BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM

ĐỒ ÁN MÔN HỌC

THU THẬP VÀ PHÂN TÍCH DỮ LIỆU GIAO DỊCH ETH TỪ WEBSITE HTTPS://ETHERSCAN.IO SỬ DỤNG CÔNG CỤ MÃ NGUỒN MỞ BEAUTIFULSOUP VÀ MONGODB

Ngành: **Khoa học Dữ liệu**

Môn học: **Mã nguồn mở trong Khoa học Dữ liệu**

Giảng viên hướng dẫn: ThS. Lê Nhật Tùng

Sinh viên thực hiện: 2286400881 – Nguyễn Phi Long

2286400039 – Chu Quốc Trung

2286400018–Châu Nguyễn Phương Nam

Lớp: 22DKHA1

TP. Hồ Chí Minh, 2024

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

TP.HCM, Ngày…..tháng…..năm 2024

Giảng viên hướng dẫn

LỜI CAM ĐOAN

Chúng tôi xin cam đoan mọi thông tin và nghiên cứu được trình bày trong bài báo cáo này, dưới sự hướng dẫn khoa học của ThS. Lê Nhật Tùng là trung thực, hoàn toàn khách quan được thu thập và phân tích một cách cẩn thận dựa trên nguồn chính thống và trích dẫn theo đúng quy định.

Chúng tôi cam đoan bài báo cáo không có bất kỳ sao chép từ bất kỳ nguồn tài liệu nào mà không được trích dẫn đầy đủ hoặc sử dụng những thông tin không rõ nguồn gốc.

Toàn bộ nội dung trong bài báo cáo là công trình nghiên cứu của chúng tôi chưa từng được công bố ở bất kỳ nơi nào, dưới bất kỳ hình thức nào. Chúng tôi cam đoan rằng đã tuân thủ quy định của môn học về việc tham khảo và sử dụng các công cụ nghiên cứu phục vụ cho bài báo cáo.

TP.HCM, Ngày 30 tháng 10 năm 2024

Nhóm sinh viên

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT VÀ TỪ KHÓA

Ethereum	Một nền tảng điện toán có tính chất phân tán, công cộng, mã nguồn mở dựa trên công nghệ Blockchain. Nó có tính năng hợp đồng thông minh, tạo thuận lợi cho các thỏa thuận hợp đồng trực tuyến.[1]
Etherscan	Trang web dùng để tra cứu và theo dõi mọi hoạt động trên mạng Etherscan.
Blockchain	Là công nghệ lưu trữ dữ liệu phân tán.
ETH	Đơn vị tiền tệ của mạng Ethereum, được sử dụng để thanh toán phí giao dịch và tương tác với các ứng dụng.
BeautifulSoup	Một gói trong Python dùng để phân tích cú pháp các tài liệu HTML và XML, bao gồm cả những tài liệu có đánh dấu không đúng định dạng.[1]
dApps	decentralized Applications – Các ứng dụng phi tập trung chạy trên nền tảng blockchain, không bị kiểm soát bởi bất kỳ cá nhân hay tổ chức nào.
HTTP	HyperText Transfer Protocol - Giao thức truyền tải siêu văn bản, dùng để truyền dữ liệu giữa trình duyệt và máy chủ web.
HTML	HyperText Markup Language - Ngôn ngữ đánh dấu siêu văn bản, dùng để tạo cấu trúc và nội dung cho trang web.
URL	Uniform Resource Locator - Địa chỉ web duy nhất dẫn đến một tài nguyên trên internet.
JS	JavaScript - Ngôn ngữ lập trình dùng để tạo các trang web tương tác và năng động.
CSS	Cascading Style Sheets - Ngôn ngữ định kiểu cho các trang web, dùng để thiết kế giao diện và bố cục.
XML	eXtensible Markup Language - Ngôn ngữ đánh dấu mở rộng, dùng để lưu trữ và truyền tải dữ liệu dưới dạng có cấu trúc.

JSON	JavaScript Object Notation - Định dạng dữ liệu nhẹ, dễ đọc và ghi, thường được dùng để truyền tải dữ liệu giữa máy chủ và trình duyệt.
Wei	Đơn vị nhỏ nhất của Ether (ETH) trong hệ thống Ethereum.
Gwei	Đơn vị đo lường phổ biến của Ether, tương đương 1 tỷ Wei (10^9 Wei), thường được dùng để tính phí gas.
BSON	Binary JSON - Định dạng mã hóa nhị phân của JSON, dùng để lưu trữ và truyền tải dữ liệu, phổ biến trong MongoDB.

MỤC LỤC

DANH SÁCH BẢNG

DANH SÁCH HÌNH

CHƯƠNG 1: TỔNG QUAN

1.1 Giới thiệu đề tài

[2]Ethereum là nền tảng điện toán phân tán, mã nguồn mở dựa trên công nghệ chuỗi khối (blockchain) có khả năng thực thi hợp đồng thông minh (smart contract) - tức là điều khoản được ghi trong hợp đồng sẽ được thực thi một cách tự động khi các điều kiện trước đó được thỏa mãn, không ai có thể can thiệp vào. Sự phát triển mạnh mẽ của hệ sinh thái Ethereum đã tạo ra một lượng lớn giao dịch mỗi ngày.
[2]Ethereum có đồng tiền điện tử gốc là ether (ETH). Ether là một loại cryptocurrency (tiền điện tử kỹ thuật số) được xây dựng vào năm 2013 bởi Vitalik Buterin, thường được gọi là cryptocurrency 2.0.

[2]Ether rất cần thiết trong việc thực hiện hầu hết mọi hoạt động trên Ethereum và khi nó được sử dụng để thực thi các liên hệ thông minh trên mạng, nó thường được gọi là "gas".

Việc thu thập và phân tích dữ liệu giao dịch trên mạng Ethereum thông qua công cụ Etherscan giúp hiểu rõ hơn về cách thức hoạt động của mạng, các yếu tố ảnh hưởng đến phí giao dịch, xu hướng sử dụng các ứng dụng phi tập trung và hành vi của người dùng. Đề tài này tập trung vào việc thu thập, xử lý và phân tích dữ liệu giao dịch từ Etherscan để cung cấp thông tin hữu ích cho các nhà phát triển, các nhà đầu tư và các bên liên quan khác.

1.2 Nhiệm vụ của đề tài

1.2.1 Tính cấp thiết của đề tài

Sự tăng trưởng nhanh chóng của các ứng dụng dựa trên Ethereum, đặc biệt là trong lĩnh vực tài chính phi tập trung (DeFi) và tài sản số không thay thế (NFTs), khiến lượng giao dịch trên mạng lưới tăng lên đáng kể. Điều này dẫn đến các vấn đề về phí giao dịch cao và thời gian xử lý giao dịch lâu. Việc thu thập và phân tích dữ liệu giao dịch là cần thiết để đánh giá hiệu suất của mạng, từ đó tối ưu hóa hoạt động và dự báo các xu hướng trong tương lai.

1.2.2 Ý nghĩa khoa học và thực tiễn của đề tài

Ý nghĩa khoa học: Đề tài sẽ cung cấp phương pháp thu thập và phân tích dữ liệu giao dịch blockchain một cách có hệ thống, tạo tiền đề cho các nghiên cứu liên quan

đến việc tối ưu hóa phí giao dịch, hiệu suất mạng, và hành vi giao dịch trên mạng phi tập trung.

Ý nghĩa thực tiễn: Kết quả từ đề tài có thể được ứng dụng để cải thiện trải nghiệm người dùng trên các ứng dụng Ethereum, giúp người dùng tối ưu hóa giao dịch và cung cấp thông tin quan trọng cho các nhà phát triển về hiệu suất hệ thống.

1.3 Mục tiêu

1.3.1 Mục tiêu tổng quát

Mục tiêu của đề tài là thu thập và phân tích dữ liệu giao dịch trên Etherscan để cung cấp những thông tin hữu ích về hiệu suất mạng Ethereum và hành vi người dùng, từ đó đưa ra các khuyến nghị giúp tối ưu hóa việc sử dụng mạng lưới này.

1.3.2 Mục tiêu cụ thể

Thu thập dữ liệu từ Etherscan, bao gồm các thông tin như địa chỉ ví, lượng ETH giao dịch, phí giao dịch và thời gian thực hiện.

Phân tích các yếu tố như tổng lượng ETH giao dịch trong ngày, phí giao dịch trung bình, khối có lượng giao dịch lớn nhất và tỷ lệ giao dịch trên các sàn giao dịch.

Đánh giá xu hướng và hành vi giao dịch của người dùng trong các khoảng thời gian khác nhau.

1.4 Đối tượng và phạm vi

1.4.1 Đối tượng

Đối tượng nghiên cứu của đề tài là các giao dịch trên mạng lưới Ethereum, được ghi lại và truy xuất thông qua trình khám phá blockchain Etherscan.

1.4.2 Phạm vi

Phạm vi nghiên cứu tập trung vào việc thu thập và phân tích các giao dịch trên mạng Ethereum trong một khoảng thời gian cụ thể. Các giao dịch bao gồm việc gửi, nhận ETH và các token khác, phí giao dịch (gas) và các thông số liên quan đến hợp đồng thông minh.

1.5 Phương pháp nghiên cứu

1.5.1 Phương pháp nghiên cứu sơ bộ

Nghiên cứu sơ bộ bao gồm việc khảo sát và tìm hiểu các phương pháp thu thập dữ liệu từ blockchain nói chung và từ Etherscan nói riêng, sử dụng các công cụ như BeautifulSoup 4 và API của Etherscan.

1.5.2 Phương pháp nghiên cứu tài liệu

Phương pháp này bao gồm việc thu thập các tài liệu về công nghệ blockchain, Ethereum và các phương pháp phân tích dữ liệu trên blockchain. Các tài liệu khoa học và báo cáo liên quan đến Ethereum, dApps, và giao dịch trên mạng lưới sẽ được tham khảo.

1.5.3 Phương pháp thực nghiệm

Thực hiện thu thập dữ liệu trực tiếp từ Etherscan bằng cách sử dụng BeautifulSoup để trích xuất dữ liệu giao dịch từ trang web và lưu trữ vào trong MongoDB. Sau đó, tiến hành các truy vấn và phân tích dữ liệu đã thu thập vào bằng MongoDB để đưa ra kết quả.

1.5.4 Phương pháp đánh giá

Dữ liệu sau khi thu thập và phân tích sẽ được đánh giá dựa trên các chỉ số như tổng lượng ETH giao dịch, phí giao dịch trung bình, và số lượng giao dịch qua các tháng/năm. Kết quả này sẽ được so sánh với các tài liệu tham khảo để đánh giá tính hiệu quả của phương pháp thu thập và phân tích.

1.6 Những đóng góp nghiên cứu của đề tài

Đề tài mang lại các đóng góp cụ thể như:

Xây dựng phương pháp thu thập dữ liệu tự động từ Etherscan một cách hiệu quả, tiết kiệm thời gian và tài nguyên.

Cung cấp phân tích chi tiết về hiệu suất của mạng Ethereum và hành vi giao dịch, từ đó giúp tối ưu hóa việc sử dụng mạng lưới.

Góp phần vào các nghiên cứu về blockchain và tài chính phi tập trung, hỗ trợ việc phát triển các ứng dụng phi tập trung trên Ethereum.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Công nghệ Web Scraping

2.1.1 Khái niệm Web Scraping

Web Scraping là gì?

Web Scraping là quá trình tự động thu thập dữ liệu từ các trang web công cộng và lưu trữ dữ liệu để phân tích hoặc sử dụng theo nhiều cách khác. Quá trình thu thập thường được thực hiện bằng cách sử dụng các chương trình máy tính hoặc các con bot để trích xuất thông tin từ trang web theo một cấu trúc được xác định trước.[3]

Dữ liệu từ Web Scraping được dùng để nghiên cứu thị trường, giám sát giá cả, phân tích thông tin. Bên cạnh đó, khi triển khai thu thập dữ liệu bằng Web Scraping cần phải tuân thủ các quy định và điều kiện sử dụng hoặc "chấp thuận" các yêu cầu từ trang web. Nếu không, có thể bị vi phạm luật bản quyền hoặc chính sách của trang web.

Cách hoạt động của Web Scraping

- Đầu tiên người dùng cần cung cấp URL của website cần thu thập cho các web scraper. Sau đó, web scraper sẽ load toàn bộ code HTML hoặc JS và CSS của trang web.
- Người dùng cần lựa chọn những trường dữ liệu cụ thể mà họ muốn để các scraper tiến hành duyệt qua và lấy dữ liệu. Các thông tin về trường dữ liệu có thể được cấu hình trước, nhưng thường người dùng phải chọn thủ công cho mỗi website vì cấu trúc của từng website không giống nhau.
- Cuối cùng, web scraper sẽ xuất ra tất cả những dữ liệu đã thu thập được thành định dạng mong muốn của người dùng.

2.2 BeautifulSoup

BeautifulSoup là một thư viện Python mã nguồn mở để phân tích cú pháp HTML và XML. BeautifulSoup giúp các nhà phát triển trích xuất các yếu tố cụ thể từ một trang nguồn, chẳng hạn như danh sách hình ảnh, thông tin cần thiết hoặc video. [4]

BeautifulSoup cần được sử dụng kết hợp với các thư viện khác (ví dụ: requests) để gửi yêu cầu HTTP và lấy nội dung web. Nó sử dụng các thẻ, nội dung văn bản và thuộc tính làm tiêu chí tìm kiếm và cung cấp một cách đơn giản, linh hoạt và trực quan để trích xuất dữ liệu từ các trang web, giúp điều hướng và tìm kiếm HTML dễ dàng hơn.

