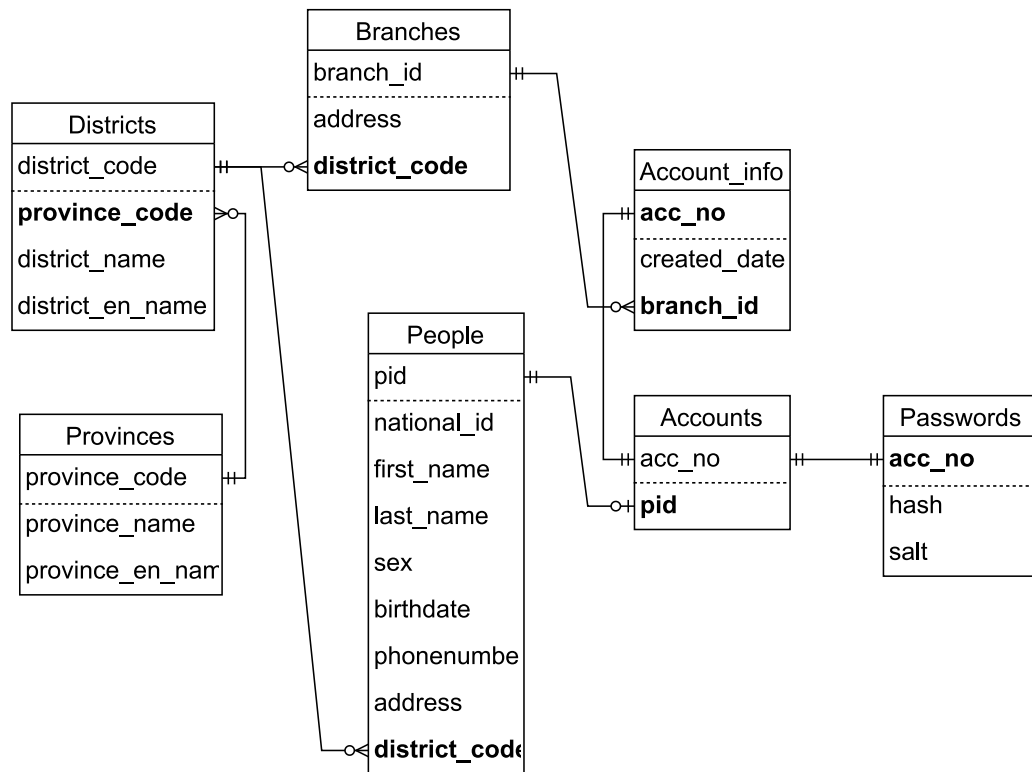
Công ty chia làm nhiều chi nhánh để tiện giao dịch rút hay nạp tiền. Chi nhánh phân biệt bằng địa chỉ. Mỗi chi nhánh có nhiều nhân viên làm việc. Và công ty cũng cần quản lý lịch sử ngày tháng làm việc của nhân viên. Nhân viên cũng cần tài khoản đăng nhập để cập nhật tiền bạc lên hệ thống.



Hình 3.1.1: Database cho tổ chức công ty.

Quản lí người dùng và tài khoản

Mỗi người dùng chỉ có một tài khoản. Tài khoản được phân biệt với nhau bởi chính số tài khoản (account number). Để bảo mật, mỗi tài khoản đi cùng với mật khẩu đã băm và salt tương ứng.



Hình 3.1.2: Database cho phần quản lí tài khoản.

