

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



BÁO CÁO BÀI TẬP LỚN
MÔN LẬP TRÌNH NÂNG CAO
(CO2039)

HIỆN THỰC WEBSITE TRA CỨU SAO KÊ

GVHD: Lê Đình Thuận

—o0o—

SVTH1: Vũ Ngọc Thuận (2112394)

SVTH2: Hoàng Văn Hải (2111134)

SVTH3: Phan Thanh Bình (2210332)

SVTH4: Lê Ngọc Tâm (2213018)

SVTH5: Nguyễn Hồng Minh (2212059)

TP. HỒ CHÍ MINH, THÁNG 12/2024

TỔNG QUAN

Mục tiêu tổng quát của báo cáo này là thiết kế và hiện thực một website cho phép tra cứu dữ liệu sao kê của Mặt Trận Tổ Quốc Việt Nam (MTTQ VN). Ứng dụng sẽ sử dụng dữ liệu từ một file *.csv và cung cấp giao diện để người dùng có thể tra cứu thông tin sao kê theo một số tiêu chí nhất định.

Kết quả cần đạt được là một website có có giao diện web thân thiện và dễ sử dụng cho phép người dùng tìm kiếm thông tin sao kê theo số tiền, tên người gửi, và nội dung. Giới hạn là có thể sử dụng các thư viện bên thứ ba để hỗ trợ xây dựng các cấu trúc dữ liệu cho việc tìm kiếm, nhưng không được sử dụng hệ thống/ứng dụng dịch vụ độc lập để xử lý toàn bộ việc tìm kiếm.

MỤC LỤC

1	Phân tích yêu cầu	1
1.1	Tổng quan yêu cầu	1
1.2	Phân tích bài toán	1
1.3	Các mục tiêu đặt ra	1
2	Tổng quan lý thuyết	3
2.1	Ý tưởng giải quyết bài toán	3
2.2	Các cấu trúc dữ liệu hỗ trợ giải quyết bài toán	3
2.2.1	Cây AVL	3
2.2.2	Cây tiền tố (Trie Tree)	7
3	Thiết kế hệ thống	10
3.1	Ứng dụng các cấu trúc dữ liệu hỗ trợ tìm kiếm	10
3.2	Kiến trúc hệ thống	10
3.3	Giao diện người dùng	12
3.4	API Endpoints	13
4	Công nghệ sử dụng để hiện thực website	15
4.1	Node.js	15
4.2	Express	15
4.3	Bootstrap	15
5	Tổng kết	17
5.1	Nhận xét	17
5.2	Hướng phát triển	17

Danh sách hình vẽ

1	Dữ liệu sao kê trong file chuyen_khoan.csv	1
2	Cấu trúc của một cây AVL	4
3	Phép xoay trái trên cây AVL	4
4	Phép xoay phải trên cây AVL	5
5	Phép xoay trái-phải trên cây AVL	5
6	Phép xoay phải-trái trên cây AVL	5
7	Minh họa tìm kiếm trên cây AVL với key = 6	6
8	Minh họa một cây tiền tố chứa 4 câu "tôi/bạn là nam/nữ"	7
9	So sánh 2 Trie khi chèn câu "tôi là nam"	8
10	Kiến trúc hệ thống	11
11	Giao diện của website tra cứu	12
12	Hộp Tìm kiếm theo giá trị cụ thể	12
13	Hộp Lọc giá trị theo khaonrg nhất định	13
14	Bảng kết quả tra cứu	14

Danh sách bảng

1	So sánh giữa Trie thông thường và Trie được cấu hình lại.	8
---	---	---

1 Phân tích yêu cầu

1.1 Tổng quan yêu cầu

Mục đích chung nhất của bài toán là xây dựng được một website cho phép người dùng tra cứu dữ liệu sao kê từ một nguồn dữ liệu nào đó. Website tra cứu này phải giúp người dùng tra cứu dữ liệu sao kê một cách nhanh chóng so với phương pháp tra cứu dữ liệu thủ công khi nguồn dữ liệu cần tra cứu là quá lớn.

1.2 Phân tích bài toán

Từ yêu cầu tổng quan, dễ thấy rằng bài toán đặt ra mà website được hiện thực cần giải quyết là tra cứu dữ liệu sao kê, bản chất của bài toán tra cứu dữ liệu chính là bài toán tìm kiếm dữ liệu (search data problem)

Dữ liệu cần tra cứu là sao kê ngân hàng. Đây là dữ liệu đặc thù với bản ghi thường là các giao dịch ngân hàng. Một giao dịch như vậy có thể gồm các nội dung như (tham khảo nội dung của file dữ liệu *chuyen_khoan.csv*, minh họa **Hình 1**): thời gian diễn ra giao dịch (dữ liệu dạng *Date-time*), số tiền giao dịch (dữ liệu dạng số), người gửi, người nhận, nội dung gửi,... (dữ liệu dạng văn bản).