2.3 Blockchain và Mạng lưới Ethereum

2.3.1 Khái niệm Blockchain

Công nghệ Blockchain là một cơ chế cơ sở dữ liệu tiên tiến cho phép chia sẻ thông tin minh bạch trong một mạng lưới kinh doanh. Cơ sở dữ liệu chuỗi khối lưu trữ dữ liệu trong các khối được liên kết với nhau trong một chuỗi. Dữ liệu có sự nhất quán theo trình tự thời gian vì người dùng không thể xóa hoặc sửa đổi chuỗi mà không có sự đồng thuận từ mạng lưới. [5]

Công nghệ Blockchain sử dụng cơ chế đồng thuận để hoạt động. Do đó các nút trên blockchain phải đạt được sự đồng thuận về một phần dữ liệu mới trước khi nó được thêm vào một khối. Tính bất biến của công nghệ chuỗi là một trong những thuộc tính nổi bật nhất của nó.

2.3.2 Ethereum

Ethereum là một nền tảng blockchain mã nguồn mở và phi tập trung, được ra mắt vào 2015 bởi Vitalik Buterin. Tương tự như Bitcoin nhưng khác biệt ở khả năng xây dựng và triển khai "hợp đồng thông minh" và "ứng dụng phi tập trung (DApps)".

- Cấu trúc và cơ chế hoạt động:
    - Cấu trúc blockchain: Cũng giống như các blockchain khác, Ethereum sử dụng các khối (blocks) để lưu trữ các thông tin giao dịch. Mỗi khối gồm tập hợp các giao dịch và được liên kết với khối trước bằng một mã hash để tạo nên một chuỗi các khối liên tục.
    - Ethereum Virtual Machine (EVM): Đây là thành phần cốt lõi của Ethereum, cho phép thực thi các đoạn mã được viết bằng nhiều ngôn ngữ lập trình khác nhau, ví dụ như Solidity. EVM giúp xử lý các logic hợp đồng thông minh trên toàn bộ mạng lưới.
- Hợp đồng thông minh:

  Hợp đồng thông minh là những đoạn mã tự động thực thi ở trên blockchain, đảm bảo tính bảo mật và minh bạch. Điều này cho phép tạo ra các ứng dụng tài chính phi tập trung, hệ thống tín nhiệm, và nhiều loại hình ứng dụng khác.
- Ứng dụng phi tập trung (DApps):

  DApps là các ứng dụng hoạt động trên một mạng lưới phân tán thay vì một máy chủ tập trung. DApps được xây dựng trên nền tảng Ethereum bằng các hợp đồng thông minh. [6]
- Tiền mã hóa Ether (ETH):

Ether là đồng tiền điện tử của Ethereum, dùng để trả phí giao dịch và là động lực kinh tế chính của mạng lưới cũng như thúc đẩy hệ sinh thái kỹ thuật số này. [7]

- Proof-of-Stake và Ethereum 2.0:

  Kể từ Ethereum 2.0, mạng lưới đã chuyển sang cơ chế Proof-of-Stake (PoS) để tăng cường tính hiệu quả và bảo mật, đồng thời giảm tiêu thụ năng lượng cũng như khả năng mở rộng cao hơn. [8]

2.4 Cơ sở dữ liệu phân tán và MongoDB

2.4.1 Cơ sở dữ liệu phân tán

Là một hệ thống cơ sở dữ liệu mà dữ liệu được phân phối trên nhiều cơ sở dữ liệu vật lý, máy chủ, trung tâm dữ liệu hoặc thậm chí là các mạng riêng biệt. . Các hệ thống quản lý cơ sở dữ liệu phân tán có khả năng phục hồi tốt hơn, cung cấp độ trễ thấp hơn và bảo vệ dữ liệu hiệu quả hơn. [9]

- Đặc điểm chính của cơ sở dữ liệu phân tán:
  - Phân phối dữ liệu: Dữ liệu được lưu trữ trên nhiều máy chủ (nodes) cũng như có thể ở nhiều vị trí địa lý khác nhau.
  - Tính đồng bộ (Replication): Dữ liệu thường được sao chép giữa các nút để bảo đảm tính toàn vẹn và khả năng khôi phục khi có sự cố.
  - Khả năng mở rộng (Scalability): Hệ thống dễ dàng mở rộng quy mô bằng cách thêm nhiều máy chủ vào mạng lưới mà không ảnh hưởng đến hiệu suất.
  - Chịu lỗi (Fault Tolerance): Trong trường hợp một hoặc nhiều máy chủ gặp sự cố, hệ thống vẫn có thể tiếp tục hoạt động bình thường nhờ cơ chế sao chép dữ liệu.

2.4.2 MongoDB

Là một cơ sở dữ liệu NoSQL mã nguồn mở, là một database hướng tài liệu (document-oriented). MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ các các kích cỡ và các document khác nhau. Các dữ liệu được lưu trữ trong document kiểu JSON nên truy vấn sẽ rất nhanh. [10]

- Các đặc điểm chính của MongoDB:
  - Document-based Storage: Dữ liệu được lưu dưới dạng các tài liệu JSON (hoặc BSON), giúp dễ dàng mô hình hóa các dữ liệu phức tạp.

- Tính năng phân tán (Sharding): MongoDB hỗ trợ sharding, cho phép phân chia dữ liệu thành các phần nhỏ hơn và lưu trữ trên nhiều máy chủ, giúp tăng cường khả năng mở rộng và truy vấn.
- Khả năng mở rộng và phân phối: Dễ dàng mở rộng hệ thống bằng cách thêm các shard (phân mảnh dữ liệu) hoặc replicas (bản sao dữ liệu).
- Replication: MongoDB sử dụng Replica Sets để cung cấp tính năng sao lưu và khôi phục dữ liệu khi xảy ra sự cố.

2.5 Truy vấn dữ liệu và phân tích

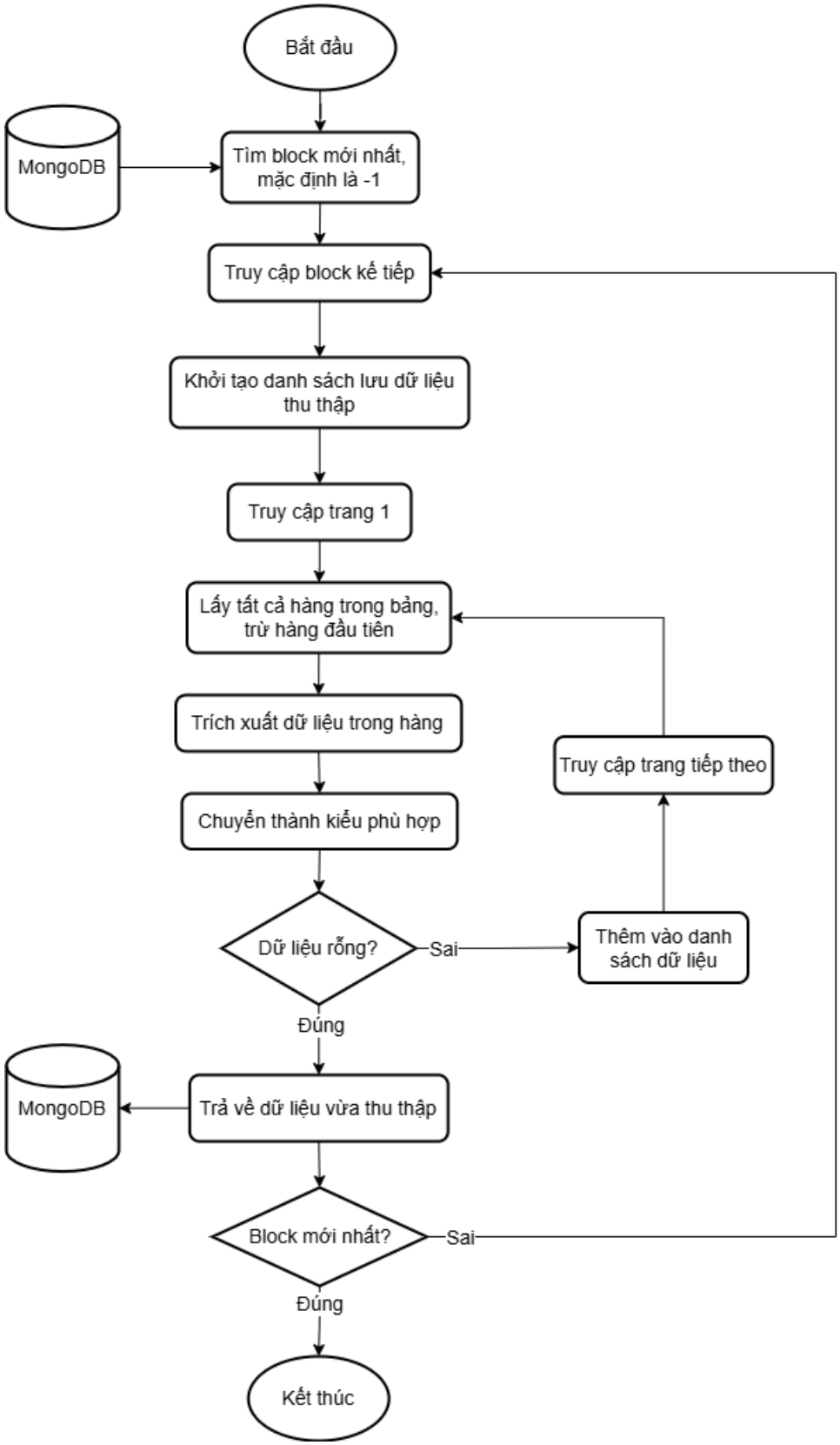
2.5.1 Truy vấn bằng MongoDB

Là một hệ quản trị cơ sở dữ liệu NoSQL linh hoạt, MongoDB cung cấp cú pháp truy vấn dữ liệu phong phú và mạnh mẽ. Cho phép thực hiện dễ dàng các thao tác như tìm kiếm, sắp xếp hay lọc các dữ liệu giao dịch phức tạp đã thu thập từ Etherscan.

2.5.2 Phân tích dữ liệu

Dữ liệu thu thập từ Ethereum có thể được phân tích để tìm ra xu hướng, hoặc để trả lời các câu hỏi cụ thể về các giao dịch như phí giao dịch, khối lượng giao dịch, phương thức giao dịch là gì…

CHƯƠNG 3: PHƯƠNG PHÁP THỰC NGHIỆM



Hình 3.1: Sơ đồ thu thập, lưu trữ và cập nhật dữ liệu

3.1 Phương pháp thu thập

Chúng tôi đã tiến hành thu thập dữ liệu bằng cách gửi yêu cầu HTTP thông qua thư viện urllib tới máy chủ của Etherscan. Mục tiêu là tải trang giao dịch dựa trên số khối và trích xuất dữ liệu từ các giao dịch chứa trong khối đó. Quá trình này được thực hiện theo các bước sau:

3.1.1 Tạo URL và gửi yêu cầu

URL truy cập tới trang số "page" của khối "block_number" được tạo ra bằng cách kết hợp các thông số block và page để tạo thành đường dẫn đầy đủ đến máy chủ Etherscan. Công thức URL như sau:

https://etherscan.io/txs?block={block_number}&ps=100&p={page}

Trong đó:

- https://etherscan.io/ là địa chỉ trang web Etherscan.
- block đại diện cho số khối, với tham số là "block_number" xác định khối cần truy xuất.
- ps quy định số lượng giao dịch tối đa hiển thị trên mỗi trang (ở đây là 100 giao dịch).
- p là số trang, với tham số "page" xác định trang cụ thể trong block đó.
- Sau khi có URL, chúng tôi sử dụng urllib.request.Request và cấu hình User-Agent để mô phỏng truy cập từ trình duyệt thực nhằm tránh bị chặn bởi Etherscan.

3.1.2 Phân tích HTML

Khi nhận được phản hồi HTML từ Etherscan, chúng tôi dùng BeautifulSoup để xử lý và phân tích cấu trúc trang. Từng dòng giao dịch sẽ được trích xuất từ bảng thông tin giao dịch trên trang web, lấy chi tiết từ các cột tương ứng trong mỗi hàng giao dịch, đảm bảo thu thập đầy đủ các thông tin cần thiết cho từng giao dịch.

3.1.3 Trích xuất dữ liệu giao dịch

Mỗi giao dịch bao gồm các chi tiết như: mã hash của giao dịch (transaction_hash), phương thức giao dịch (method), số khối (block), thời gian (age), địa chỉ người gửi (sender_address), địa chỉ người nhận (recipient_address), số lượng ETH (amount) và phí giao dịch (txn_fee).

Các giá trị trên được đưa về kiểu dữ liệu phù hợp. Đặc biệt, số lượng ETH được chuyển đổi từ nhiều đơn vị khác nhau (ETH, gwei, wei) thành đơn vị chuẩn là ETH.

3.1.4 Xử lý lỗi và trả về danh sách giao dịch trong khối:

Nếu xảy ra lỗi trong quá trình thu thập dữ liệu, chương trình sẽ xử lý bằng cách trả về một danh sách rỗng và in chi tiết lỗi ra màn hình để dễ dàng theo dõi và khắc phục.

Ngược lại, danh sách chứa các giao dịch trong khối sẽ được trả về, kèm theo thông báo cho biết quá trình thu thập dữ liệu thành công.

3.2 Mô tả dữ liệu

Dữ liệu ban đầu thu thập được bao gồm thông tin về khoảng 5.000.000 giao dịch đầu tiên của năm 2024. Mỗi giao dịch gồm 8 trường dữ liệu chứa các thông tin sau:

Tên biến	**Mô tả**	**Kiểu dữ liệu**
transaction_hash	Mã băm của giao dịch	String
method	Phương thức sử dụng trong giao dịch	String
block	Số khối liên kết với giao dịch	Integer
age	Thời gian của giao dịch	Datetime
from	Địa chỉ của người gửi	String
to	Địa chỉ của người nhận	String
amount	Số lượng ETH được chuyển	Float
txn_fee	Phí giao dịch	Float

Bảng 3.1: Bảng mô tả dữ liệu

3.3 Phương pháp lưu trữ

Sau khi thu thập và cấu trúc dữ liệu giao dịch, bước tiếp theo là lưu trữ vào cơ sở dữ liệu MongoDB thông qua MongoClient. Chi tiết thực hiện như sau:

3.3.1 Kết nối MongoDB

- Tạo client bằng MongoClient thuộc thư viện pymongo để thiết lập kết nối tới MongoDB, với thông số kết nối là URL của MongoDB server.
- Kết nối tới cơ sở dữ liệu ether_db trên MongoDB, tạo điều kiện cho việc truy xuất và lưu trữ dữ liệu.
- Tạo mới hoặc truy cập collection transactions trong cơ sở dữ liệu ether_db. Nếu collection chưa tồn tại, MongoDB sẽ tự động tạo mới khi có dữ liệu đầu tiên được thêm vào.