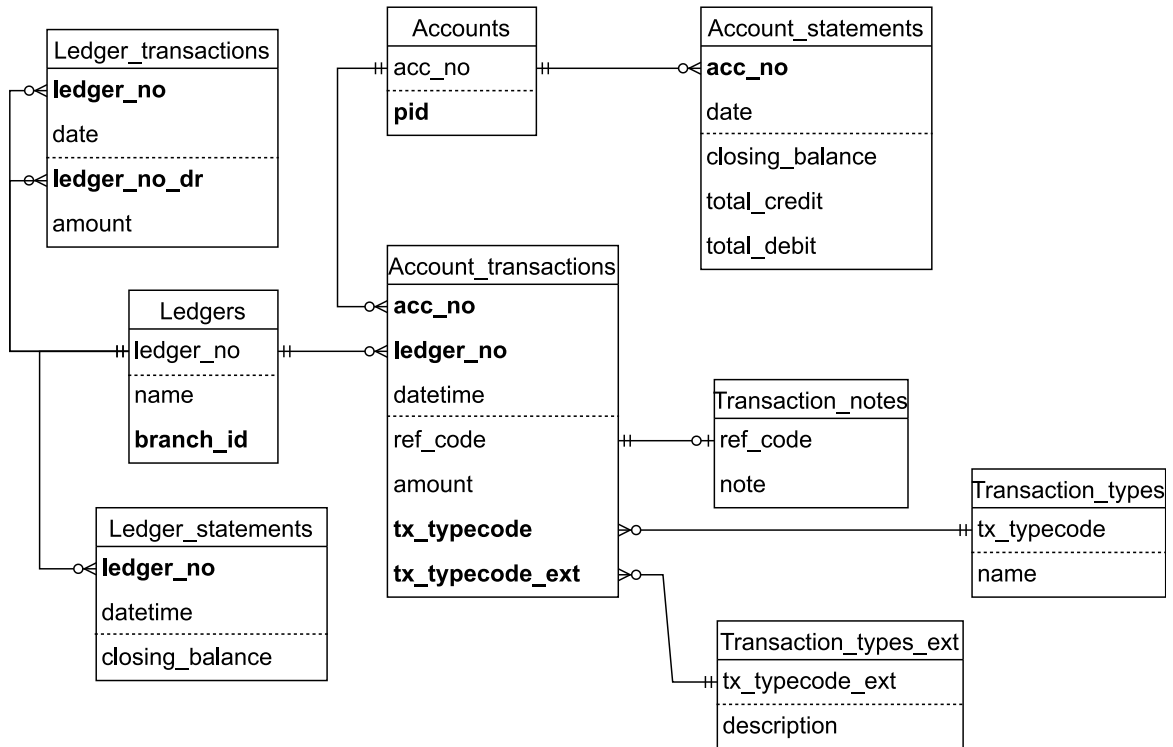
Giao dịch

Theo phương pháp kế toán ghi sổ kép, mỗi giao dịch bao gồm 2 bên: Credit (bên gửi) và Debit (bên nhận). Trong hệ thống ví điện tử, một giao dịch thực tế bao gồm nhiều giao dịch con trong hệ thống. Những giao dịch con hoặc là Account-Ledger, hoặc là Ledger-Ledger; không tồn tại giao dịch giữa hai account với nhau.

Phân tích table Account_transactions:

- Cột amount là số lượng tiền di chuyển. Giá trị tiền luôn là số dương. (Thực tế không có tờ tiền nào in là -\$100. Và làm việc với hai con số âm sẽ cho ra một thứ rất khác.)
- Hướng di chuyển của tiền là cột tx_typecode: Debit hoặc Credit, gán cho tài khoản người dùng (cột acc_no).

- Có người cho rằng theo chuẩn hóa chúng ta chia một transaction thành hai dòng, một để trừ bớt bên gửi, một để thêm tiền bên nhận. Đây là nhận định sai lầm. Nếu Alice chuyển tiền cho Bob thì cũng đồng nghĩa ta suy ra Bob nhận tiền từ Alice. Như vậy việc lưu hai dòng là thừa thãi.
- Cột tx_typecode_ext để chỉ rõ nó trên thực tế là kiểu giao dịch gì. Ví dụ như: rút tiền, phí giao dịch,... hay đơn giản chỉ là tiền di chuyển.



Hình 3.1.3: Database lưu trữ giao dịch.

Quản lý số dư

Một trong những yêu cầu cơ bản nhất của ví điện tử là quản lý số dư của tài khoản. Chúng ta cần xem xét vài thứ sau:

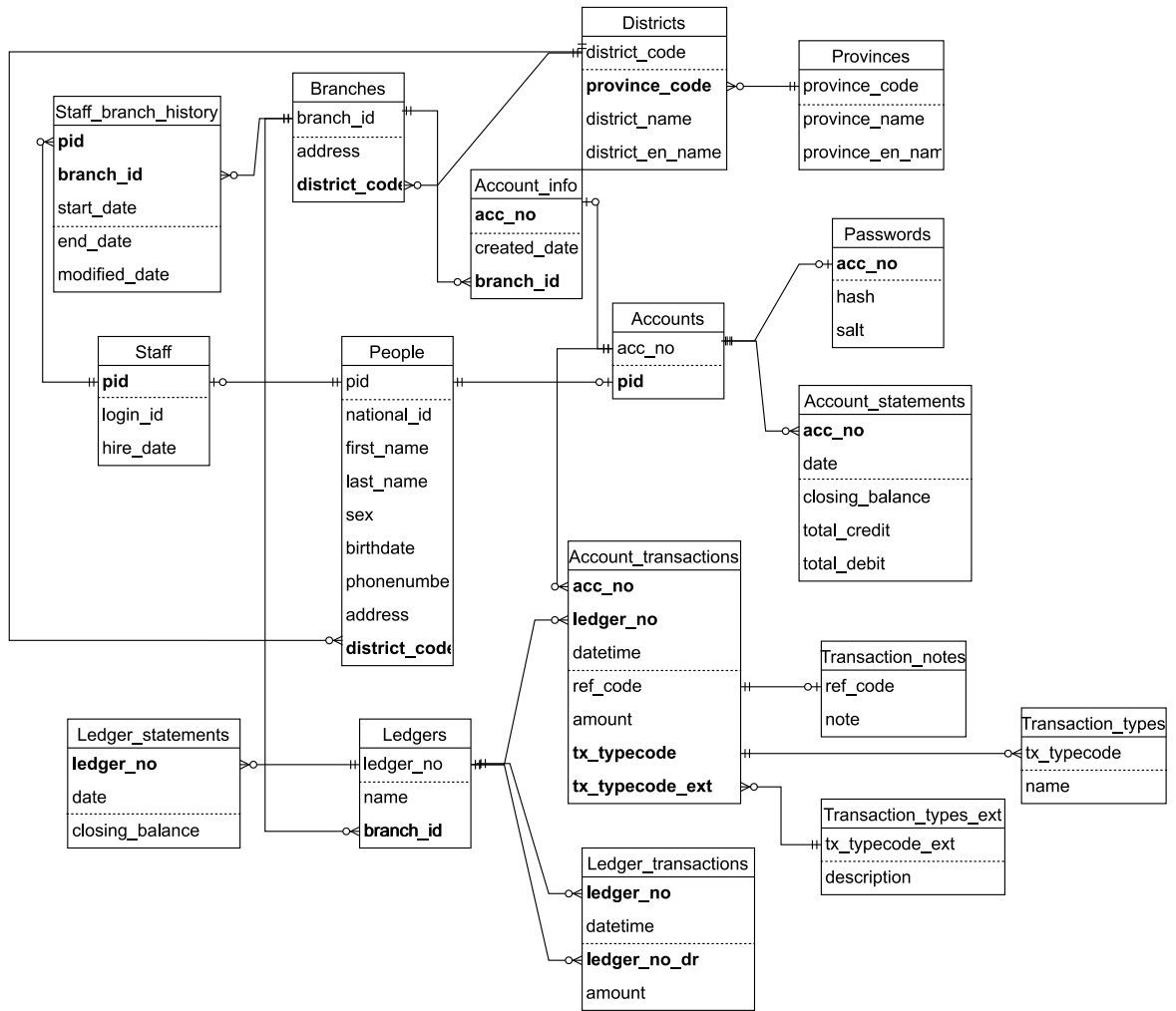
- Nếu chúng ta tính toán được số dư từ các giao dịch trước đó (derived balance), thì việc tạo một cột mới để ghi lại số dư là không cần thiết. Cột đó sẽ gây ra trùng lặp dữ liệu (duplication), phá vỡ nguyên tắc chuẩn hóa. Thứ hai, dữ liệu sẽ xuất hiện các update bất thường (anomaly).
- Nếu chúng ta sử dụng cột ghi số dư, khi một giao dịch mới được thêm vào CSDL, thì cột này trở nên lạc hậu và buộc phải được update liên tục mỗi giao dịch. Đây