	A	B	C	D	E	F	G	H	I	J	K	L
1	date_time	trans_no	credit	debit	detail							
2	01/09/2024	1	3000		0 267515.010924.122904.NGUYEN THI MAO Chuyen tien							
3	01/09/2024	2	10000		0 018806.010924.213139.chuc mung ngay 29 nuoc cong hoa xa hoi chu nghia Viet Nam							
4	02/09/2024	3	10000		0 055464.020924.064157.Ung Ho Nha Nuoct Viet Nam Va Giup do Nguoit dan (by TPBank ChatPay							
5	02/09/2024	4	10000		0 MBVCB.6924605040.gia dinh Dung Thuy Giang chuc to quoc khoe manh, binh an, hanh phuc, th							
6	02/09/2024	5	50000		0 MBVCB.6925071164.mung ngay quoc khanh.CT tu 1028808193 toi 0011001932418 Uy Ban Tru							
7	02/09/2024	6	2000		0 524322.020924.175952.2091945							
8	02/09/2024	7	500000		0 0200970415090220272520240Ppe432198.64042. 202714.NGUYEN THI LAN HANH Quyen gop							
9	03/09/2024	8	10000		0 881384.030924.070324.Ung Ho Nha Nuoct Viet Nam Va Giup do Nguoit dan							
10	03/09/2024	9	200000		0 120167.030924.100642.NGUYEN HOAI NAM ung ho							
11	03/09/2024	10	300000		0 069690.030924.111352.Ung ho dong bao Viet Nam FT24248787369079							
12	03/09/2024	11	370000		0 524035.030924.111445.IBFT TRAN THI MY PHUONG chuyen tien							
13	03/09/2024	12	100000		0 160707.030924.112000.NGUYEN HOAI PHUONG CK BCT TW							

Hình 1: Dữ liệu sao kê trong file *chuyen_khoan.csv*

Với dạng dữ liệu như vậy, bài toán tìm kiếm dữ liệu có thể chia thành hai bài toán nhỏ hơn:

- Tìm kiếm dữ liệu/nội dung cụ thể - thường áp dụng với tất cả các dạng dữ liệu. Ví dụ: tìm kiếm số tiền giao dịch, nội dung (một phần nội dung), ngày giờ giao dịch...
- Tìm kiếm dữ liệu/nội dung theo khoảng giá trị - thường áp dụng với các nội dung có dữ liệu dạng số hoặc ngày giờ. Ví dụ: tìm kiếm theo khoảng tiền giao dịch, khoảng thời gian giao dịch...

1.3 Các mục tiêu đặt ra

- Website hiện thực có giao diện giúp người dùng tra cứu thông tin sao kê theo: ngày giao dịch, số tiền, tên người gửi, nội dung. Cụ thể:

- Tra cứu thông tin theo nội dung cụ thể: tra cứu giao dịch theo ngày giao dịch, số tiền, tên người dùng cụ thể hoặc theo một phần của nội dung giao dịch.
- Tra cứu thông tin theo khoảng giá trị cụ thể: tra cứu các giao dịch diễn ra trong một khoảng thời gian hoặc các giao dịch có số tiền giao dịch nằm trong một khoảng giá trị nào đó.
- Xây dựng các cấu trúc dữ liệu cho việc tìm kiếm, không được sử dụng 1 hệ thống/ứng dụng dịch vụ độc lập xử lý toàn bộ việc tìm kiếm (vd như không được dùng DB để chứa dữ liệu và chỉ query tìm kiếm một cách out-of-the-box)

2 Tổng quan lý thuyết

2.1 Ý tưởng giải quyết bài toán

Với mỗi nội dung cần tìm kiếm, chúng ta duy trì một cấu trúc dữ liệu hỗ trợ riêng cho việc tìm kiếm trên nội dung đó. Khi đó chúng ta chỉ mất nhiều thời gian cho việc tạo cấu trúc dữ liệu thay vì mất thời gian tìm kiếm tuần tự trên nội dung đó.

Với các yêu cầu tìm kiếm dữ liệu phức hợp (tìm kiếm trên nhiều nội dung), kết quả là phép giao (intersection) giữa các kết quả tìm kiếm trên từng nội dung.

2.2 Các cấu trúc dữ liệu hỗ trợ giải quyết bài toán

2.2.1 Cây AVL

a. Định nghĩa

Cây AVL là một cây tìm kiếm nhị phân tự cân bằng (self-balancing Binary Search Tree), trong đó chiều cao của cây con trái và chiều cao của cây con phải của một nút bất kì khác nhau không quá 1.

b. Cấu trúc

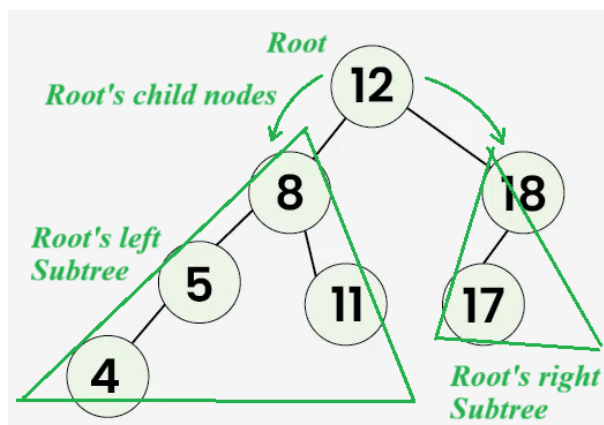
Cây AVL là một cây tìm kiếm nhị phân cân bằng (Balanced Binary Search Tree), nên cấu trúc của nó cũng giống cấu trúc của một cây tìm kiếm nhị phân tự cân bằng (xem **Hình 2**):

- **Nút gốc (Root)**: là nút đầu tiên của cây, từ đó các nút con được mở rộng
- **Nút con (Child Nodes)**: mỗi nút có thể có tối đa hai nút con, hai nút này là nút gốc của cây con trái và cây con phải
- **Cây con trái (Left Subtree)**: chứa các nút có giá trị nhỏ hơn giá trị của nút cha
- **Cây con phải (Right Subtree)**: chứa các nút có giá trị lớn hơn giá trị của nút cha.
- **Chiều cao của cây (height)**: là số lượng cạnh dài nhất từ nút gốc đến nút lá xa nhất; nói cách khác, chiều cao của cây là độ sâu lớn nhất của cây