3.3.2 Thêm dữ liệu vào cơ sở dữ liệu

Gọi hàm thu thập dữ liệu giao dịch cho từng khối cần xử lý. Mỗi lần hàm này trả về dữ liệu của một khối, chương trình sẽ thêm dữ liệu đó vào cơ sở dữ liệu thông qua phương thức insert_many() của collection transactions.

3.4 Phương pháp cập nhật

3.4.1 Truy vấn block mới nhất đã lưu

Để xác định số khối của block mới nhất đã lưu trong cơ sở dữ liệu, chương trình sẽ truy vấn để tìm số khối lớn nhất trong collection transactions thuộc cơ sở dữ liệu ether_db. Đây là số khối của block mới nhất hiện có trong cơ sở dữ liệu, làm cơ sở để thu thập dữ liệu giao dịch từ các block tiếp theo.

3.4.2 Thu thập và lưu trữ dữ liệu mới

Dựa trên số khối của block mới nhất vừa truy vấn, hệ thống sẽ tự động gửi yêu cầu tải về các giao dịch từ những block tiếp theo. Các giao dịch này sẽ được xử lý và lưu trữ trong MongoDB, đảm bảo rằng mọi dữ liệu mới đều được cập nhật kịp thời.

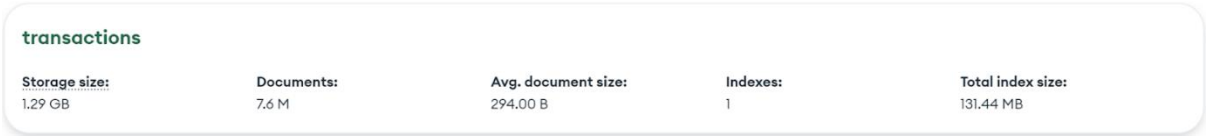
3.4.3 Lặp lại và cập nhật định kỳ

Cứ cách một thời gian chương trình sẽ được chạy lại thủ công hoặc theo lịch tự động. Việc này đảm bảo rằng dữ liệu luôn được cập nhật một cách chính xác và liên tục, duy trì tính đầy đủ và mới nhất của thông tin giao dịch trong cơ sở dữ liệu MongoDB.

CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM

4.1 Thu thập dữ liệu

Sau hơn 36 giờ chạy, chúng tôi đã thu thập được dữ liệu giao dịch của gần 50.000 khối trên mạng Ethereum với khoảng 7.6 triệu giao dịch trong tuần đầu tiên của năm 2024. Kích thước lưu trữ 1.29GB với trung bình mỗi giao dịch là 294 B.



Hình 4.1: Dữ liệu giao dịch được lưu vào MongoDB sau khi thu thập

4.2 Truy vấn dữ liệu

4.2.1 Tổng lượng ETH giao dịch trong tuần đầu tiên của tháng 1, năm 2024

Kết quả truy vấn tổng lượng giao dịch ETH trong tuần đầu tiên của tháng 1, năm 2024 được ghi nhận như sau:

Ngày	**Tổng lượng ETH giao dịch**
1/1/2024	945,167.10 ETH
2/1/2024	1,486,027.34 ETH
3/1/2024	1,676,843.48 ETH
4/1/2024	1,426,742.36 ETH
5/1/2024	1,515,975.50 ETH
6/1/2024	751,550.16 ETH
7/1/2024	677,537.83 ETH

Bảng 4.1: Tổng lượng giao dịch ETH theo từng ngày trong tuần đầu tiên của tháng 1, năm 2024

Tổng lượng giao dịch ETH trong tuần đầu tiên là 8,479,843.77 ETH.

Nhận xét:

- Ngày 3/1/2024 ghi nhận lượng giao dịch cao nhất với 1,676,843.48 ETH.
- Ngày 7/1/2024 ghi nhận lượng giao dịch thấp nhất với 677,537.83 ETH.
- Lượng giao dịch có xu hướng tập trung vào giữa tuần và giảm mạnh vào cuối tuần (các ngày 6 và 7/1).

4.2.2 Phí giao dịch trung bình trong tuần đầu tiên của tháng 1, năm 2024

Kết quả truy vấn về phí giao dịch trung bình mỗi ngày trong tuần đầu tiên của tháng 1 năm 2024:

Ngày	**Phí giao dịch trung bình**
1/1/2024	0.00145 ETH
2/1/2024	0.00213 ETH
3/1/2024	0.00394 ETH
4/1/2024	0.00254 ETH
5/1/2024	0.00227 ETH
6/1/2024	0.00214 ETH
7/1/2024	0.00328 ETH

Bảng 4.2: Phí giao dịch trung bình mỗi ngày trong tuần đầu tiên của tháng 1 năm 2024

Nhận xét:

- Ngày 3/1/2024 có phí giao dịch trung bình cao nhất, đạt 0.00394 ETH.
- Ngày 1/1/2024 ghi nhận phí giao dịch trung bình thấp nhất với 0.00145 ETH.
- Phí giao dịch trung bình có sự biến động trong tuần, với xu hướng cao vào các ngày 3 và 7/1.

4.2.3 Số lượng giao dịch mỗi ngày trong tuần đầu tiên của tháng 1, năm 2024

Dữ liệu ghi nhận số lượng giao dịch ETH mỗi ngày trong tuần đầu tiên của tháng 1 năm 2024 như sau:

Ngày	**Số lượng giao dịch**
1/1/2024	1,101,402
2/1/2024	1,128,449
3/1/2024	1,092,183
4/1/2024	1,116,530
5/1/2024	1,069,198
6/1/2024	994,884
7/1/2024	1,049,758

Bảng 4.3: Số lượng giao dịch mỗi ngày trong tuần đầu tiên của tháng 1 năm 2024

Nhận xét:

- Ngày 2/1/2024 có số lượng giao dịch cao nhất với 1,128,449 giao dịch.
- Ngày 6/1/2024 ghi nhận số lượng giao dịch thấp nhất với 994,884 giao dịch.
- Xu hướng chung cho thấy số lượng giao dịch giảm dần vào cuối tuần, với mức giảm rõ rệt nhất vào ngày 6/1.

4.2.4 Địa chỉ giao dịch phổ biến và số lượng giao dịch lớn

Địa chỉ nhận phổ biến nhất:

0xdac17f958d2ee523a2206206994597c13d831ec7: 683,237 lần nhận

0x3fc91a3afd70395cd496c647d5a6cc9d4b2b7fad: 457,611 lần nhận

0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48: 214,262 lần nhận

0xe87753eb91d6a61ea342bb9044a97764366cc7b2: 134,670 lần nhận

0xdef1c0ded9bec7f1a1670819833240f027b25eff: 81,386 lần nhận

Địa chỉ gửi phổ biến nhất:

0x75e89d5979e4f6fba9f97c104c2f0afb3f1dcb88: 77,226 lần gửi

0x28c6c06298d514db089934071355e5743bf21d60: 55,076 lần gửi

0x46340b20830761efd32832a74d7169b29feb9758: 52,148 lần gửi

0x21a31ee1afc51d94c2efccaa2092ad1028285549: 52,133 lần gửi

0xdfd5293d8e347dfe59e90efd55b2956a1343963d: 51,792 lần gửi

Các địa chỉ có tổng giá trị ETH gửi đi cao nhất:

0x28c6c06298d514db089934071355e5743bf21d60: 579,986.58 ETH

0xceb69f6342ece283b2f5c9088ff249b5d0ae66ea: 234,383.82 ETH

0x4976a4a02f38326660d17bf34b431dc6e2eb2327: 129,145.60 ETH

0xeae7380dd4cef6fbd1144f49e4d1e6964258a4f4: 122,103.81 ETH

0x267be1c1d684f78cb4f6a176c4911b741e4ffdc0: 101,254.10 ETH

Nhận xét:

- Địa chỉ nhận phổ biến nhất là 0xdac17f958d2ee523a2206206994597c13d831ec7 với số lần nhận cao nhất, đạt 683,237 lần.
- Địa chỉ gửi có tổng giá trị ETH gửi đi cao nhất là 0x28c6c06298d514db089934071355e5743bf21d60, với 579,986.58 ETH.
- Có một lượng lớn giao dịch (288,548) có giá trị lớn hơn mức trung bình, cho thấy nhiều giao dịch lớn đáng chú ý trong tuần.

4.2.5 Các câu truy vấn khác

Tổng số lượng giao dịch từ tất cả các địa chỉ gửi là 7552404 giao dịch.

Khối có lượng ETH giao dịch lớn nhất là 18922044 với hơn 73047.44 ETH được giao dịch.

Tổng cộng: 288,548 giao dịch có giá trị lớn hơn trung bình 1.1228 ETH.

- Giao dịch có phí thấp nhất:

Mã giao dịch:
0xd59590d1690743fa36804c5bc95c8147f8a60001dd944abedcc20a4ea7620cbc

Phương thức: Transfer

Block: 18,911,018

Thời gian: 7:10:11, ngày 1/1/2024

Từ: 0x1f9090aae28b8a3dceadf281b0f12828e676c326

Đến: 0x4675c7e5baafbffbca748158becba61ef3b0a263

Số lượng: 0.071248556 ETH

Phí giao dịch: 0.0001653 ETH

- Giao dịch có phí cao nhất:

Mã giao dịch:
0xf7f8a81f07fe845b59142f3bf8cff079b59800a20e18bc7c931dec5befed0b62

Phương thức: 0x608edee3

Block: 18,954,080

Thời gian: 8:38:47, ngày 7/1/2024

Từ: 0xae2fc483527b8ef99eb5d9b44875f005ba1fae13

Đến: 0x6b75d8af000000e20b7a7ddf000ba900b4009a80

Số lượng: 0.000000042 ETH

Phí giao dịch: 9.25514728 ETH

- Giao dịch mới thực hiện gần đây nhất:

Mã giao dịch:
0xe6fa8ea06fc08fe830efb8daa815ff1a591f330455d945226cd3eb9f46422fbd

Phương thức: Transfer

Block: 18,958,621

Thời gian: 23:59:59, ngày 7/1/2024

Từ: 0x4838b106fce9647bdf1e7877bf73ce8b0bad5f97

Đến: 0x388c818ca8b9251b393131c08a736a67ccb19297

Số lượng: 0.043344697 ETH

Phí giao dịch: 0.00063395 ETH

Nhận xét:

- Giao dịch có phí cao nhất ngày 7/1 có mức phí cao vượt trội (9.25514728 ETH), cao hơn rất nhiều so với các giao dịch khác.
- Giao dịch gần đây nhất thực hiện vào phút cuối cùng của ngày 7/1/2024, đánh dấu sự kết thúc cho tuần giao dịch đầu tiên của tháng.

CHƯƠNG 5: KẾT LUẬN VÀ KIẾN NGHỊ

5.1 Kết luận

Qua quá trình thu thập và phân tích dữ liệu giao dịch ETH từ Etherscan.io, chúng tôi nhận thấy rằng việc kết hợp Beautiful Soup – một công cụ mã nguồn mở mạnh mẽ – với MongoDB đã mang lại những kết quả quan trọng. Trước hết, cơ sở dữ liệu giao dịch lớn mà chúng tôi xây dựng cho phép phân tích chi tiết lịch sử giao dịch, giúp nhận diện xu hướng và hành vi người dùng trên mạng Ethereum. Thêm vào đó, cơ sở dữ liệu này đã tạo điều kiện thuận lợi cho việc tính toán các chỉ báo phân tích on-chain, từ đó cung cấp cái nhìn rõ ràng hơn về tiềm năng thị trường và hỗ trợ các quyết định đầu tư. Cuối cùng, đây cũng là nền tảng quan trọng để phát triển các nghiên cứu chuyên sâu không chỉ về Ethereum mà còn về các mạng lưới tiền kỹ thuật số khác.

5.2 Kiến nghị

Để nâng cao chất lượng và mở rộng phạm vi phân tích, chúng tôi đề xuất một số kiến nghị sau. Thứ nhất, tiếp tục nghiên cứu và khai thác sâu hơn các dữ liệu on-chain, giúp hiểu rõ hơn đặc điểm hành vi người dùng và hoạt động của các ví lớn trên mạng lưới. Thứ hai, áp dụng các mô hình học máy để khám phá các quy luật tiềm ẩn trong giao dịch, từ đó dự đoán chính xác xu hướng biến động và tối ưu hóa chiến lược đầu tư. Cuối cùng, mở rộng ứng dụng phương pháp này sang các mạng lưới tiền kỹ thuật số khác như Bitcoin, Binance Smart Chain (BSC) và các loại token đặc thù như NFT để thử nghiệm tính hiệu quả.