là một lý do nữa để ta không sử dụng cách lưu số dư.

- Số dư có thể được công bố trong các bản báo cáo tài chính hằng tháng; các tài liệu đó thường có ràng buộc về mặt pháp lý. Giá trị số dư hiện tại, giả sử là \$450, sau khi được công bố vào ngày 01/09/2023, không được phép thay đổi. Bất kỳ thay đổi nào sau ngày công bố đều là bằng chứng của hành vi không trung thực và gian lận. Con số \$450 đó phải được xem là con số kiểm toán, đã công bố và không thay đổi. Nếu ta dùng cách update số dư sau mỗi giao dịch, thì sau này không thể tra ra chính xác con số \$450 vào ngày 01/9/2023.
- Để sửa một giao dịch lỗi trong quá khứ, ta phải tạo một giao dịch mới trong tháng hiện tại, bởi vì giao dịch cũ đã được kiểm toán và không thể thay đổi.

Như vậy, cách tốt nhất là tính toán số dư từ các giao dịch trước đó. Nhưng một vấn đề nữa, là dữ liệu giao dịch càng ngày càng phình to, dẫn đến thời gian truy vấn số dư rất chậm. Giải pháp là lưu lại số dư của mỗi tháng (số dư chốt sổ/closing balance). Khi cần tính số dư ở thời điểm bất kỳ, ta chỉ cần lấy số dư đã lưu của tháng trước cộng trừ với các giao dịch gần đây là ra. Với ledger tần suất giao dịch khổng lồ thì lưu mỗi tuần. Ở bản thiết kế database trên chúng được lưu trong table statements.

Tổng thể database



Hình 3.1.4: Database cho ứng dụng ví điện tử.

3.2 Mô phỏng dữ liệu giao dịch

Ở mục này ta khảo sát dữ liệu nào sẽ được insert vào database khi có giao dịch. Từ đó có cái nhìn cụ thể về hướng đi của dòng tiền và cách truy vấn số dư.

Nạp tiền

Giả sử Bob nạp vào tài khoản của mình 123 với số tiền \$75 ở chi nhánh nào đó ở Hà Nội. Ở table transactions thêm một record cho giao dịch này. tx_type = Dr tức Debit, ám chỉ chi nhánh ở HN chịu thêm liability từ Bob. Ngầm hiểu tài khoản 123 là phía Credit. tx_type_ext = Dp nghĩa là deposit.

acc_no	ledger_no	time	ref_code	amount	tx_type	tx_type_ext
123	HN Liability	2024-11-20	REF001	75.00	Dr	Dp

Đến lượt ledger của chi nhánh Hà Nội phản ánh sự thay đổi này ở table Ledger-Transactions, nếu Bob nạp bằng tiền mặt thì Ledger Cash sẽ được cập nhật, ngược lại sẽ dùng Ledger Digital. Khi này Ledger Cash là phía Dr, phản ánh tiền mặt từ Bob.

ledger_no	ledger_no_dr	tx_time	amount
HN Liability	HN Cash	2024-11-20	75.00

Rút tiền

Ngược lại với nạp tiền thì rút tiền chỉ là đổi chiều Debit và Credit. Giả sử Bob rút tiền mặt \$25. Để ý tx_type chuyển thành Cr ám chỉ chi nhánh giảm liability đối với Bob và tx_type_code = Wd kí hiệu withdrawal.