Cây AVL duy trì một thông số đặc biệt - Hệ số cân bằng: hệ số cân bằng của một nút trong cây AVL được định nghĩa là sự chênh lệch chiều cao giữa cây con trái và cây con phải của nút:

$$BF(N) = \text{height}(N_{\text{left}}) - \text{height}(N_{\text{right}})$$

Trong đó:



Hình 2: Cấu trúc của một cây AVL

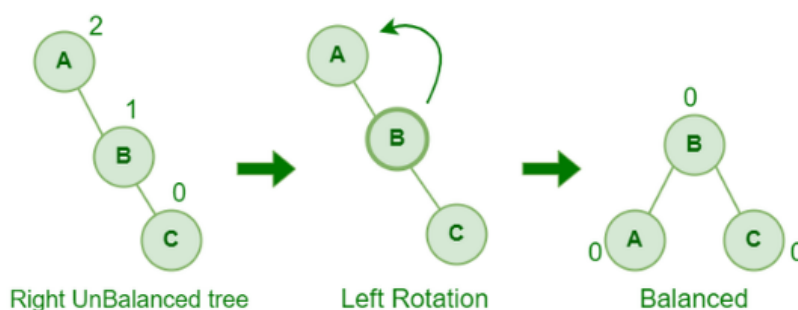
- $height(N_{left})$ là chiều cao của cây con trái của nút N .
- $height(N_{right})$ là chiều cao của cây con phải của nút N .
- $BF(N)$ là hệ số cân bằng của nút N , hệ số này có thể là -1, 0, hoặc 1. Nếu hệ số cân bằng vượt quá các giá trị này, cây sẽ thực hiện các phép quay để cân bằng lại.

c. Đặc điểm

- Cây AVL cân bằng chiều cao, sự chênh lệch giữa cây con trái và cây con phải của bất kỳ nút nào không được vượt quá 1
- Các thao tác tìm kiếm, chèn và xóa đều có độ phức tạp là $O(\log(N))$

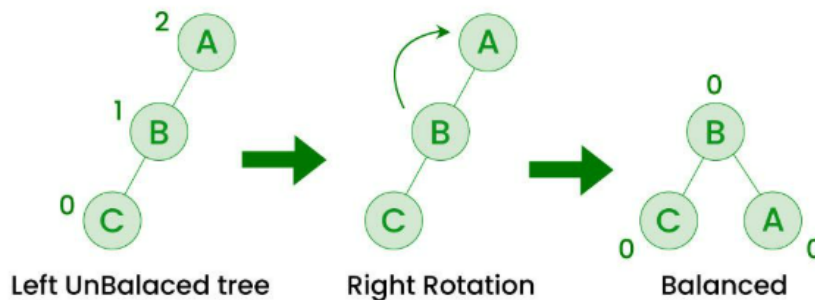
d. Các thao tác trên cây AVL

- **Xoay các cây con:** cây AVL có thể xoay theo 4 cách để tự giữ cho nó cân bằng
 1. **Xoay trái (Left Rotation):** Khi một nút được thêm vào cây con bên phải của cây con bên phải, nếu cây mất cân bằng, chúng ta sẽ thực hiện một phép xoay trái.

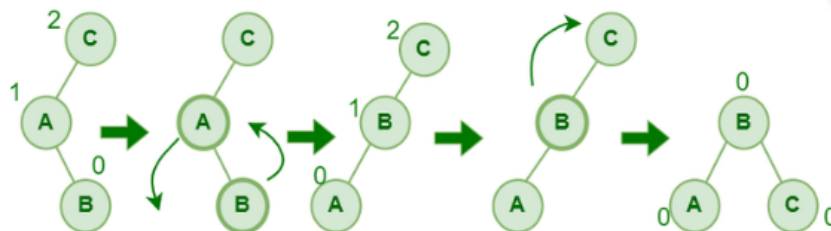


Hình 3: Phép xoay trái trên cây AVL

2. **Xoay phải (Right Rotation):** Nếu một nút được thêm vào cây con bên trái của cây con bên trái, cây AVL có thể mất cân bằng, chúng ta thực hiện một phép xoay phải.



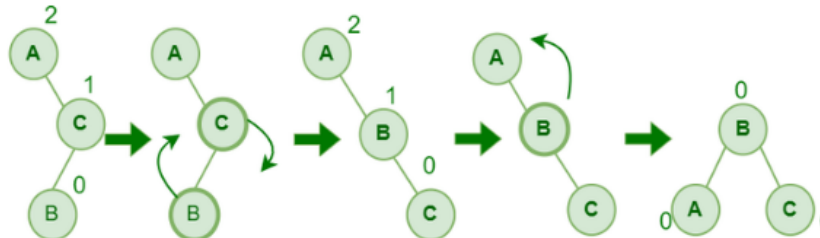
Hình 4: Phép xoay phải trên cây AVL



Hình 5: Phép xoay trái-phải trên cây AVL

3. **Xoay trái-phải (Left-Right Rotation):** Xoay trái-phải là sự kết hợp trong đó thực hiện xoay trái trước sau đó thực hiện xoay phải sau.