TÀI LIỆU THAM KHẢO

[1] "Beautiful Soup (HTML parser)," *Wikipedia*. Jun. 28, 2024. Accessed: Oct. 30, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Beautiful_Soup_(HTML_parser)&oldid=1231440478

[2] "Ethereum là gì và hoạt động như thế nào?" Accessed: Oct. 28, 2024. [Online]. Available: https://www.elcom.com.vn/ethereum-la-gi-va-hoat-dong-nhu-the-nao-1696933749

[3] FPT C. ty C. phần B. lẻ K., "Giới thiệu về đặc điểm và cách ứng dụng Web Scraping trong cuộc sống hiện nay." Accessed: Oct. 30, 2024. [Online]. Available: https://fptshop.com.vn/tin-tuc/danh-gia/web-scraping-166852

[4] "Nstbrowser - Trình duyệt chống phát hiện nâng cao để quét web và quản lý nhiều tài khoản." Accessed: Oct. 30, 2024. [Online]. Available: https://www.nstbrowser.io/vi/blog/scrapy-vs-beautifulsoup

[5] "Công nghệ Blockchain là gì? - Giải thích về công nghệ Blockchain - AWS," Amazon Web Services, Inc. Accessed: Oct. 30, 2024. [Online]. Available: https://aws.amazon.com/vi/what-is/blockchain/

[6] "Ứng Dụng Phi Tập Trung (DApp) Là Gì?," Binance Academy. Accessed: Oct. 30, 2024. [Online]. Available: https://academy.binance.com/vi/articles/what-are-decentralized-applications-dapps

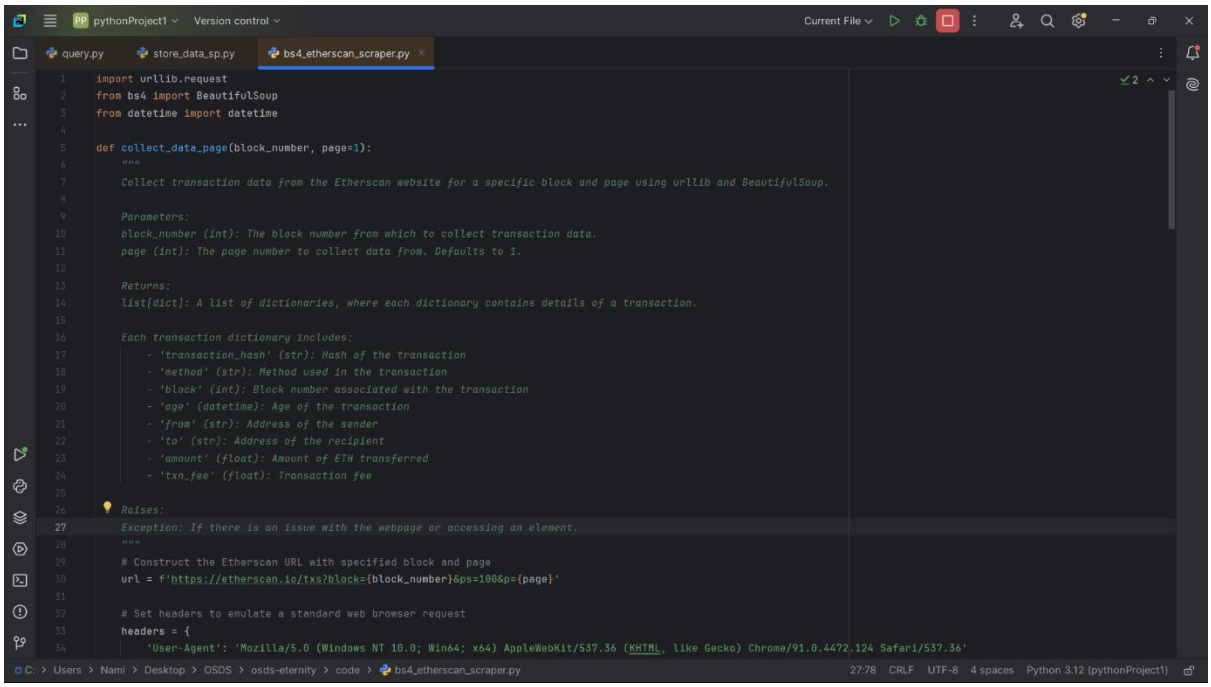
[7] "What Is Ether (ETH), the Cryptocurrency of Ethereum Apps?," Investopedia. Accessed: Oct. 30, 2024. [Online]. Available: https://www.investopedia.com/tech/what-ether-it-same-ethereum/

[8] "Proof-of-Work vs. Proof-of-Stake: Why did Ethereum Switch to Proof-of-Stake?," Bake Blog. Accessed: Oct. 30, 2024. [Online]. Available: https://blog.bake.io/why-did-ethereum-switch-to-proof-of-stake/

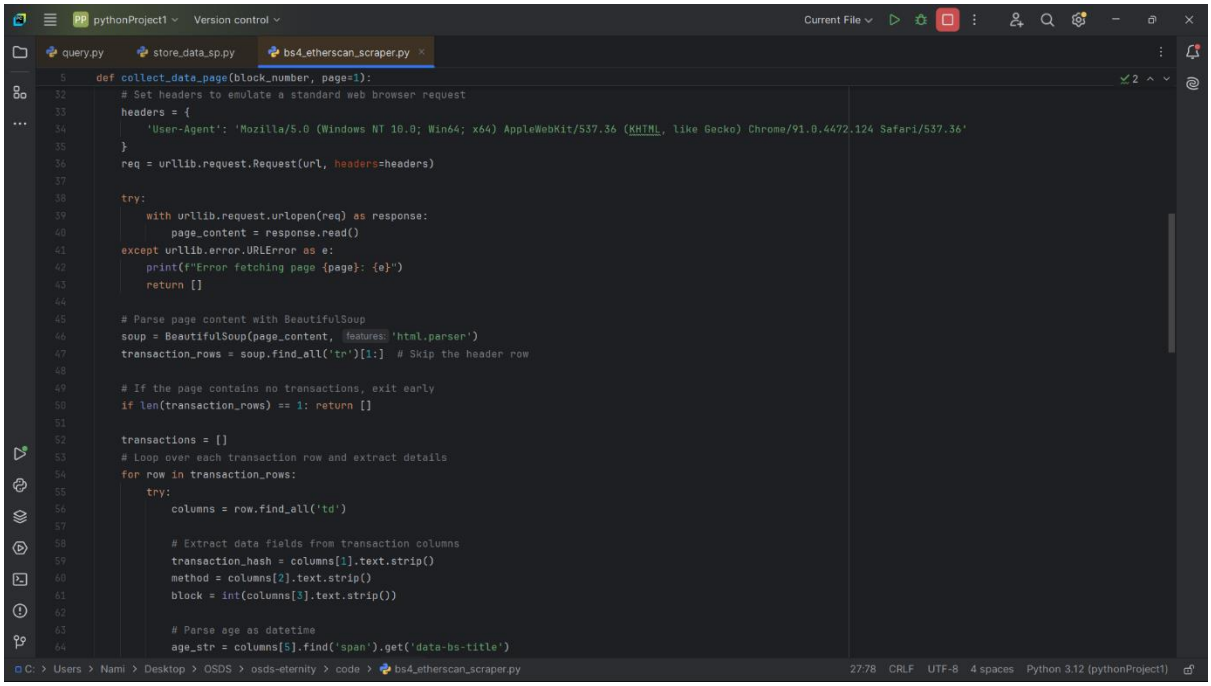
[9] CMS T., "Distributed Database," ScyllaDB. Accessed: Oct. 30, 2024. [Online]. Available: https://www.scylladb.com/glossary/distributed-database/

[10] TopDev, "MongoDB là gì? Định nghĩa và chi tiết về MongoDB," TopDev. Accessed: Oct. 30, 2024. [Online]. Available: https://topdev.vn/blog/mongodb-la-gi/

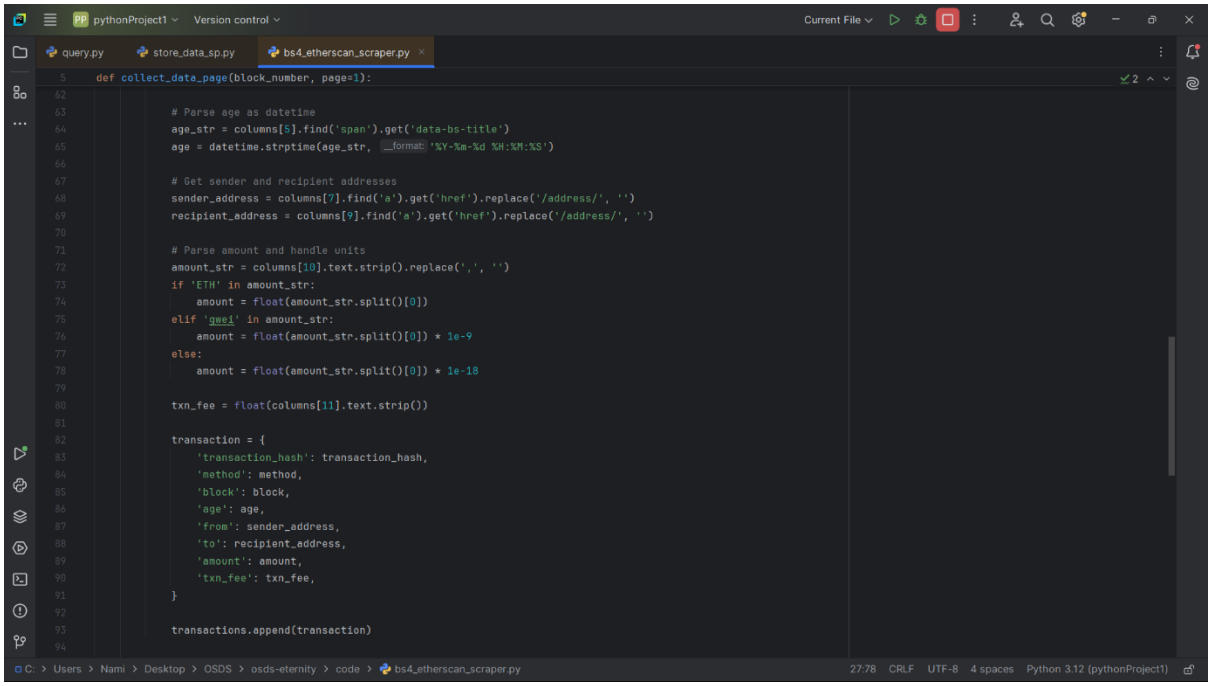
Phụ lục 1



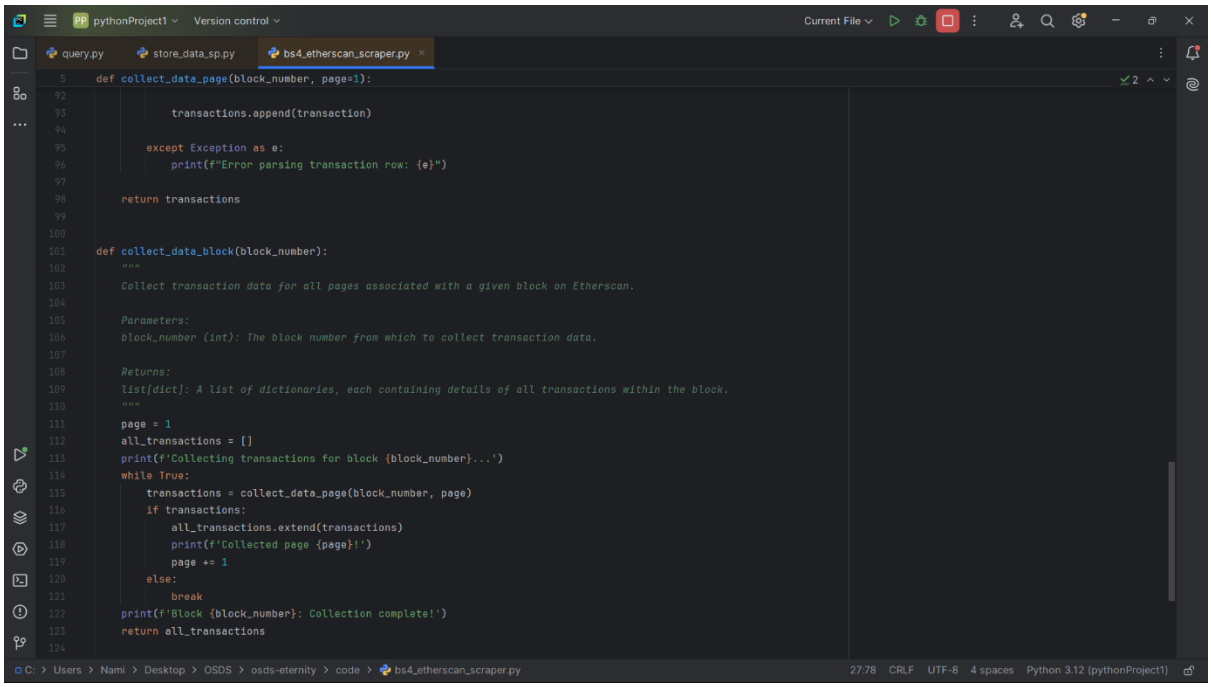
```python
import urllib.request
from bs4 import BeautifulSoup
from datetime import datetime

def collect_data_page(block_number, page=1):
    """
    Collect transaction data from the Etherscan website for a specific block and page using urllib and BeautifulSoup.

    Parameters:
    block_number (int): The block number from which to collect transaction data.
    page (int): The page number to collect data from. Defaults to 1.

    Returns:
    list[dict]: A list of dictionaries, where each dictionary contains details of a transaction.

    Each transaction dictionary includes:
        - 'transaction_hash' (str): Hash of the transaction
        - 'method' (str): Method used in the transaction
        - 'block' (int): Block number associated with the transaction
        - 'age' (datetime): Age of the transaction
        - 'from' (str): Address of the sender
        - 'to' (str): Address of the recipient
        - 'amount' (float): Amount of ETH transferred
        - 'txn_fee' (float): Transaction fee

    Raises:
    Exception: If there is an issue with the webpage or accessing an element.
    """
    # Construct the Etherscan URL with specified block and page
    url = f'https://etherscan.io/txs?block={block_number}&ps=100&p={page}'

    # Set headers to emulate a standard web browser request
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36'
```