acc_no	ledger_no	time	ref_code	amount	tx_type	tx_type_ext
123	HN Liability	2024-11-22	REF002	25.00	Cr	Wd

ledger_no	ledger_no_dr	tx_time	amount
HN Cash	HN Liability	2024-11-22	25.00

Chuyển tiền nội bộ

Giả sử Bob chuyển cho Alice (456) số tiền \$10. Khi này tiền đi trong hệ thống, hay nói cách khác là không có sự tham gia của chi nhánh. Khi này liability sẽ đi qua Central Ledger. Bản chất khi chuyển tiền giữa hai tài khoản chỉ là sự chuyển đổi liability, giảm cho người này và tăng cho người kia. Central Ledger phản ánh tổng liability của công ty khắp các tài khoản của khách hàng và không đổi sau giao dịch này. Để ý dòng thứ nhất (tài khoản 123), với khái niệm debit/credit thì giống với lúc rút tiền, tiền đi khỏi tài khoản Bob (công ty giảm liability đối với Bob). Dòng thứ hai, tài khoản 456 thì giống như trường hợp nạp tiền, tiền đi vào tài khoản Alice (công ty tăng liability đối với Alice.) Kí hiệu AD: adjusting Debit, tức tiền của giảm; AC: adjusting Credit, tức tiền tăng.

acc_no	ledger_no	time	ref_code	amount	tx_type	tx_type_ext
123	Central Liabi	2024-11-25	REF003	10.00	Cr	AD
456	Central Liabi	2024-11-25	REF003	10.00	Dr	AC

3.3 Tính số dư

Như đã đề cập, số dư phải được tính dựa trên giao dịch (derived balance). Tuy nhiên lượng giao dịch phình to mỗi ngày, cho nên để tăng tốc tính, người ta sẽ lưu lại số dư chốt sổ (closing balance) tháng trước, rồi sau đó tổng hợp với các giao dịch tháng này. Số dư tài khoản được tính bằng:

$$\text{Balance} = \text{Closing Balance} + \text{Total Credit (tháng này)} - \text{Total Debit (tháng này)}$$

Với thiết kế database này, chúng ta sẽ cần bảng AccountStatements lấy closing balance sau khi kiểm toán. Còn Credit và Debit có thể được tính trực tiếp từ table AccountTransactions như ta vừa trình bày ở trên, với tiền tài khoản tăng khi tx_type_ext là Deposit hoặc AC; ngược lại nếu nó không là hai cái này thì nó chỉ tiền tài khoản giảm. Dưới đây là mã giả SQL:

```
CREATE VIEW Account_Current_Balance AS
SELECT AccountNo,
       Date = first day of this month
       ASS.ClosingBalance,

       TotalCredit = (
         SELECT SUM(amount)
         FROM AccountTransactions ATX
         WHERE ATX.acc_no = ASS.acc_no
               AND tx_typecode_ext IN ("AC", "Dp")
               AND DateTime between first-date-and-today
       ),
       TotalDebit = (
         SELECT SUM(amount)
         FROM AccountTransactions ATX
         WHERE ATX.acc_no = ASS.acc_no
               AND tx_typecode_ext NOT IN ("AC", "Dp")
               AND DateTime between first-day-and-today
       ),
       CurrentBalance = ClosingBalance + TotalCredit - TotalDebit
FROM AccountStatements ASS
WHERE ASS.Date = First day of this month
```