4. **Xoay phải-trái (Right-Left Rotation):** Xoay phải-trái là sự kết hợp trong đó thực hiện xoay phải đầu tiên sau đó thực hiện xoay trái



Hình 6: Phép xoay phải-trái trên cây AVL

• **Chèn:** các bước thực hiện chèn nút mới vào cây AVL

1. Đặt nút mới được chèn vào là w . Thực hiện chèn BST chuẩn cho w .
2. Bắt đầu từ w , đi lên và tìm nút mất cân bằng đầu tiên. Giả sử:
 - z là nút mất cân bằng đầu tiên
 - y là nút con của z đi trên đường từ w đến z
 - x là nút cháu của z đi trên đường từ w đến z .
3. Cân bằng lại cây bằng cách thực hiện phép quay thích hợp trên cây con có gốc là z . Có thể có 4 trường hợp có thể xảy ra cần được xử lý:
 - y là con trái của z và x là con trái của y (trường hợp trái trái) → Thực hiện phép xoay phải

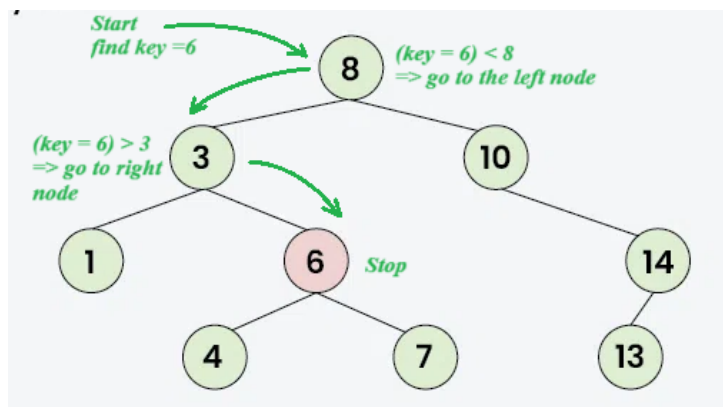
- y là con trái của z và x là con phải của y (trường hợp trái-phải) → Thực hiện phép xoay trái-phải
- y là con phải của z và x là con phải của y (trường hợp phải-phải) → Thực hiện phép xoay trái
- y là con phải của z và x là con trái của y (trường hợp phải-trái) → Thực hiện phép xoay phải-trái

• **Xóa bỏ:** các bước thực hiện xóa bỏ một nút trong cây AVL

1. Giả sử w là nút cần xóa. Thực hiện xóa BST tiêu chuẩn cho w .
2. Bắt đầu từ w , di chuyển lên và tìm nút mất cân bằng đầu tiên. Giả sử
 - z là nút mất cân bằng đầu tiên
 - y là nút con có chiều cao lớn hơn của z
 - x là nút con có chiều cao lớn hơn của y
3. Cân bằng lại cây bằng cách thực hiện các phép xoay thích hợp trên cây con có gốc là z . Có thể có 4 trường hợp có thể xảy ra cần được xử lý tương tự như trong thuật toán chèn nút mới.

• **Tìm kiếm:** quá trình thực hiện tìm kiếm một nút có khóa cho trước giống như trong BST. Giả sử chúng ta muốn tìm nút có khóa X , chúng ta bắt đầu từ gốc. Sau đó:

- So sánh giá trị cần tìm kiếm với khóa của nút gốc. Nếu bằng nhau thì hoàn tất việc tìm kiếm, nếu nhỏ hơn thì phải chuyển đến cây con bên trái; ngược lại chuyển sang cây con bên phải.
- Lặp lại các bước trên cho đến khi không thể duyệt cây được nữa
- Nếu tại bất kỳ lần lặp nào, tìm thấy khóa, trả về nút tìm kiếm được. Nếu không thì trả về **null**.



Hình 7: Minh họa tìm kiếm trên cây AVL với key = 6

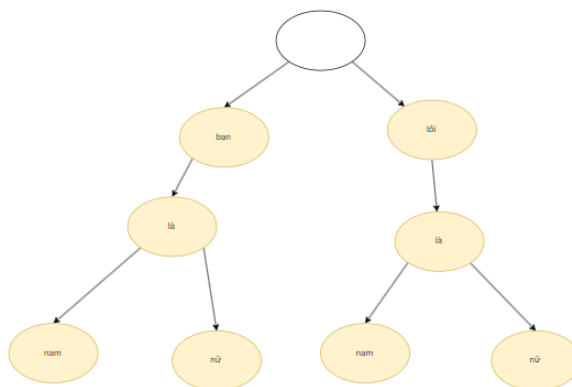
- **Duyệt cây trung thứ tự trong khoảng giá trị (n_1, n_2):** Đối với một nút nhất định, nếu giá trị của khóa lớn hơn n_2 , thì đệ quy gọi vào cây con bên trái và nếu giá trị của nút nhỏ hơn n_1 , thì xử lý cây con bên phải

d. Ưu và nhược điểm• **Ưu điểm:**

- Do cây AVL luôn duy trì sự cân bằng, các thao tác tìm kiếm, chèn, và xóa đều có thời gian thực hiện là $O(\log(N))$. Điều này giúp tăng hiệu suất và đảm bảo thời gian truy cập dữ liệu nhanh chóng.
- Là một cây tìm kiếm nhị phân, các phần tử trong cây AVL có thể được duyệt theo thứ tự tăng dần.

• **Nhược điểm:**

- Việc thực hiện các phép quay để duy trì sự cân bằng đòi hỏi nhiều công sức và kỹ thuật hơn.
- Cây AVL yêu cầu lưu trữ thêm thông tin về chiều cao của các nút, điều này có thể làm tăng lượng bộ nhớ cần thiết.