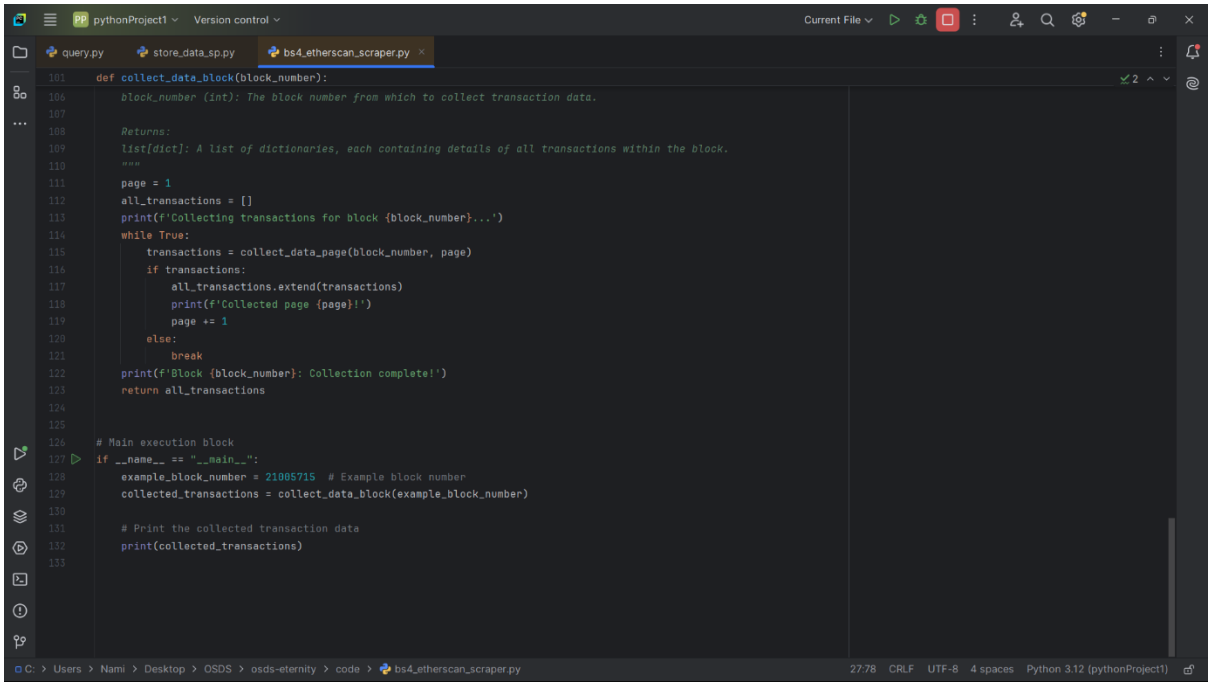
```python
def collect_data_page(block_number, page=1):
    # Set headers to emulate a standard web browser request
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36'
    }
    req = urllib.request.Request(url, headers=headers)

    try:
        with urllib.request.urlopen(req) as response:
            page_content = response.read()
    except urllib.error.URLError as e:
        print(f"Error fetching page {page}: {e}")
        return []

    # Parse page content with BeautifulSoup
    soup = BeautifulSoup(page_content, features='html.parser')
    transaction_rows = soup.find_all('tr')[1:]  # Skip the header row

    # If the page contains no transactions, exit early
    if len(transaction_rows) == 1: return []

    transactions = []
    # Loop over each transaction row and extract details
    for row in transaction_rows:
        try:
            columns = row.find_all('td')

            # Extract data fields from transaction columns
            transaction_hash = columns[1].text.strip()
            method = columns[2].text.strip()
            block = int(columns[3].text.strip())

            # Parse age as datetime
            age_str = columns[5].find('span').get('data-bs-title')
```

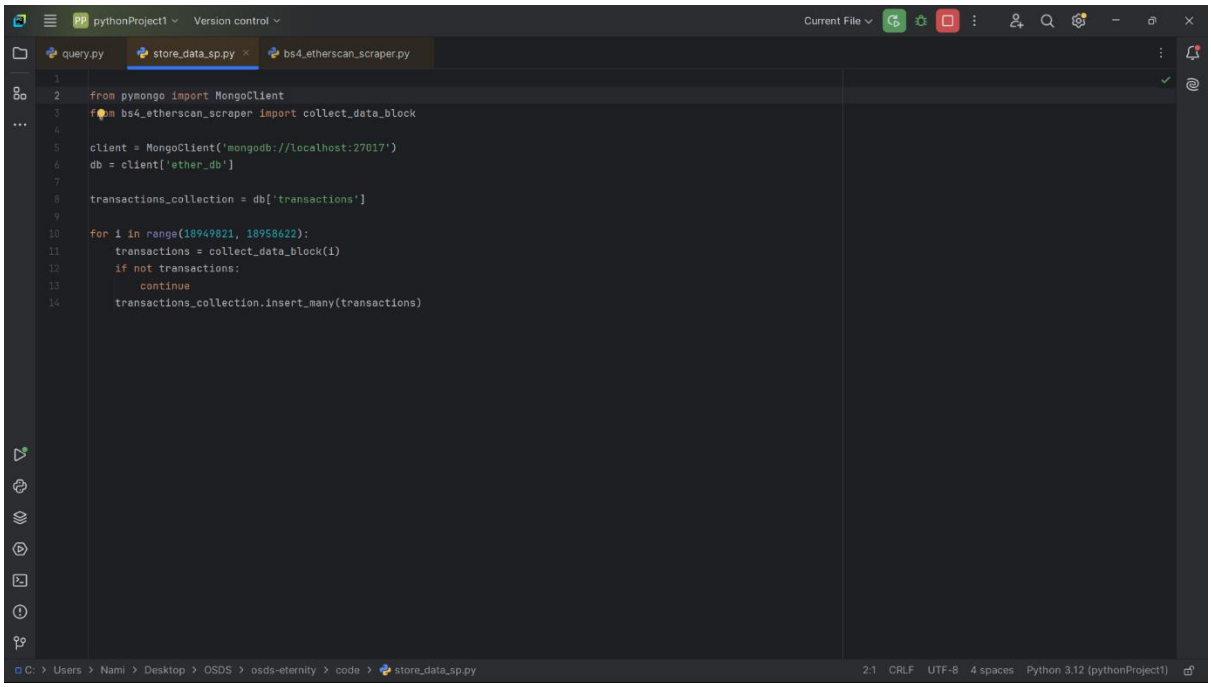


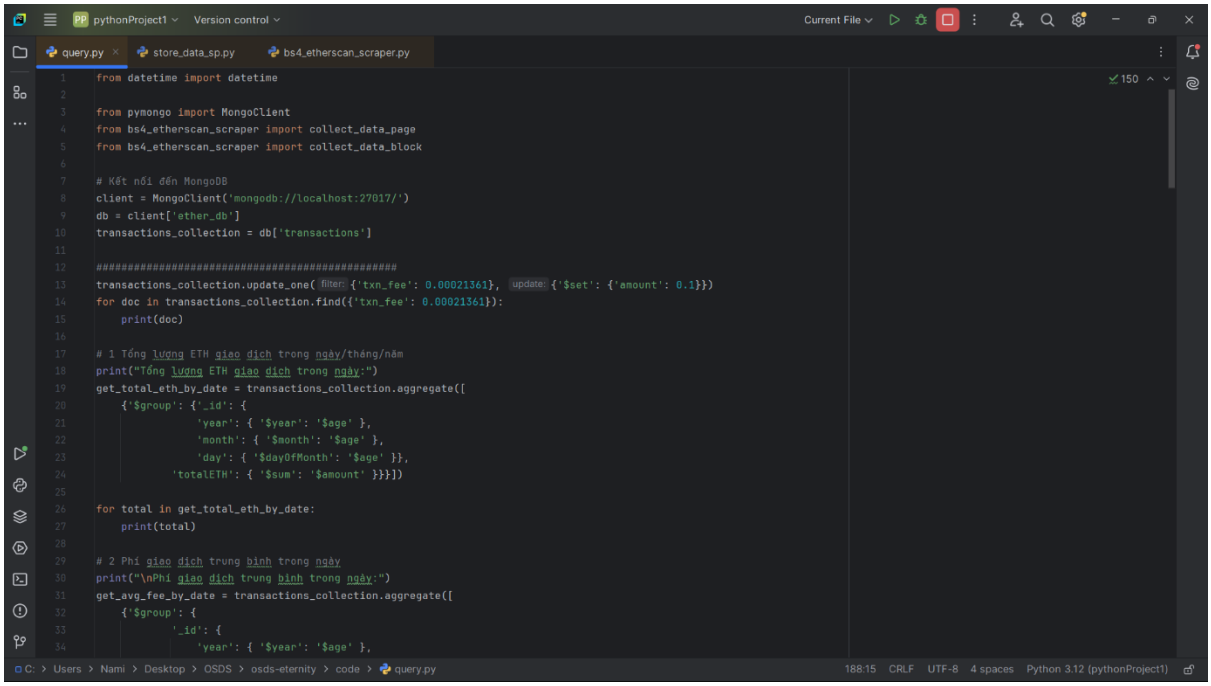
```
def collect_data_page(block_number, page=1):

            # Parse age as datetime
            age_str = columns[5].find('span').get('data-bs-title')
            age = datetime.strptime(age_str, __format: '%Y-%m-%d %H:%M:%S')

            # Get sender and recipient addresses
            sender_address = columns[7].find('a').get('href').replace('/address/', '')
            recipient_address = columns[9].find('a').get('href').replace('/address/', '')

            # Parse amount and handle units
            amount_str = columns[10].text.strip().replace(',', '')
            if 'ETH' in amount_str:
                amount = float(amount_str.split()[0])
            elif 'gwei' in amount_str:
                amount = float(amount_str.split()[0]) * 1e-9
            else:
                amount = float(amount_str.split()[0]) * 1e-18

            txn_fee = float(columns[11].text.strip())

            transaction = {
                'transaction_hash': transaction_hash,
                'method': method,
                'block': block,
                'age': age,
                'from': sender_address,
                'to': recipient_address,
                'amount': amount,
                'txn_fee': txn_fee,
            }

            transactions.append(transaction)
```



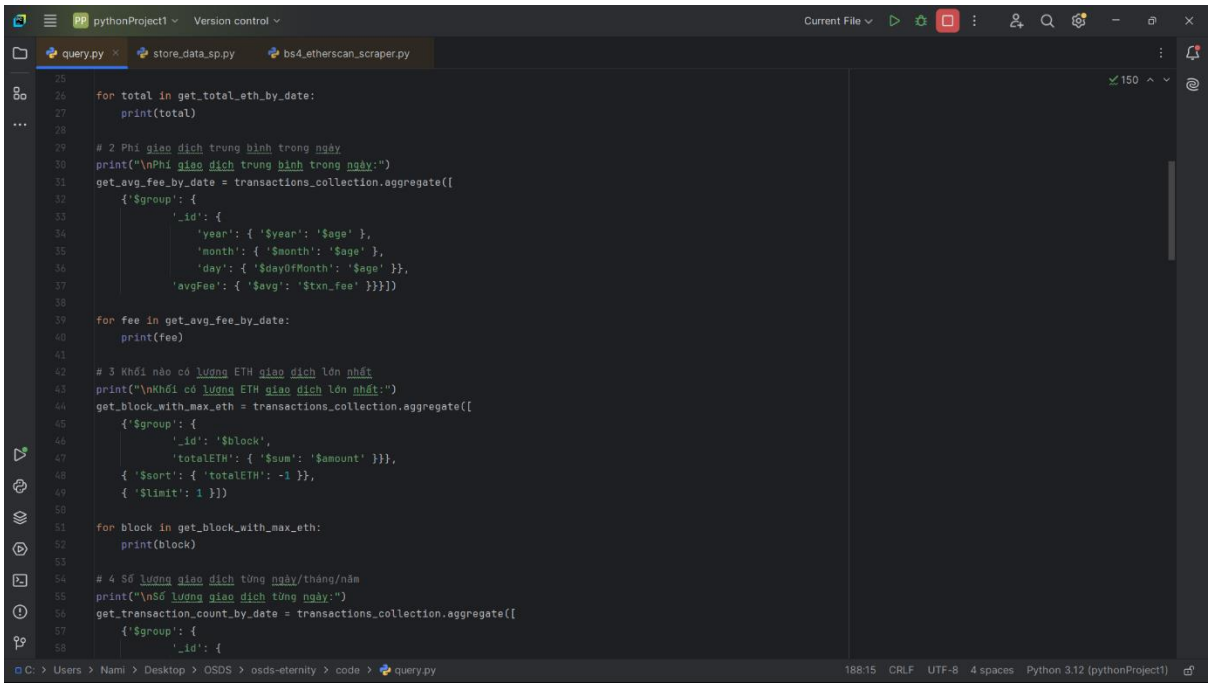
```
            transactions.append(transaction)

        except Exception as e:
            print(f"Error parsing transaction row: {e}")

    return transactions


def collect_data_block(block_number):
    """
    Collect transaction data for all pages associated with a given block on Etherscan.

    Parameters:
    block_number (int): The block number from which to collect transaction data.

    Returns:
    list[dict]: A list of dictionaries, each containing details of all transactions within the block.
    """
    page = 1
    all_transactions = []
    print(f'Collecting transactions for block {block_number}...')
    while True:
        transactions = collect_data_page(block_number, page)
        if transactions:
            all_transactions.extend(transactions)
            print(f'Collected page {page}!')
            page += 1
        else:
            break
    print(f'Block {block_number}: Collection complete!')
    return all_transactions
```


pythonProject1
Version control
Current File
query.py
store_data_sp.py
bs4_etherscan_scraper.py
def collect_data_block(block_number):
block_number (int): The block number from which to collect transaction data.
Returns:
list[dict]: A list of dictionaries, each containing details of all transactions within the block.
"""
page = 1
all_transactions = []
print(f'Collecting transactions for block {block_number}...')
while True:
transactions = collect_data_page(block_number, page)
if transactions:
all_transactions.extend(transactions)
print(f'Collected page {page}!')
page += 1
else:
break
print(f'Block {block_number}: Collection complete!')
return all_transactions
# Main execution block
if __name__ == "__main__":
example_block_number = 21005715 # Example block number
collected_transactions = collect_data_block(example_block_number)
# Print the collected transaction data
print(collected_transactions)
C: > Users > Nami > Desktop > OSDS > osds-eternity > code > bs4_etherscan_scraper.py
27:78 CRLF UTF-8 4 spaces Python 3.12 (pythonProject1)