Cài đặt phần mềm

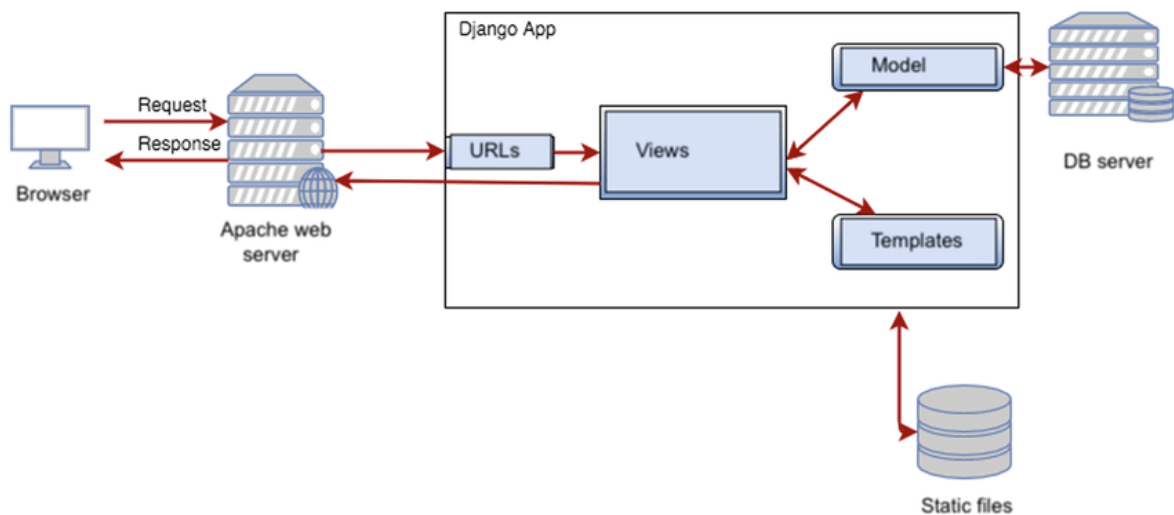
4.1 Công cụ

Ở đề tài này sử dụng Django Framework để lập trình ví điện tử, chạy trên nền web. Phía dưới cơ sở dữ liệu là PostgreSQL. Django sẽ tương tác với cơ sở dữ liệu thông qua các *models*. Các *models* này chính là các table trong database. Truy cập database từ Django sẽ giúp giảm công sức sử dụng query thuần rất phức tạp, ngoài ra còn tránh được SQL-injection (vì không chạy trực tiếp SQL). Có thể hiểu Django là một server với nhiệm vụ render giao diện người dùng, nhận và xử lý dữ liệu từ frontend, rồi giao tiếp với database phía sau.

4.1.1 Django Framework

Django là một framework web mã nguồn mở được viết bằng ngôn ngữ lập trình Python. Django giúp việc phát triển các ứng dụng web trở nên nhanh chóng, dễ dàng và bảo mật hơn nhờ vào bộ công cụ mạnh mẽ, các mô-đun có sẵn và các nguyên lý thiết kế chuẩn. Django được thiết kế theo kiến trúc *Model-View-Template* (MVT), một biến thể của mô hình *Model-View-Controller* (MVC). Trong kiến trúc này, ứng dụng web được chia thành ba phần chính:

- **Model (Mô hình):** Đại diện cho dữ liệu và logic kinh doanh của ứng dụng. Model trong Django là các lớp Python được sử dụng để định nghĩa các bảng trong cơ sở dữ liệu, cùng với các trường và các phương thức liên quan. Django hỗ trợ việc quản lý cơ sở dữ liệu thông qua *ORM* (Object-Relational Mapping), cho phép các lập trình viên làm việc với cơ sở dữ liệu bằng cách sử dụng các đối tượng Python thay vì viết câu lệnh SQL thủ công.
- **View (Giao diện):** Chịu trách nhiệm xử lý các yêu cầu HTTP và trả về các phản hồi phù hợp. Các view trong Django có thể là các hàm hoặc các lớp, thực hiện các tác vụ như xử lý dữ liệu từ yêu cầu, gọi các mô hình và trả về kết quả dưới dạng các template HTML.



Hình 4.1.1: Sơ đồ của một web app Django

- **Template (Mẫu):** Là phần giao diện người dùng của ứng dụng. Template trong Django sử dụng một ngôn ngữ template riêng biệt, cho phép tách biệt logic hiển thị và logic xử lý, giúp việc phát triển giao diện trở nên dễ dàng hơn. Template giúp dễ dàng tái sử dụng mã HTML, cùng với việc truyền dữ liệu từ view đến giao diện người dùng.

Một ứng dụng Django thường bao gồm một số thành phần chính:

- **URL dispatcher (Trình xử lý URL):** Django sử dụng một hệ thống URL mạnh mẽ, cho phép lập trình viên dễ dàng ánh xạ các URL vào các view cụ thể. Điều này giúp tách biệt các phần điều khiển và dễ dàng thay đổi cấu trúc URL mà không ảnh hưởng đến các phần còn lại của ứng dụng.
- **Admin interface (Giao diện quản trị):** Django cung cấp một giao diện quản trị tự động, giúp quản lý các mô hình và dữ liệu trong cơ sở dữ liệu một cách dễ dàng mà không cần phải viết nhiều mã.
- **Middleware (Lớp trung gian):** Các middleware là các lớp xử lý nằm giữa các yêu cầu HTTP và các view. Middleware giúp thực hiện các tác vụ như xác thực, ghi log, và xử lý lỗi.

Tóm lại, Django là một framework mạnh mẽ và dễ sử dụng, cung cấp tất cả các công cụ cần thiết để xây dựng các ứng dụng web hiện đại và bảo mật. Với kiến trúc MVT và các thành phần hỗ trợ mạnh mẽ, Django là lựa chọn phổ biến cho các dự án web từ nhỏ đến lớn.

4.1.2 PostgreSQL

PostgreSQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở, được phát triển và duy trì bởi cộng đồng PostgreSQL Global Development Group. PostgreSQL hỗ trợ nhiều tính năng mạnh mẽ và tiên tiến, giúp quản lý và truy xuất dữ liệu hiệu quả, bảo mật và có thể mở rộng. Đây là một hệ quản trị cơ sở dữ liệu phổ biến, được sử dụng rộng rãi trong các ứng dụng web, hệ thống quản lý dữ liệu lớn và các ứng dụng yêu cầu tính toàn vẹn dữ liệu cao như Ví điện tử.

Một trong những ưu điểm lớn của PostgreSQL là khả năng hỗ trợ các loại dữ liệu phức tạp và các kiểu dữ liệu người dùng tự định nghĩa, giúp dễ dàng mở rộng và tùy biến. Hệ thống này hỗ trợ các tính năng như *ACID* (Atomicity, Consistency, Isolation, Durability), đảm bảo rằng transaction diễn ra một cách chính xác và an toàn.