2.2.2 Cây tiền tố (Trie Tree)**a. Định nghĩa****Hình 8:** Minh họa một cây tiền tố chứa 4 câu "tôi/bạn là nam/nữ"

Trie (hay Prefix Tree) là một cấu trúc dữ liệu dạng cây được sử dụng phổ biến để quản lý và tìm kiếm các dữ liệu dạng chuỗi. Trie đặc biệt hiệu quả trong việc lưu trữ các từ hoặc câu và thực hiện các thao tác tìm kiếm tiền tố nhanh chóng.

Từ Trie bắt nguồn từ re TRIE val, có nghĩa là tìm kiếm hoặc có được thứ gì đó.

b. Đặc điểm (minh họa **Hình 8**)

- Mỗi Trie có một nút gốc (root) trống, có liên kết (hoặc tham chiếu) đến các nút khác.
- Mỗi nút trong Trie đại diện cho một phần trong chuỗi. (1 word)
- Đường dẫn từ gốc đến nút biểu thị tiền tố của chuỗi được lưu trữ trong Trie.

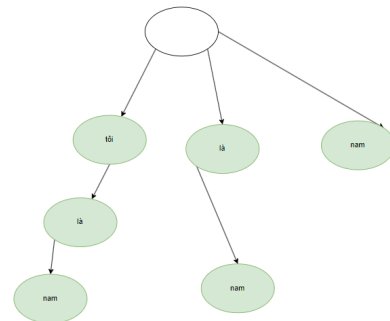
- Các chuỗi có tiền tố giống nhau sẽ có chung các nút tương ứng.
- Trie thường được sử dụng trong các ứng dụng như kiểm tra chính tả, tự động hoàn thành (autocomplete), và tìm kiếm chuỗi.

c. Cấu trúc nút của cây tiền tố được sử dụng để truy vấn theo detail

1. children: Một đối tượng (object) lưu trữ các nút con, mỗi nút con được ánh xạ theo từ khóa (word).
2. ids: Một mảng lưu trữ chứa id của câu chứa đoạn chứa đoạn substring được biểu diễn bởi nhánh từ gốc đến nút hiện tại.



(a) Trie thông thường



(b) Trie được cấu hình lại

Hình 9: So sánh 2 Trie khi chèn câu "tôi là nam"

Đặc điểm	Trie thông thường	Trie được cấu hình lại
Độ phức tạp trung bình	$O(n \cdot k)$	$O(n \cdot k^2)$
Ý nghĩa	n : Số câu, k : Số từ trung bình mỗi câu	n : Số câu, k : Số từ trung bình mỗi câu
Khả năng tìm kiếm	Theo tiền tố	Mọi đoạn substring
Ưu điểm	Nhanh, bộ nhớ hiệu quả	Linh hoạt, hỗ trợ tìm kiếm đa dạng
Nhược điểm	Giới hạn ở tìm kiếm tiền tố	Độ phức tạp cao, tốn bộ nhớ

Bảng 1: So sánh giữa Trie thông thường và Trie được cấu hình lại.

d. Các thao tác trên cây tiền tố

- **Chèn câu mới vào cây tiền tố:**
 1. Chia câu cần chèn thành các từ con (sau khi đã định dạng lại)
 2. Với mỗi từ, chèn toàn bộ các đoạn substring bắt đầu bằng từ đó vào cây.
 3. Với mỗi đoạn substring, bắt đầu từ nút gốc, nếu đoạn substring không có trong cây, thêm nút mới.

4. Đối với mỗi đoạn substring trong câu, lưu trữ toàn bộ thông tin giao dịch (transaction) tương ứng tại các nút trong cây.

- ***Tìm kiếm chuỗi từ cây tiền tố:***

1. Làm chuẩn lại chuỗi con cần tìm.
2. Duyệt qua các nút trong Trie theo từng từ thuộc chuỗi con. Nếu một từ không tồn tại trong Trie, trả về mảng rỗng.
3. Nếu tìm thấy, trả về tất cả các giao dịch (transactions) liên quan tại nút cuối cùng.

e. Ưu và nhược điểm

- ***Ưu điểm:***

- Cho phép tìm kiếm các chuỗi một cách nhanh chóng, không đổi so với Trie thông thường, với thời gian tìm kiếm là $O(m)$, trong đó m là độ dài của chuỗi cần tìm. Đặc biệt hữu ích khi làm việc với các từ điển lớn hoặc các tập hợp chuỗi lớn.
- Tiết kiệm bộ nhớ bằng cách chia sẻ các tiền tố chung giữa các chuỗi. Giúp giảm thiểu sự lặp lại và tiết kiệm không gian lưu trữ.
- Hỗ trợ tìm kiếm tiền tố/ trung tố/ hậu tố một cách hiệu quả

- ***Nhược điểm:***

- Vẫn có thể tốn bộ nhớ hơn so với các cấu trúc dữ liệu khác như bảng băm (hash table) khi làm việc với các chuỗi ngắn hoặc không có nhiều tiền tố chung..
- Phức tạp hơn để cài đặt và duy trì so với các cấu trúc dữ liệu khác.
- Đối với các tập hợp chuỗi ngắn hoặc không có nhiều tiền tố chung, Trie Tree có thể không hiệu quả bằng các cấu trúc dữ liệu khác
- Cần nhập các đoạn substring nguyên từ (không cho phép search chuỗi như "tôi là na")
- Việc xóa một chuỗi khỏi Trie Tree có thể phức tạp và đòi hỏi phải kiểm tra và xóa các nút không cần thiết để tránh lãng phí bộ nhớ.