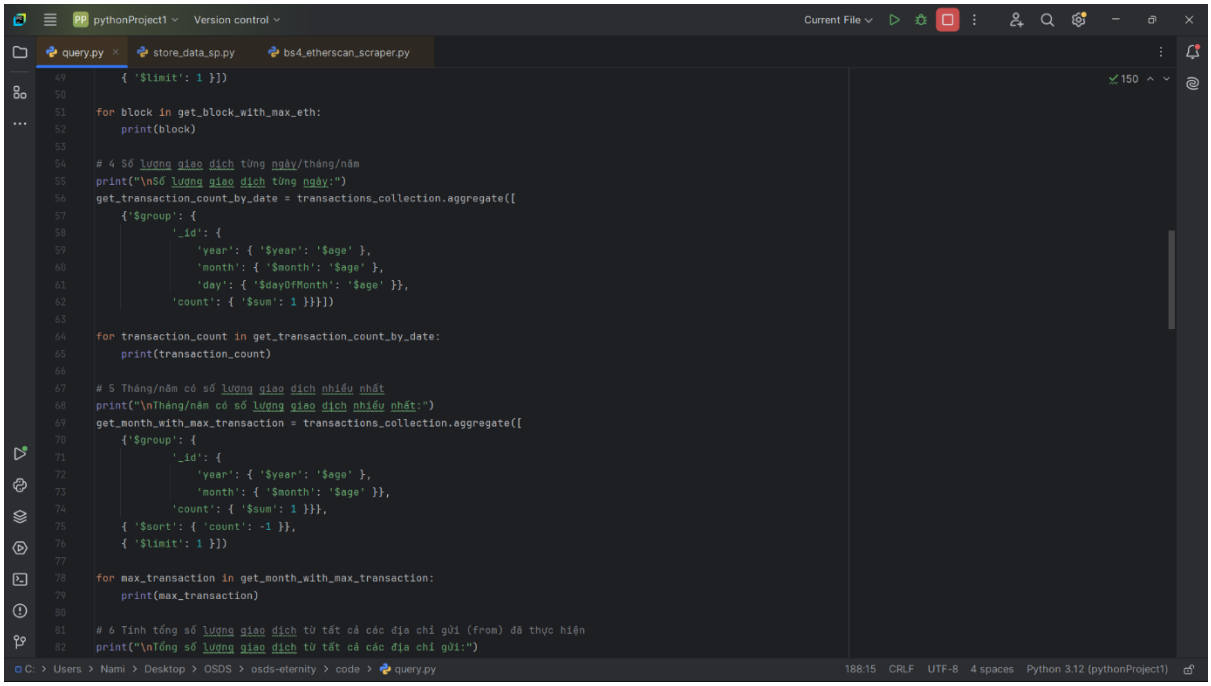

pythonProject1
Version control
Current File
query.py
store_data_sp.py
bs4_etherscan_scraper.py
from pymongo import MongoClient
from bs4_etherscan_scraper import collect_data_block
client = MongoClient('mongodb://localhost:27017')
db = client['ether_db']
transactions_collection = db['transactions']
for i in range(18949821, 18958622):
transactions = collect_data_block(i)
if not transactions:
continue
transactions_collection.insert_many(transactions)
C: > Users > Nami > Desktop > OSDS > osds-eternity > code > store_data_sp.py
2:1 CRLF UTF-8 4 spaces Python 3.12 (pythonProject1)



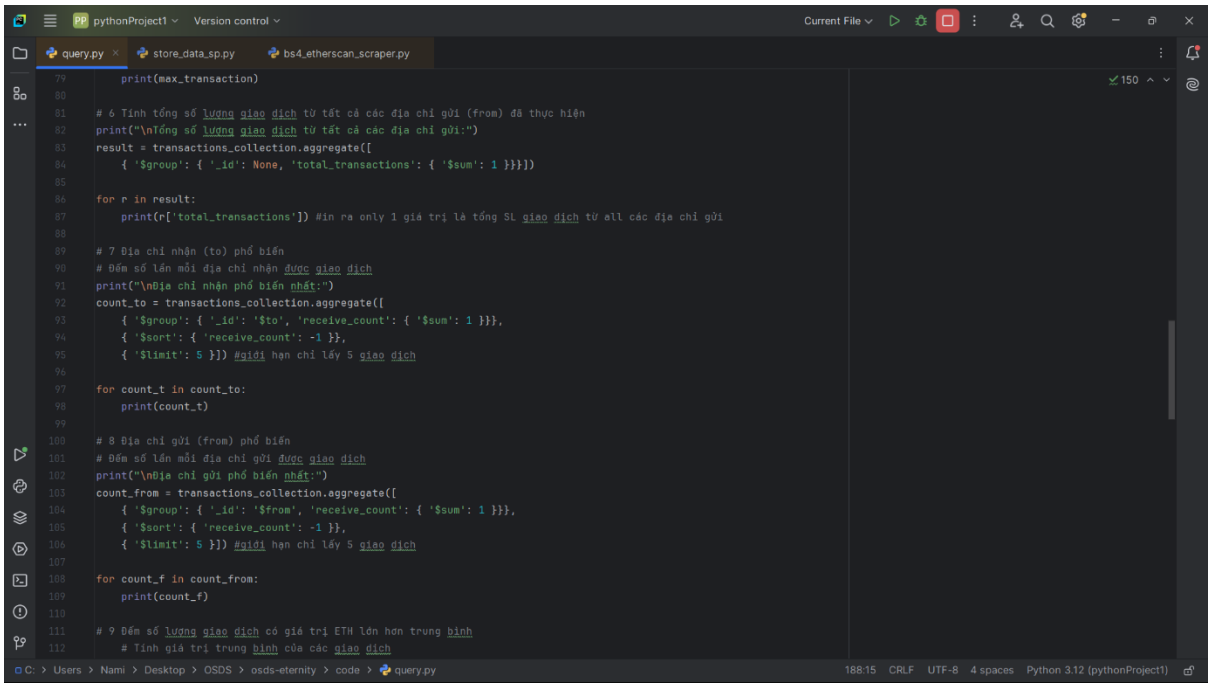
```python
from datetime import datetime

from pymongo import MongoClient
from bs4_etherscan_scraper import collect_data_page
from bs4_etherscan_scraper import collect_data_block

# Kết nối đến MongoDB
client = MongoClient('mongodb://localhost:27017/')
db = client['ether_db']
transactions_collection = db['transactions']

################################################
transactions_collection.update_one( filter: {'txn_fee': 0.00021361}, update: {'$set': {'amount': 0.1}})
for doc in transactions_collection.find({'txn_fee': 0.00021361}):
    print(doc)

# 1 Tổng lượng ETH giao dịch trong ngày/tháng/năm
print("Tổng lượng ETH giao dịch trong ngày:")
get_total_eth_by_date = transactions_collection.aggregate([
    {'$group': {'_id': {
                'year': { '$year': '$age' },
                'month': { '$month': '$age' },
                'day': { '$dayOfMonth': '$age' }},
            'totalETH': { '$sum': '$amount' }}}])

for total in get_total_eth_by_date:
    print(total)

# 2 Phí giao dịch trung bình trong ngày
print("\nPhí giao dịch trung bình trong ngày:")
get_avg_fee_by_date = transactions_collection.aggregate([
    {'$group': {
            '_id': {
                'year': { '$year': '$age' },
```

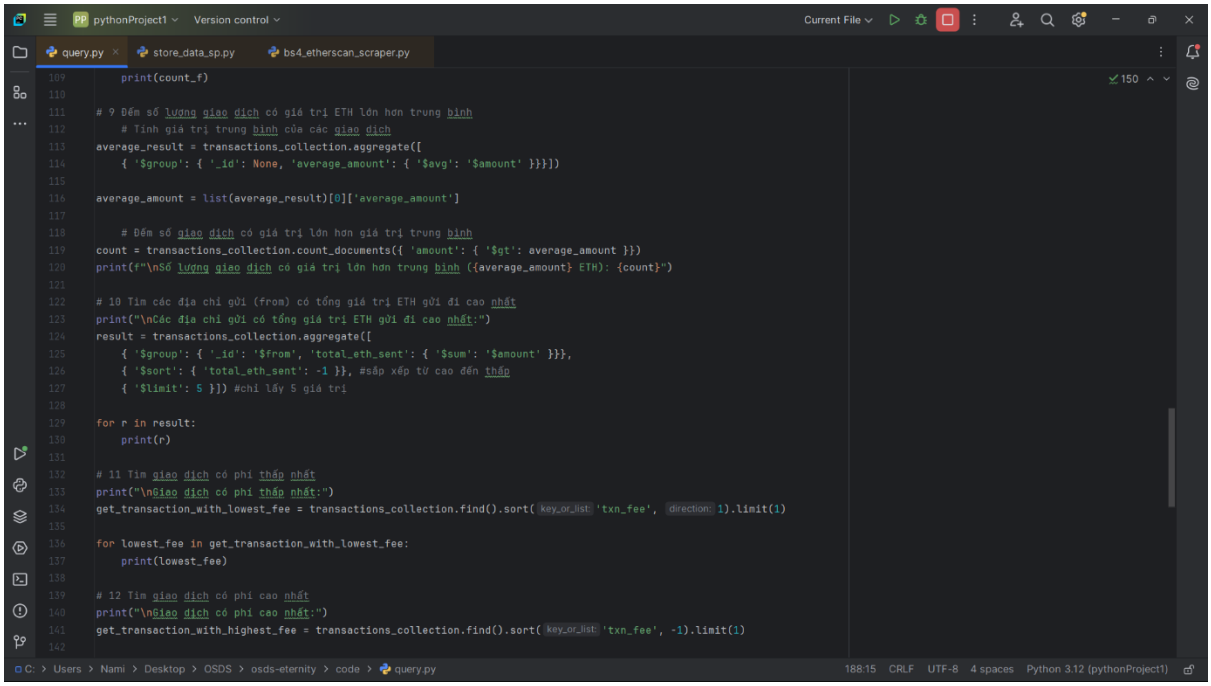
```python
for total in get_total_eth_by_date:
    print(total)

# 2 Phí giao dịch trung bình trong ngày
print("\nPhí giao dịch trung bình trong ngày:")
get_avg_fee_by_date = transactions_collection.aggregate([
    {'$group': {
            '_id': {
                'year': { '$year': '$age' },
                'month': { '$month': '$age' },
                'day': { '$dayOfMonth': '$age' }},
            'avgFee': { '$avg': '$txn_fee' }}}])

for fee in get_avg_fee_by_date:
    print(fee)

# 3 Khối nào có lượng ETH giao dịch lớn nhất
print("\nKhối có lượng ETH giao dịch lớn nhất:")
get_block_with_max_eth = transactions_collection.aggregate([
    {'$group': {
            '_id': '$block',
            'totalETH': { '$sum': '$amount' }}},
    { '$sort': { 'totalETH': -1 }},
    { '$limit': 1 }])

for block in get_block_with_max_eth:
    print(block)

# 4 Số lượng giao dịch từng ngày/tháng/năm
print("\nSố lượng giao dịch từng ngày:")
get_transaction_count_by_date = transactions_collection.aggregate([
    {'$group': {
            '_id': {
```



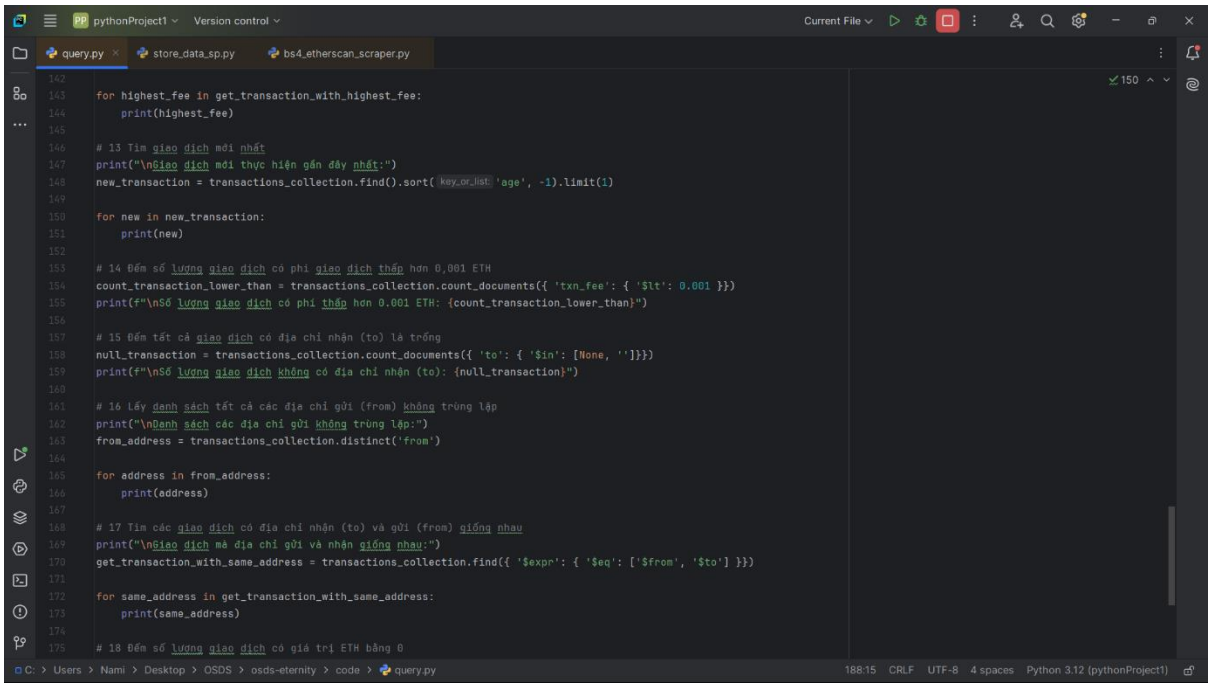
```
        { '$limit': 1 }])

    for block in get_block_with_max_eth:
        print(block)

    # 4 Số lượng giao dịch từng ngày/tháng/năm
    print("\nSố lượng giao dịch từng ngày:")
    get_transaction_count_by_date = transactions_collection.aggregate([
        {'$group': {
                '_id': {
                    'year': { '$year': '$age' },
                    'month': { '$month': '$age' },
                    'day': { '$dayOfMonth': '$age' }},
                'count': { '$sum': 1 }}}])

    for transaction_count in get_transaction_count_by_date:
        print(transaction_count)

    # 5 Tháng/năm có số lượng giao dịch nhiều nhất
    print("\nTháng/năm có số lượng giao dịch nhiều nhất:")
    get_month_with_max_transaction = transactions_collection.aggregate([
        {'$group': {
                '_id': {
                    'year': { '$year': '$age' },
                    'month': { '$month': '$age' }},
                'count': { '$sum': 1 }}},
        { '$sort': { 'count': -1 }},
        { '$limit': 1 }])

    for max_transaction in get_month_with_max_transaction:
        print(max_transaction)

    # 6 Tính tổng số lượng giao dịch từ tất cả các địa chỉ gửi (from) đã thực hiện
    print("\nTổng số lượng giao dịch từ tất cả các địa chỉ gửi:")
```