PostgreSQL cũng cung cấp khả năng đồng thời xử lý nhiều truy vấn, tối ưu hóa hiệu suất với các chỉ mục và kế hoạch truy vấn, giúp tăng tốc các thao tác đọc và ghi dữ liệu. Hệ quản trị cơ sở dữ liệu này hỗ trợ việc mở rộng quy mô dễ dàng, nhờ khả năng phân mảnh dữ liệu (sharding) và hỗ trợ các hệ thống đa máy chủ.

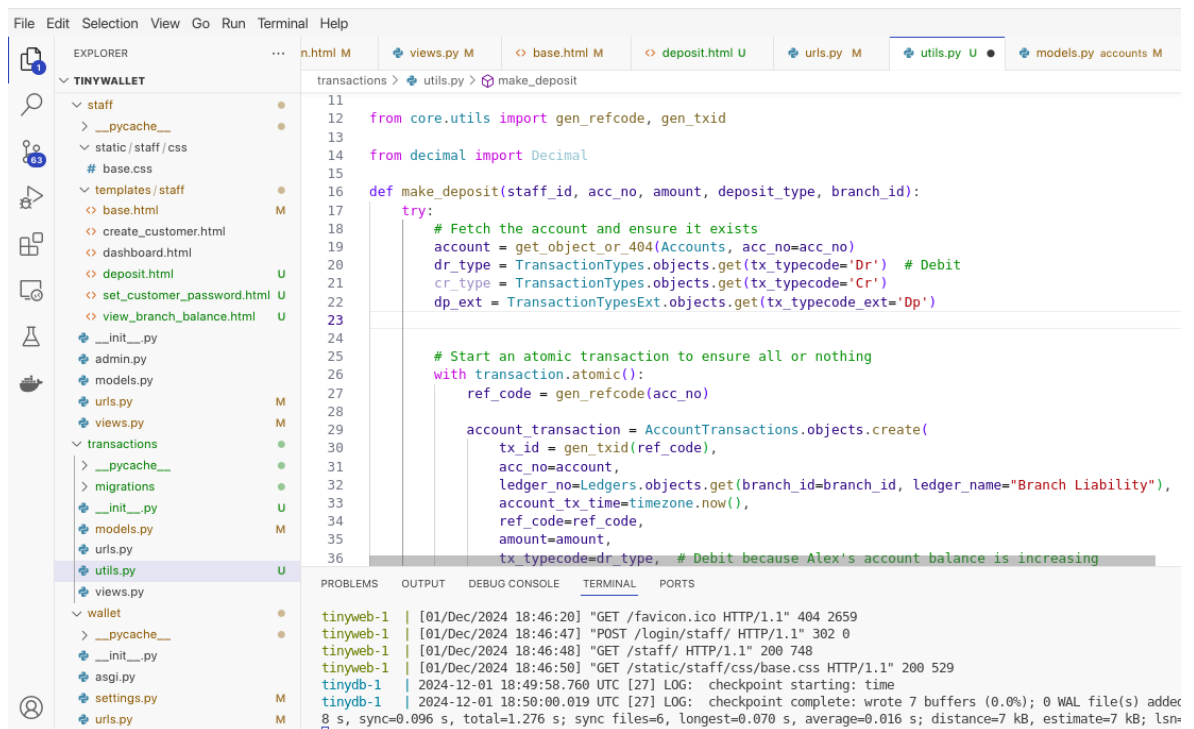
Các tính năng nổi bật của PostgreSQL bao gồm:

- **Hỗ trợ Standard SQL:** PostgreSQL tuân thủ các tiêu chuẩn SQL quốc tế và mở rộng với nhiều tính năng nâng cao.
- **Khả năng mở rộng:** PostgreSQL hỗ trợ các kiểu dữ liệu tự định nghĩa và các tính năng mở rộng qua các plugin, cho phép tích hợp với các công nghệ mới.
- **Bảo mật:** Hệ thống cung cấp các cơ chế bảo mật toàn diện như mã hóa dữ liệu, quản lý quyền truy cập người dùng và hỗ trợ SSL cho các kết nối an toàn.
- **Quản lý Transaction:** Hệ thống hỗ trợ các giao dịch phân tán, khả năng phục hồi sau sự cố và các kỹ thuật quản lý giao dịch như *MVCC* (Multi-Version Concurrency Control).

4.2 Cấu trúc tổng quát của Ví điện tử

Django project, hay Ví điện tử, được chia làm nhiều app con. Trong đó, mỗi app con đảm nhận một phần chức năng tương ứng trong project lớn.

App	Chức năng
Accounts	Quản lý login của user: Staff (nhân viên) và Customers (khách hàng). Khi login thành công sẽ điều hướng URL đến dashboard của từng user.
Staff	Chứa dashboard của nhân viên. App này bao gồm những chức năng mà nhân viên sử dụng: Mở tài khoản (và đặt mật khẩu). Nạp/Rút tiền. Xem số dư (tài sản) của chi nhánh mình đang làm.
Customers	Chứa dashboard của khách hàng. Bao gồm các chức năng: Chuyển tiền nội bộ, xem số dư, xem lịch sử giao dịch.
Transactions	App này xử lý các giao dịch trong hệ thống, tương tác chặt chẽ với CSDL. Bao gồm chuyển tiền nội bộ, nạp/rút tiền, và giao dịch giữa các ledger.
Audits	App này chứa các chức năng kiểm toán. Bao gồm tính số dư chốt sổ (closing balance) cho các ledger và tài khoản.
Core	App này gồm những hàm con, sử dụng nhỏ lẻ ở các app trên, ví dụ như sinh mã giao dịch.