3 Thiết kế hệ thống

3.1 Ứng dụng các cấu trúc dữ liệu hỗ trợ tìm kiếm

Dựa trên cấu trúc dữ liệu sao kê cần tra cứu (xem **Hình 1**), các thao tác tìm kiếm giao dịch trên dữ liệu có thể bao gồm:

- Tìm kiếm giao dịch diễn ra trong ngày cụ thể; tìm kiếm giao dịch diễn ra trong khoảng thời gian.
- Tìm kiếm giao dịch với số tiền cụ thể; tìm kiếm giao dịch với số tiền nằm trong khoảng nhất định.
- Tìm kiếm các giao dịch có nội dung/một phần nội dung cụ thể.
- Tìm kiếm giao dịch với điều kiện là sự kết hợp của các điều kiện trên

Để hỗ trợ quá trình tìm kiếm, chúng ta duy trì 3 cây AVL cho các cột: ngày giờ (date-time), số tiền gửi (credit) và số tiền nợ (debit) với các khóa của nút lần lượt là ngày giờ, số tiền gửi và số tiền nợ; và duy trì cây tiền tố cho cột nội dung.

Quá trình xử lý dữ liệu như sau:

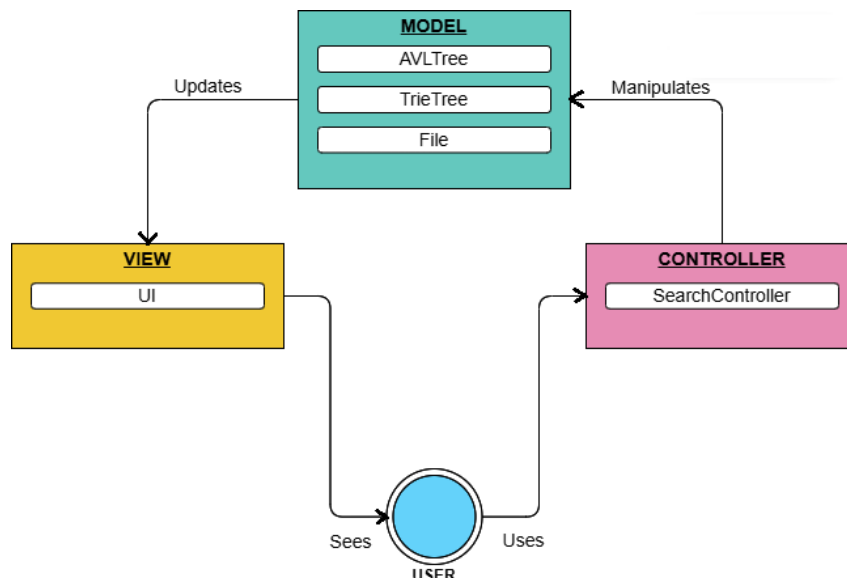
- *Chuẩn bị dữ liệu*: đọc từng bản ghi giao dịch trong dữ liệu sao kê, với mỗi giao dịch thực hiện chèn (insert) cặp dữ liệu (khóa; thứ tự bản ghi) vào cấu trúc dữ liệu phù hợp
- *Tìm kiếm dữ liệu*: thực hiện tìm kiếm (search) trên cấu trúc dữ liệu phù hợp; kết quả thu được thực hiện giao với nhau nếu truy vấn kết hợp nhiều điều kiện trên các cột nội dung khác nhau.

3.2 Kiến trúc hệ thống

Kiến trúc của ứng dụng web được chia thành 3 phần riêng biệt Với nhiệm vụ khác nhau (xem **Hình 11**):

1. *Models*:

- Chịu trách nhiệm quản lý/ tương tác với dữ liệu của ứng dụng
- Chứa các quy tắc nghiệp vụ và logic điều khiển cách dữ liệu được tạo, lưu trữ và sử dụng
- Chứa các module:
 - AVLTree, TrieTree: chứa các cấu trúc dữ liệu hỗ trợ việc tìm kiếm dữ liệu



Hình 10: Kiến trúc hệ thống

– File: quản lý/tương tác với file dữ liệu sao kê

2. View:

- Chịu trách nhiệm hiển thị dữ liệu cho người dùng.
- Lấy dữ liệu từ Model và định dạng nó để hiển thị cho người dùng.
- Tương tác người dùng: View xử lý các yếu tố giao diện người dùng như văn bản, hình ảnh và nút bấm, và cập nhật hiển thị dựa trên các thay đổi trong Model

3. Controller:

- Chịu trách nhiệm xử lý đầu vào và tương tác của người dùng. Nó hoạt động như một trung gian giữa Model và View.
- Xử lý các đầu vào của người dùng, như nhấp chuột và gửi biểu mẫu, và cập nhật Model tương ứng.

Tương tác giữa các thành phần:

- **Tương tác người dùng:** Người dùng tương tác với View, View gửi đầu vào đến Controller.
- Xử lý controller: Controller xử lý đầu vào và cập nhật Model.
- Cập nhật Model: Model cập nhật trạng thái dựa trên hướng dẫn của Controller.
- Làm mới View: View lấy dữ liệu cập nhật từ Model và làm mới hiển thị để phản ánh các thay đổi.

🔍 CHECK BANK STATEMENT 📄

Search by... 🔍

Transaction Date
Choose date...

Credit
Enter credits...

Debit
Enter debits...

Transaction Detail
Enter detail...

Filter ▾

Start Transaction Date
Choose date...

End Transaction Date
Choose date...

Start Credit Value
Enter credits...

End Credit Value
Enter credits...

Start Debit Value
Enter debits...

End Debit Value
Enter debits...

Limit results: 10

Reset

Search

Search for result!