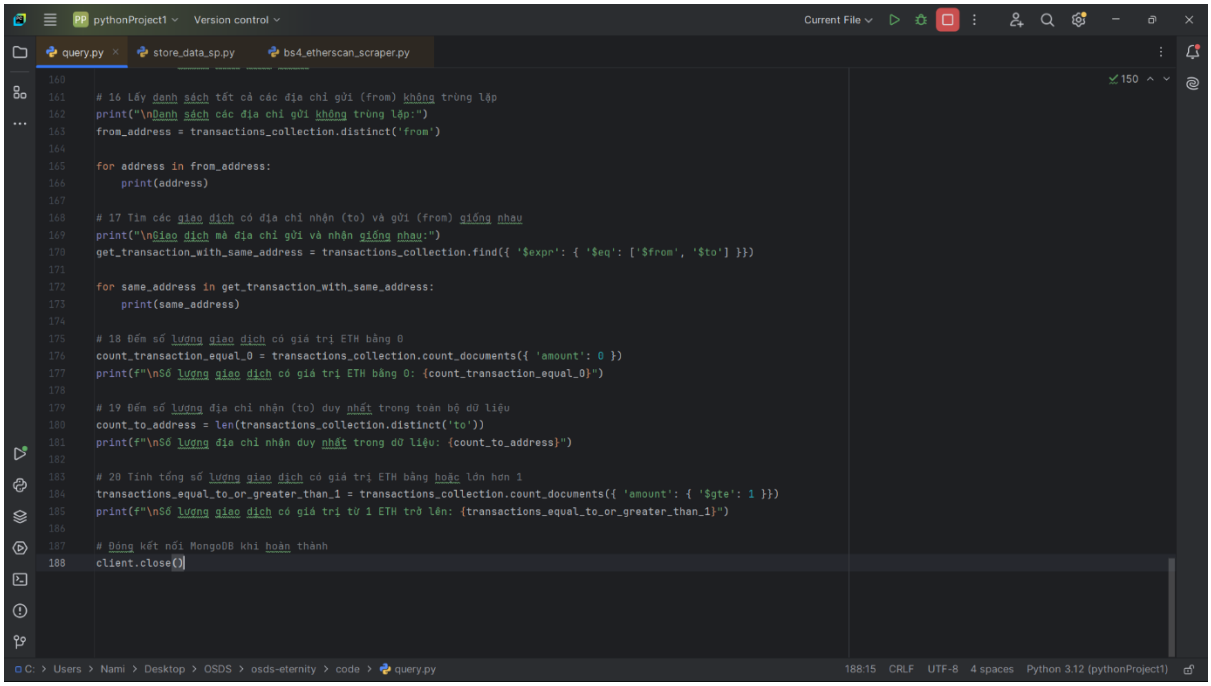
```
        print(max_transaction)

    # 6 Tính tổng số lượng giao dịch từ tất cả các địa chỉ gửi (from) đã thực hiện
    print("\nTổng số lượng giao dịch từ tất cả các địa chỉ gửi:")
    result = transactions_collection.aggregate([
        { '$group': { '_id': None, 'total_transactions': { '$sum': 1 }}}])

    for r in result:
        print(r['total_transactions']) #in ra only 1 giá trị là tổng SL giao dịch từ all các địa chỉ gửi

    # 7 Địa chỉ nhận (to) phổ biến
    # Đếm số lần mỗi địa chỉ nhận được giao dịch
    print("\nĐịa chỉ nhận phổ biến nhất:")
    count_to = transactions_collection.aggregate([
        { '$group': { '_id': '$to', 'receive_count': { '$sum': 1 }}},
        { '$sort': { 'receive_count': -1 }},
        { '$limit': 5 }]) #giới hạn chỉ lấy 5 giao dịch

    for count_t in count_to:
        print(count_t)

    # 8 Địa chỉ gửi (from) phổ biến
    # Đếm số lần mỗi địa chỉ gửi được giao dịch
    print("\nĐịa chỉ gửi phổ biến nhất:")
    count_from = transactions_collection.aggregate([
        { '$group': { '_id': '$from', 'receive_count': { '$sum': 1 }}},
        { '$sort': { 'receive_count': -1 }},
        { '$limit': 5 }]) #giới hạn chỉ lấy 5 giao dịch

    for count_f in count_from:
        print(count_f)

    # 9 Đếm số lượng giao dịch có giá trị ETH lớn hơn trung bình
        # Tính giá trị trung bình của các giao dịch
```



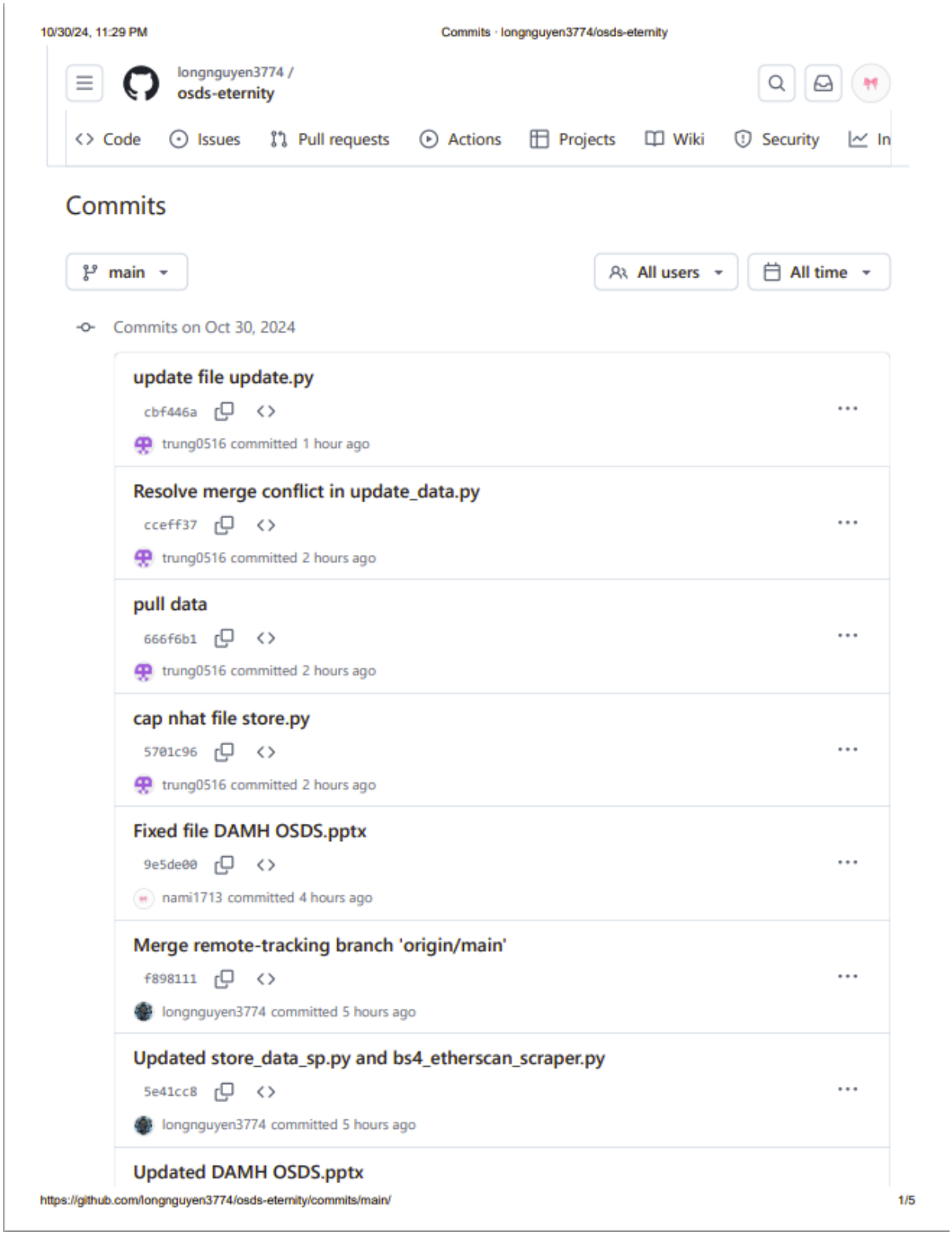
```
    print(count_f)

# 9 Đếm số lượng giao dịch có giá trị ETH lớn hơn trung bình
    # Tính giá trị trung bình của các giao dịch
average_result = transactions_collection.aggregate([
    { '$group': { '_id': None, 'average_amount': { '$avg': '$amount' }}}])

average_amount = list(average_result)[0]['average_amount']

    # Đếm số giao dịch có giá trị lớn hơn giá trị trung bình
count = transactions_collection.count_documents({ 'amount': { '$gt': average_amount }})
print(f"\nSố lượng giao dịch có giá trị lớn hơn trung bình ({average_amount} ETH): {count}")

# 10 Tìm các địa chỉ gửi (from) có tổng giá trị ETH gửi đi cao nhất
print("\nCác địa chỉ gửi có tổng giá trị ETH gửi đi cao nhất:")
result = transactions_collection.aggregate([
    { '$group': { '_id': '$from', 'total_eth_sent': { '$sum': '$amount' }}},
    { '$sort': { 'total_eth_sent': -1 }}, #sắp xếp từ cao đến thấp
    { '$limit': 5 }]) #chỉ lấy 5 giá trị

for r in result:
    print(r)

# 11 Tìm giao dịch có phí thấp nhất
print("\nGiao dịch có phí thấp nhất:")
get_transaction_with_lowest_fee = transactions_collection.find().sort( key_or_list: 'txn_fee', direction: 1).limit(1)

for lowest_fee in get_transaction_with_lowest_fee:
    print(lowest_fee)

# 12 Tìm giao dịch có phí cao nhất
print("\nGiao dịch có phí cao nhất:")
get_transaction_with_highest_fee = transactions_collection.find().sort( key_or_list: 'txn_fee', -1).limit(1)
```



```
for highest_fee in get_transaction_with_highest_fee:
    print(highest_fee)

# 13 Tìm giao dịch mới nhất
print("\nGiao dịch mới thực hiện gần đây nhất:")
new_transaction = transactions_collection.find().sort( key_or_list: 'age', -1).limit(1)

for new in new_transaction:
    print(new)

# 14 Đếm số lượng giao dịch có phí giao dịch thấp hơn 0,001 ETH
count_transaction_lower_than = transactions_collection.count_documents({ 'txn_fee': { '$lt': 0.001 }})
print(f"\nSố lượng giao dịch có phí thấp hơn 0.001 ETH: {count_transaction_lower_than}")

# 15 Đếm tất cả giao dịch có địa chỉ nhận (to) là trống
null_transaction = transactions_collection.count_documents({ 'to': { '$in': [None, '']}})
print(f"\nSố lượng giao dịch không có địa chỉ nhận (to): {null_transaction}")

# 16 Lấy danh sách tất cả các địa chỉ gửi (from) không trùng lặp
print("\nDanh sách các địa chỉ gửi không trùng lặp:")
from_address = transactions_collection.distinct('from')

for address in from_address:
    print(address)

# 17 Tìm các giao dịch có địa chỉ nhận (to) và gửi (from) giống nhau
print("\nGiao dịch mà địa chỉ gửi và nhận giống nhau:")
get_transaction_with_same_address = transactions_collection.find({ '$expr': { '$eq': ['$from', '$to'] }})

for same_address in get_transaction_with_same_address:
    print(same_address)

# 18 Đếm số lượng giao dịch có giá trị ETH bằng 0
```


pythonProject1
Version control
Current File
query.py
store_data_sp.py
bs4_etherscan_scraper.py
# 16 Lấy danh sách tất cả các địa chỉ gửi (from) không trùng lặp
print("\nDanh sách các địa chỉ gửi không trùng lặp:")
from_address = transactions_collection.distinct('from')
for address in from_address:
print(address)
# 17 Tìm các giao dịch có địa chỉ nhận (to) và gửi (from) giống nhau
print("\nGiao dịch mà địa chỉ gửi và nhận giống nhau:")
get_transaction_with_same_address = transactions_collection.find({ '$expr': { '$eq': ['$from', '$to'] } })
for same_address in get_transaction_with_same_address:
print(same_address)
# 18 Đếm số lượng giao dịch có giá trị ETH bằng 0
count_transaction_equal_0 = transactions_collection.count_documents({ 'amount': 0 })
print(f"\nSố lượng giao dịch có giá trị ETH bằng 0: {count_transaction_equal_0}")
# 19 Đếm số lượng địa chỉ nhận (to) duy nhất trong toàn bộ dữ liệu
count_to_address = len(transactions_collection.distinct('to'))
print(f"\nSố lượng địa chỉ nhận duy nhất trong dữ liệu: {count_to_address}")
# 20 Tính tổng số lượng giao dịch có giá trị ETH bằng hoặc lớn hơn 1
transactions_equal_to_or_greater_than_1 = transactions_collection.count_documents({ 'amount': { '$gte': 1 } })
print(f"\nSố lượng giao dịch có giá trị từ 1 ETH trở lên: {transactions_equal_to_or_greater_than_1}")
# Đóng kết nối MongoDB khi hoàn thành
client.close()
C: > Users > Nami > Desktop > OSDS > osds-eternity > code > query.py
188:15
CRLF
UTF-8
4 spaces
Python 3.12 (pythonProject1)