Hình 4.2.2: Môi trường phát triển app (VS Code on Ubuntu)



Staff Dashboard

Hình 4.3.4: Giao diện login của Staff.

29

sẽ lưu lại `staff_id` và `branch_id` (tìm trong database dựa trên khóa) để thuận tiện xác định nhân viên và chi nhánh xử lý giao dịch.

The screenshot shows a web application interface for a staff member. On the left is a blue sidebar titled 'Staff Dashboard' with navigation links: Home, Make Deposit, Make Withdrawal, Branch's Balance, Open Account (highlighted), and Set Customer Password. The main content area is titled 'Open Account' and contains a form with the following fields: National ID (201290002000), First Name (Minh), Last Name (Nhật), Sex (Male), Phone Number (0984111111), Province (Hồ Chí Minh), and District (Thủ Đức). There is a 'Preview' button below the form. Below the form is a section titled 'Account Info' displaying: Account: 774256305979, Owner: Minh Nhật, Created At: 12/2/2024, 2:07:15 AM, Staff ID: hno12345, and Branch ID: HNO00. A 'Submit' button is at the bottom of this section.

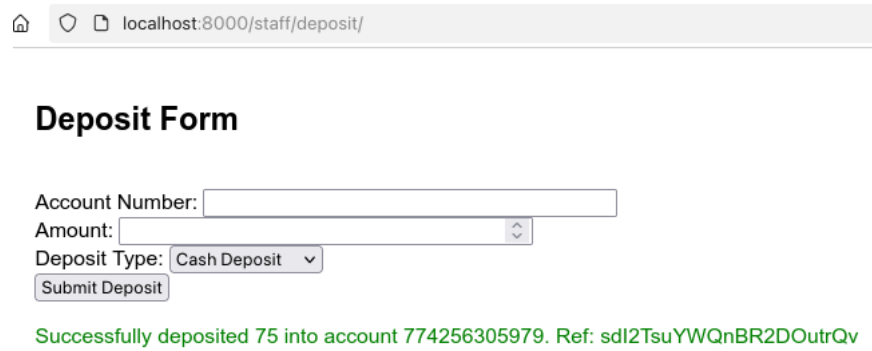
Hình 4.3.5: Mở tài khoản ở chi nhánh Hà Nội.

Nhân viên sử dụng chức năng mở tài khoản. Sau khi điền thông tin khách hàng ở form trên, hệ thống sẽ tạo số tài khoản và truy vấn thông tin nhân viên và địa điểm tạo tài khoản này. Sau khi preview, nhấn Submit và app sẽ chính thức lưu vào database.

Sau khi tạo tài khoản, nhân viên tiến hành đặt mật khẩu và thực hiện nạp tiền.

The screenshot shows the 'Set Customer Password' form. At the top, the URL is `localhost:8000/staff/set-customer-password/`. Below the title, there is a green message: 'Password set successfully'. The form contains two input fields: 'National ID' and 'New Password'. Below these fields is a 'Set Password' button.

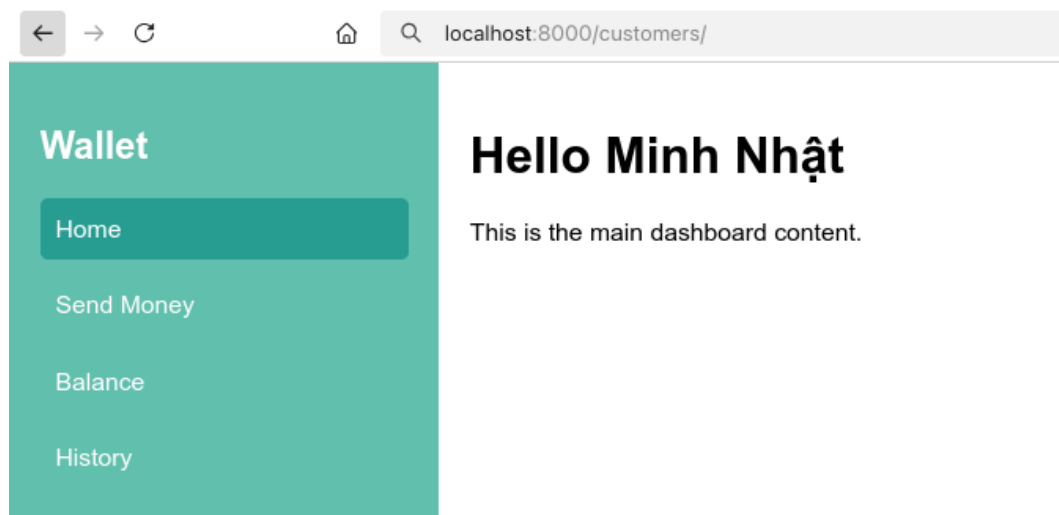
Hình 4.3.6: Đặt mật khẩu cho tài khoản mới.