No.	Date	Credit	Debit	Transaction Detail
No data				

Currently on page 0/0

Go to page 0

⏪ ⏩ ⏴ ⏵

© 2024 VNTWebsite. All Rights Reserved.

[Privacy Policy](#) | [Contact Us](#)

Follow us on: [Facebook](#) [Twitter](#)

Hình 11: Giao diện của website tra cứu

3.3 Giao diện người dùng

Các thành phần chính của giao diện:

1. *Hộp Tìm kiếm giá trị cụ thể* (xem **Hình 12**): Cho phép người dùng nhập các giá trị cụ thể theo ngày, số tiền, số ghi nợ và nội dung để tìm kiếm theo giá trị cụ thể

Search by... 🔍

Transaction Date
01/12/2024

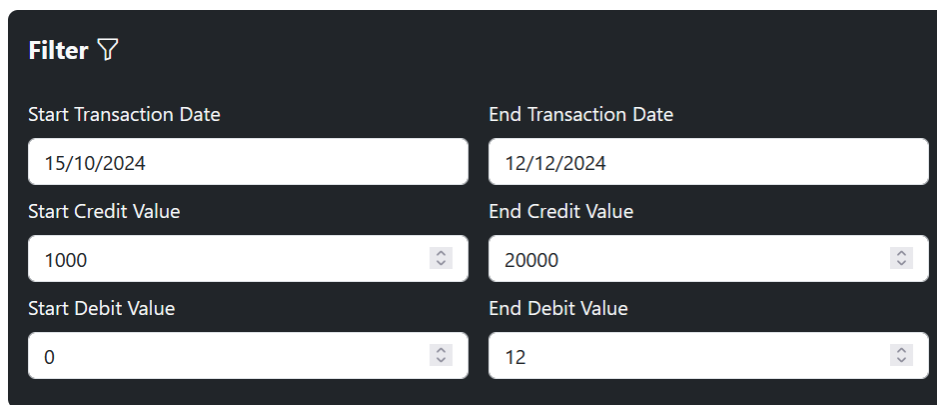
Credit
10000

Debit
100

Transaction Detail
chuyen khoan

Hình 12: Hộp Tìm kiếm theo giá trị cụ thể

2. *Hộp Lọc giá trị* (xem **Hình 13**): Cho phép người dùng tìm kiếm giao dịch theo khoảng giá nhất định: khoảng thời gian giao dịch, khoảng giá trị tiền gửi, khoảng giá trị tiền ghi nợ.
3. *Bảng kết quả tra cứu* (xem **Hình 14**): Hiển thị kết quả tra cứu cho người dùng.



The image shows a dark-themed 'Filter' dialog box. It contains six input fields arranged in a 3x2 grid. The first row is for 'Transaction Date', with 'Start Transaction Date' set to '15/10/2024' and 'End Transaction Date' set to '12/12/2024'. The second row is for 'Credit Value', with 'Start Credit Value' set to '1000' and 'End Credit Value' set to '20000'. The third row is for 'Debit Value', with 'Start Debit Value' set to '0' and 'End Debit Value' set to '12'. Each input field has a small downward arrow icon on its right side, indicating a dropdown menu.

Hình 13: Hộp Lọc giá trị theo khaonrg nhất định

3.4 API Endpoints

Có 2 API Endpoint chính được thiết kế để hỗ trợ việc tra cứu sao kê dữ liệu

1. API tìm kiếm dữ liệu:

- **Mô tả:** API tìm kiếm tất cả các giao dịch thỏa điều kiện tìm kiếm
- **URL:** /search
- **Phương thức HTTP:** GET
- **Tham số:**
 - Tham số truy vấn:
 - * *dateSearch*: ngày giao dịch
 - * *fromDate*: ngày bắt đầu trong khoảng lọc giao dịch theo ngày
 - * *toDate*: ngày kết thúc trong khoảng lọc giao dịch theo ngày
 - * *creditSearch*: số tiền giao dịch
 - * *fromCredit*: số tiền ít nhất mà giao dịch thực hiện
 - * *toDate*: số tiền nhiều nhất mà giao dịch thực hiện *debitSearch*: số ghi nợ giao dịch
 - * *fromDebit*: số ghi nợ ít nhất mà giao dịch thực hiện
 - * *toDebit*: số ghi nợ nhiều nhất mà giao dịch thực hiện
 - * *detailSearch*: nội dung của giao dịch
 - * *limit*: giới hạn số giao dịch với nội dung chi tiết được trả về
- **Cấu trúc phản hồi:**
 - *ids*: mảng các Id giao dịch thỏa mãn điều kiện tìm kiếm
 - *data*: mảng chứa nội dung giao dịch chi tiết. Độ dài của mảng không vượt quá *limit*

2. API truy xuất dữ liệu:

Found 57 result(s)