Phụ lục 2


10/30/24, 11:29 PM
Commits · longnguyen3774/osds-eternity
longnguyen3774 /
osds-eternity
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
In
Commits
main
All users
All time
Commits on Oct 30, 2024
update file update.py
cbf446a
trung0516 committed 1 hour ago
Resolve merge conflict in update_data.py
cceff37
trung0516 committed 2 hours ago
pull data
666f6b1
trung0516 committed 2 hours ago
cap nhat file store.py
5701c96
trung0516 committed 2 hours ago
Fixed file DAMH OSDS.pptx
9e5de00
nami1713 committed 4 hours ago
Merge remote-tracking branch 'origin/main'
f898111
longnguyen3774 committed 5 hours ago
Updated store_data_sp.py and bs4_etherscan_scraper.py
5e41cc8
longnguyen3774 committed 5 hours ago
Updated DAMH OSDS.pptx
https://github.com/longnguyen3774/osds-eternity/commits/main/
1/5

7614432

nami1713 committed 5 hours ago

Added folder and file ppt

a38c5a0

nami1713 committed 8 hours ago

Reup query.py

fe20286

nami1713 committed 10 hours ago

Commits on Oct 29, 2024

Updated bs4_etherscan_scraper.py

4e648fe

longnguyen3774 committed yesterday

Updated bs4_etherscan_scraper.py

0e8c4a5

longnguyen3774 committed yesterday

Updated store_data_sp.py

6efb68b

longnguyen3774 committed yesterday

Updated store_data_sp.py

803f7f8

longnguyen3774 committed yesterday

Added store_data_sp.py

35fdd7f

longnguyen3774 committed yesterday

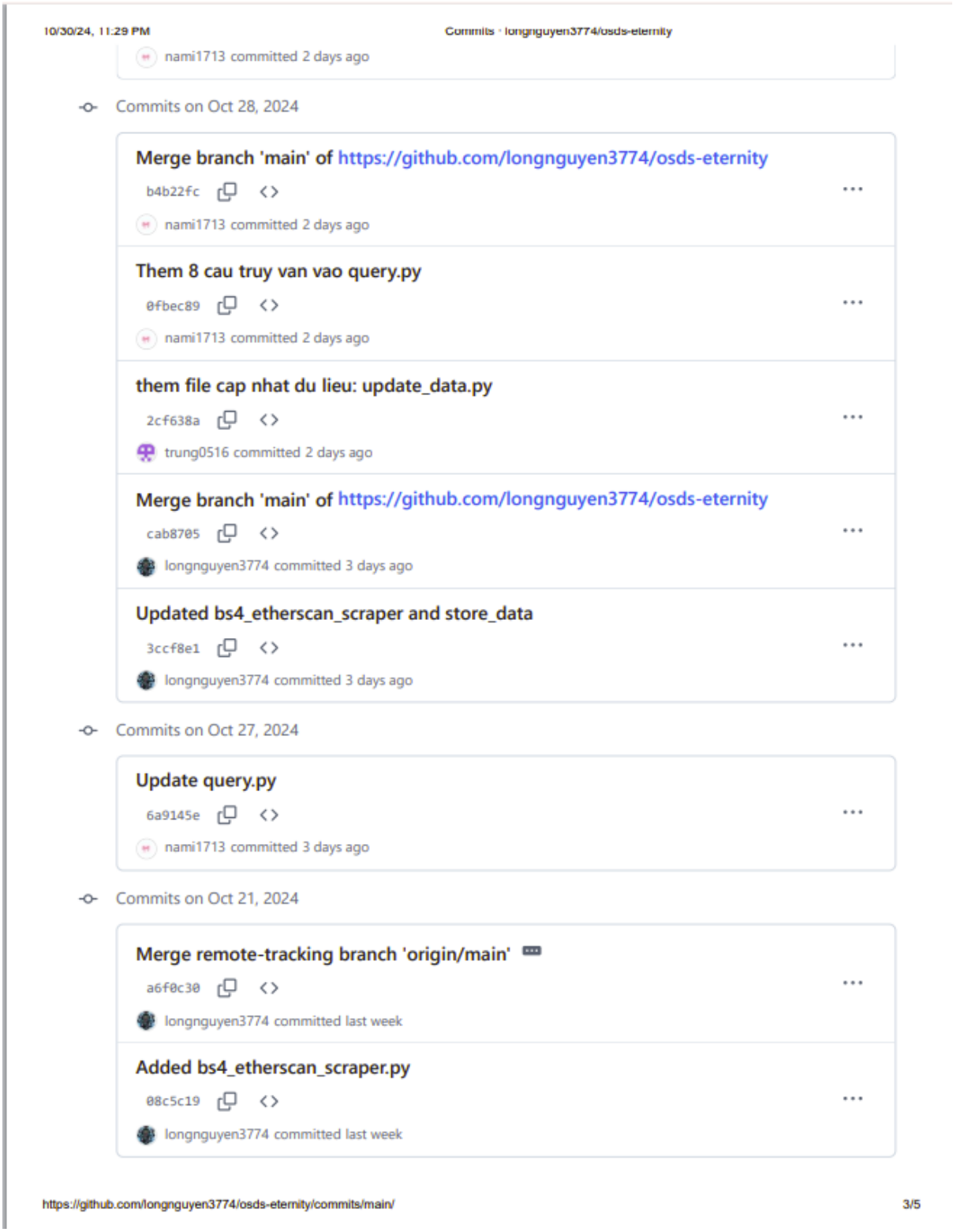
Added 2 more queries and fixed code in query.py

8313e5a

nami1713 committed yesterday

Tat code chua truy van duoc data trong query.py

1c12ef8


nami1713 committed 2 days ago
Commits on Oct 28, 2024
Merge branch 'main' of https://github.com/longnguyen3774/osds-eternity
b4b22fc
nami1713 committed 2 days ago
Them 8 cau truy van vao query.py
0fbec89
nami1713 committed 2 days ago
them file cap nhat du lieu: update_data.py
2cf638a
trung0516 committed 2 days ago
Merge branch 'main' of https://github.com/longnguyen3774/osds-eternity
cab8705
longnguyen3774 committed 3 days ago
Updated bs4_etherscan_scraper and store_data
3ccf8e1
longnguyen3774 committed 3 days ago
Commits on Oct 27, 2024
Update query.py
6a9145e
nami1713 committed 3 days ago
Commits on Oct 21, 2024
Merge remote-tracking branch 'origin/main'
a6f0c30
longnguyen3774 committed last week
Added bs4_etherscan_scraper.py
08c5c19
longnguyen3774 committed last week

Commits on Oct 20, 2024

Update query.py
586df24 nami1713 committed last week

Update query.py
82968ff nami1713 committed last week

Added query.py
aa4e688 nami1713 committed last week

Updated etherscan_scraper
1bcaca5 longnguyen3774 committed last week

Updated ethercan_scraper
3b43475 longnguyen3774 committed last week

thêm file update.py
99c9cb0 trung0516 committed last week

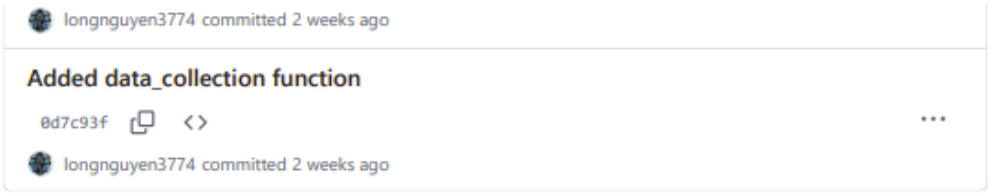
thêm file store.py
33efbf8 trung0516 committed last week

Commits on Oct 18, 2024

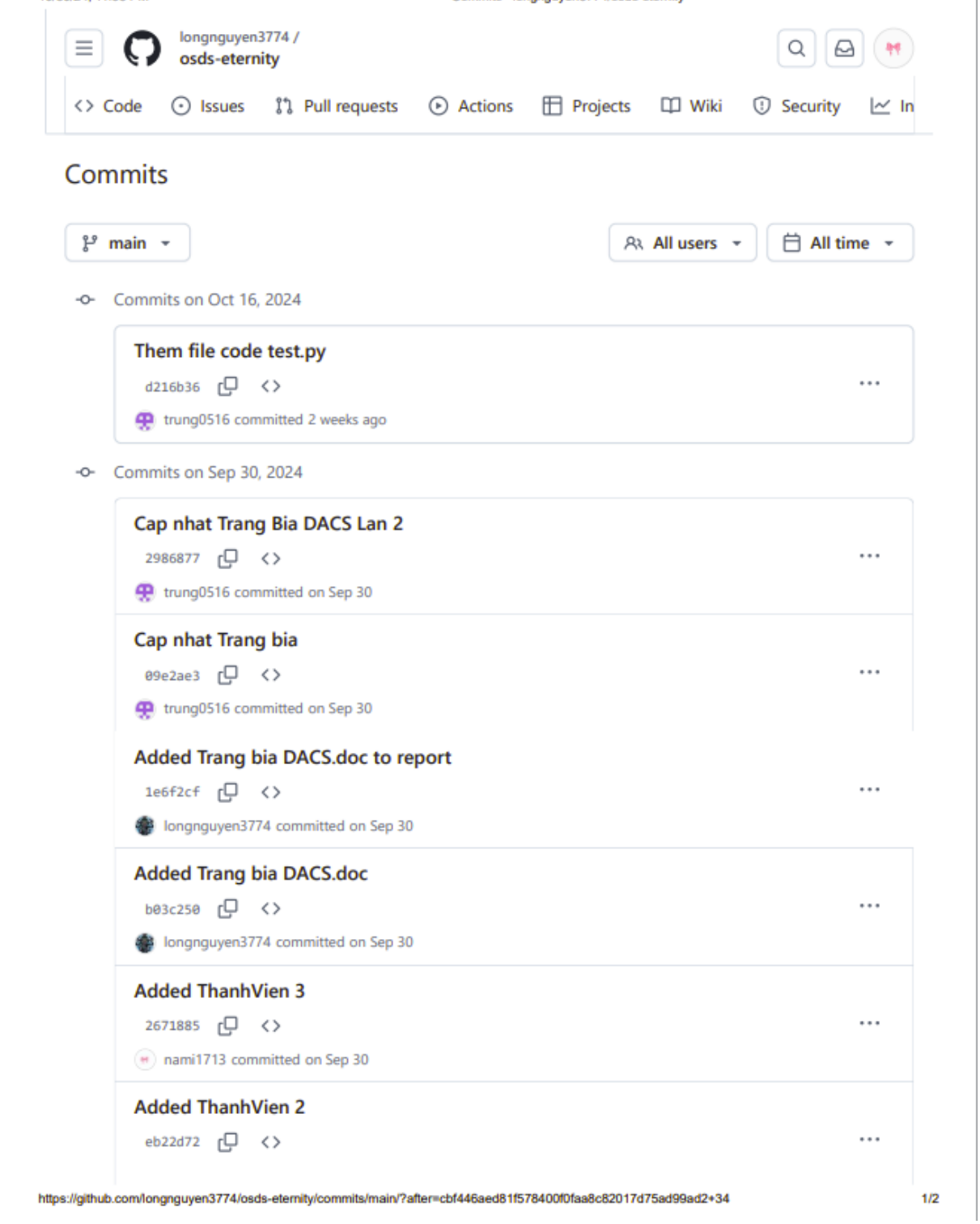
cập nhật file test.py
41e32df trung0516 committed 2 weeks ago

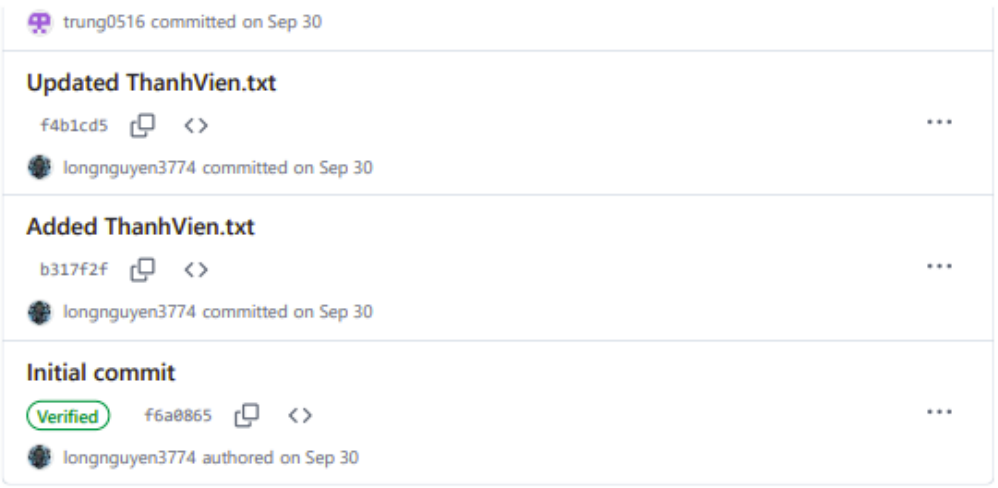
Commits on Oct 16, 2024

Renamed collect_data.py file to etherscan_scaper.py
1a0d901



Previous Next >


longnguyen3774 /
osds-eternity
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
In
Commits
main
All users
All time
Commits on Oct 16, 2024
Them file code test.py
d216b36
trung0516 committed 2 weeks ago
Commits on Sep 30, 2024
Cap nhat Trang Bia DACS Lan 2
2986877
trung0516 committed on Sep 30
Cap nhat Trang bia
09e2ae3
trung0516 committed on Sep 30
Added Trang bia DACS.doc to report
1e6f2cf
longnguyen3774 committed on Sep 30
Added Trang bia DACS.doc
b03c250
longnguyen3774 committed on Sep 30
Added ThanhVien 3
2671885
nami1713 committed on Sep 30
Added ThanhVien 2
eb22d72



Previous Next