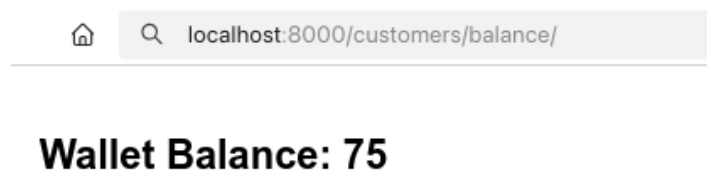
The screenshot shows a web browser window with the address bar displaying 'localhost:8000/staff/deposit/'. The page title is 'Deposit Form'. It contains a form with the following fields: 'Account Number:' with a text input field, 'Amount:' with a text input field and a small up/down arrow icon, 'Deposit Type:' with a dropdown menu showing 'Cash Deposit', and a 'Submit Deposit' button. Below the form, a green message states: 'Successfully deposited 75 into account 774256305979. Ref: sdI2TsuYWQnBR2DOutrQv'.

Hình 4.3.7: Nạp tiền vào số tài khoản thành công.

Customer Wallet



Hình 4.3.8: Giao diện sau khi người dùng đăng nhập.



Hình 4.3.9: Xem số dư sau khi vừa nạp tiền.

localhost:8000/customers/transfer

Send Money

Recipient:

Amount:

Successfully sent 50 to account 243633866555. Ref: 8e9u7EQ6cBzWUzXVA9KSO

Hình 4.3.10: Thực hiện chuyển tiền sang cho tài khoản ví điện tử khác.

← → ↻

🏠 🔍 localhost:8000/history/

Wallet

- Home
- Send Money
- Balance
- History**

History

Send 50 to 243633866555
Ref: 8e9u7EQ6cBzWUzXVA9KSO
at 2024-12-01 19:27:06

Deposited 75 from hno12345
Ref: sdI2TsuYWQnBR2DOutrQv
at 2024-12-01 19:20:36

Hình 4.3.11: Xem lịch sử giao dịch.

Kết luận

Đề tài đã tìm hiểu tương đối hoàn chỉnh về Ví điện tử, đặc biệt từ khía cạnh của kiểm toán, bởi vì hệ thống tài chính yêu cầu chính xác và toàn vẹn các giao dịch. Do đó cơ sở dữ liệu được thiết kế lấy kiểm toán làm trung tâm. Cũng vì tối ưu về mặt chính xác mà đánh đổi hiệu năng truy vấn, chưa được đề cập trong đây.

Ở phía lập trình, đề tài có xem xét bảo mật và một tính năng thú vị là kỹ thuật sinh mã giao dịch. Sinh viên đã cài đặt thử ví điện tử đơn giản bằng Django Framework và PostgreSQL, cho ra kết quả thử nghiệm đáp ứng được nhu cầu cơ bản cần đạt. Tuy nhiên vì giới hạn kiến thức về web nên chỉ dừng ở mức đơn sơ.

Hệ thống Ví điện tử trong đề tài này có thể được mở rộng thêm ở nhiều phương diện. Thứ nhất nó có thể kết nối với ngân hàng hay các điểm thanh toán bằng cách dùng ledger của những chỗ đó. Thứ hai về mặt bảo mật ta cần thêm các cơ chế xác nhận (OTP), và làm sao để khóa balance khi đang giao dịch để ngăn cản hiện tượng một số tiền được chuyển đi hai lần (double spending).

Tài liệu tham khảo

Books

- [1] T. S. Grewal. Double Entry Book Keeping: Financial Accounting Textbook for CBSE Class 11. S. Chand Publishing, 2022. As per 2022-23 syllabus.
- [2] Michael J. Hernandez. Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design. Addison-Wesley Professional, 2013.
- [3] Jagdish Kothari and Elisabetta Barone. Financial Accounting: An International Approach. Pearson, 2006.
- [4] Marshall B. Romney and Paul John Steinbart. Accounting Information Systems. Pearson, 2017.
- [5] Abraham Silberschatz, Henry F. Korth, and S. Sudarshan. Database System Concepts, 7th edition. McGraw-Hill Education, 2019.
- [6] William S. Vincent. Django for Beginners. WelcomeToCode, 2022.

Websites

- [7] Harold Averkamp. Bookkeeping explanation - double entry accounting, 2013. [⟨https://www.accountingcoach.com/bookkeeping/explanation/3⟩](https://www.accountingcoach.com/bookkeeping/explanation/3). Accessed: 01-11-2024.
- [8] Shayne Parikh. Sharding ids at instagram, 2012. [⟨https://instagram-engineering.com/sharding-ids-at-instagram-1cf5a71e5a5c⟩](https://instagram-engineering.com/sharding-ids-at-instagram-1cf5a71e5a5c). Accessed: 05-11-2024.
- [9] Wall Street Prep. Double entry accounting: Everything you need to know, 2024. [⟨https://www.wallstreetprep.com/knowledge/double-entry-accounting/⟩](https://www.wallstreetprep.com/knowledge/double-entry-accounting/). Accessed: 01-11-2024.