No.	Date	Credit	Debit	Transaction Detail
392	08/09/2024	1000	0	MBVCB.6977766225.Khokhan.CT tu 0441000722147 toi 0011001932418 Uy Ban Trung uong Mat tran To quoc Viet Nam
658	08/09/2024	1000	0	MBVCB.6979123351.LE MINH DUC chuyen tien.CT tu 1015919694 LE MINH DUC toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW
5758	09/09/2024	1000	0	MBVCB.6987713025.ungho.CT tu 9824331995 LUONG TRUNG GIANG toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW
47782	10/09/2024	1000	0	MBVCB.6991865969.The Ngoc que 74 ung ho thiet hai do bao so 3.CT tu 1025049194 NGUYEN THE NGOC toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW
55424	10/09/2024	1000	0	MBVCB.6992179951.NGUYEN SON HAI DUC chuyen tien.CT tu 9918858685 NGUYEN SON HAI DUC toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW
55887	10/09/2024	1000	0	MBVCB.6992207714.NGUYEN SON HAI DUC chuyen tien.CT tu 9918858685 NGUYEN SON HAI DUC toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW
56083	10/09/2024	1000	0	MBVCB.6992211998.NGUYEN VAN THAI chuyen tien.CT tu 1016806432 NGUYEN VAN THAI toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW
56457	10/09/2024	1000	0	MBVCB.6992220545.NGUYEN SON HAI DUC chuyen tien.CT tu 9918858685 NGUYEN SON HAI DUC toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW
58642	10/09/2024	1000	0	MBVCB.6992328539.cuu tro lu lut.CT tu 0641000032465 VO NGOC TAN toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW
67049	10/09/2024	1000	0	MBVCB.6992691376.VU TRAN KHANH HUY chuyen tien.CT tu 1041003546 DANG CAM KHANH LY toi 0011001932418 MAT TRAN TO QUOC VN - BAN CUU TRO TW

Currently on page 1/6

Go to page 0

« « « » » »

Hình 14: Bảng kết quả tra cứu

- **Mô tả:** API truy xuất dữ liệu giao dịch dựa trên Id của giao dịch - Hỗ trợ phân trang bảng kết quả
- **URL:** /search/get
- **Phương thức HTTP:** GET
- **Tham số:**
 - Tham số truy vấn:
 - * *ids*: chuỗi các Id của giao dịch, cách nhau bởi dấu ", "
- **Cấu trúc phản hồi:**
 - *data*: mảng chứa nội dung giao dịch chi tiết. *limit*

4 Công nghệ sử dụng để hiện thực website

4.1 Node.js

Node.js là môi trường chạy JavaScript mã nguồn mở và đa nền tảng. Node.js cho phép chạy JavaScript trên phía máy chủ, thay vì chỉ trên trình duyệt. Điều này giúp xây dựng các ứng dụng web toàn diện chỉ với một ngôn ngữ lập trình.

Một số đặc điểm chính của Node.js:

- Sử dụng mô hình không đồng bộ và điều khiển bởi sự kiện, giúp xử lý nhiều yêu cầu đồng thời mà không bị chặn.
- Có hiệu suất cao và khả năng thực thi nhanh chóng
- Đi kèm với NPM, một hệ thống quản lý gói mạnh mẽ, cho phép dễ dàng cài đặt, quản lý và chia sẻ các thư viện và module JavaScript.

4.2 Express

Express là một framework web ứng dụng cho Node.js, được thiết kế để xây dựng các ứng dụng web và API một cách nhanh chóng và dễ dàng.

Một số ưu điểm của Express:

- Cấu trúc đơn giản và dễ hiểu, giúp các nhà phát triển nhanh chóng bắt đầu và xây dựng ứng dụng.
- Sử dụng khái niệm middleware, cho phép thêm các chức năng vào ứng dụng của mình một cách dễ dàng
- Cung cấp một hệ thống routing mạnh mẽ, cho phép định nghĩa các tuyến đường (routes) và xử lý các yêu cầu HTTP tương ứng

4.3 Bootstrap

Bootstrap là một framework front-end mã nguồn mở, được sử dụng để phát triển các giao diện web hiện đại và đáp ứng (responsive).

Một số điểm chính về Bootstrap:

- Hỗ trợ thiết kế đáp ứng (responsive design)
- Cung cấp một bộ sưu tập phong phú các thành phần giao diện người dùng

- Đi kèm với nhiều tiện ích CSS giúp bạn dễ dàng áp dụng các kiểu dáng như căn chỉnh, khoảng cách, màu sắc, và nhiều thuộc tính khác mà không cần viết CSS từ đầu.
- Bao gồm các plugin JavaScript dựa trên jQuery để thêm các tính năng động, giúp tăng cường trải nghiệm người dùng

5 Tổng kết

5.1 Nhận xét

Website tra cứu sao kê đã hoàn thành mục tiêu cung cấp một công cụ tiện lợi và hiệu quả cho người dùng trong việc tìm kiếm dữ liệu sao kê trên nguồn dữ liệu do MTTQVN cung cấp.

Với giao diện thân thiện và dễ sử dụng, người dùng có thể dễ dàng tra cứu dữ liệu sao kê. Hệ thống đã được xây dựng trên nền tảng công nghệ hiện đại như Node.js, Express và Bootstrap, đảm bảo hiệu suất cao và khả năng mở rộng trong tương lai.

5.2 Hướng phát triển

- Mở rộng khả năng tra cứu trên nhiều file dữ liệu sao kê cùng một lúc: Hiện tại website chỉ cho phép tra cứu một file dữ liệu nahast định, có thể mở rộng khả năng xử lý và tra cứu dữ liệu trên nhiều file
- Cải Thiện Giao Diện Người Dùng (UI/UX): Nâng cấp giao diện người dùng để tăng tính thẩm mỹ và cải thiện trải nghiệm người dùng. Sử dụng các công nghệ mới như React hoặc Vue.js để tạo ra các giao diện động và tương tác hơn.
- Hỗ Trợ Đa Ngôn Ngữ: Mở rộng hỗ trợ đa ngôn ngữ để phục vụ người dùng từ nhiều quốc gia và vùng lãnh thổ khác nhau.

PHỤ LỤC

- Data source: https://s.thuanle.me/chuyen_khoan.csv
- Github repo: <https://github.com/thuanvu301103/Bank-statement-Search-website.git>